Neural Language Generation for Content Adaptation: Explainable, Efficient Low-Resource Text Simplification and Evaluation

by

Georgeta-Cristina Gârbacea

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2023

Doctoral Committee:

Professor Qiaozhu Mei, Chair
Professor Joyce Chai
Professor Emily Mower Provost
Associate Professor Kevyn Collins Thompson
Associate Professor Lu Wang

Georgeta-Cristina Gârbacea
garbacea@umich.edu
ORCID iD: 0000-0001-5340-594X

DEDICATION

To my beloved parents, Aurelia and Gică Gârbacea,
and to my amazing brother, Alexandru-Marius Gârbacea.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor Prof. Qiaozhu Mei for his wise advice and outstanding support throughout the years. I am deeply grateful to Qiaozhu for giving me the freedom to work on research topics I found interesting, for teaching me to think logically and critically, for his patience whenever results were taking longer than expected, for being an exceptional role model in terms of scientific rigour and work ethic, and for always encouraging me to pursue my dreams and goals. For all this and a lot more, thank you so much Qiaozhu!

I would like to thank Prof. Joyce Chai, Prof. Emily Mower Provost, Prof. Kevyn Collins Thompson and Prof. Lu Wang for kindly accepting the invite to be part of my doctoral committee. I am honoured to have their advice and feedback in this final version of the thesis!

Thoughout my graduate studies, I was fortunate to intern several times with research labs from industry. I will always fondly remember my time as a Research Scientist intern at Google Deepmind London with the Wavenet team, thanks to the amazing reseachers I had the chance to collaborate with: Thomas Walters, Aäron van den Oord, Yazhe Li, Oriol Vinyals, Florian Stimberg, Norman Casagrande, and so many others that inspired me! I am also very grateful to my wonderful mentors from Microsoft Research (MSR), including Tong Wang and Alessandro Sordoni from MSR Montreal, Paul Bennett and Fernando Diaz from MSR Redmond, and Fahad Dogar, Thomas Karagiannis and Antony Rowstron from MSR Cambridge. Fahad, Thomas and Ant, thank you for instilling in me the courage to move across continents to make the PhD dream come true!

Moreover, I am very much indebted to Prof. Maarten de Rijke and Prof. Evangelos Kanoulas for giving me the chance as a MSc student to join their academic research efforts in a vibrant research environment, for mentally stimulating conversations that stirred my curiosity and inspired me to keep learning, as well as for the generous support and invaluable help when applying to PhD programs in the USA and even beyond that. Thank you more than words can express!

I also thank Prof. John Laird for being my academic advisor throughout the years, Prof. Paul Resnick for welcoming me at the School of Information, Prof. Atul Prakash, Prof. Kang Shin, Prof. Eytan Adar, Prof. Paramveer Dhillon for their generosity in offering me helpful advice, and Prof. Jenna Wiens and Prof. Jia Deng for being part of my preliminary exam committee.

I also appreciate the support I received on behalf of CSE administrative staff. I would like to thank Ashley Andreae, Karen Liska, Sylvia Galaty, Jasmin Stubblefield and Angela Farrehi.

I would also like to thank my fellow colleagues, past and present members of the Foreseer lab, including Yue Wang, Cheng Li, Jian Tang, Xin Rong, Samuel Carton, Wei Ai, Shiyan Yan, Teng Ye, Jiaqi Ma, Xuan Lu, Zhuofeng Wu, Jie Gui, Huoran Li, Xuedong Li, Yutong Xie, Yachuan Liu, Xingjian Zhang, and the talented students I had the chance to work with, Mengtian Guo, Yunzhe Jiang and Taoan Lu. I am also thankful for all the inspiring chats with colleagues from CSE, School of Information, and many other departments across the University of Michigan. My thanks to Lucy Jiang Lape, Linfeng Li, Harini Muthukrishnan, Janarthanan Rajendran, Caitlyn Renowden, Vivek Veeriah, Suresh Kumaar, Sandipp Krishnan, Manikandasriram Ramanagopal, Karthik Desingh, Aditya Modi, Hariharan Subramonyam, Srayan Datta, Tawfiq Ammari, Ryan Burton, Shiqing He, Tong Guo, Jane Im, Pei-Yi Kuo, Jiazhao Li, Eshwar Chandrasekharan, Earlence Fernandes, Amir Rahmati, Kassem Fawaz, Mert Pesé, Lajanugen Logeswaran, Ruben Villegas, Chaowei Xiao, Shuang Qiu, Mahmoud Azab, Mohamed El Banani, Maani Ghaffari, Corina Barbalata, Daniela Morar, Bhanu Gupta. In addition, I would also like to thank members of the ILPS lab from University of Amsterdam, who were a source of inspiration in my PhD journey!

I am fortunate to have amazing friends I could rely on in both good and challenging times, thank you for enriching my life with novel perspectives, kindness and good cheer!

Finally, I am immensely grateful to my beloved parents Aurelia and Gică and my dearest brother Alexandru-Marius for the endless love and unwavering support throughout the years! You always encouraged me to dream big and work hard to make dreams come true! I love you so much!

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

ABSTRACT

There are rich opportunities to reduce the language complexity of professional content (either human-written or computer-generated) and make it accessible to a broad audience. As a sub-task of natural language generation (NLG), text simplification has considerable potential to improve the fairness and transparency of text information systems.

Recent approaches to text simplification usually complete the task in an end-to-end fashion, employing neural machine translation models in a monolingual setting regardless of the type of simplifications to be done. These models are limited on the one hand due to the absence of large-scale parallel (complex $\rightarrow$ simple) monolingual training data, and on the other hand due to the lack of interpretability of their black-box procedures. Furthermore, despite fast development of algorithms, there is an urgency to fill the huge gap in evaluating NLG systems in general (including text simplification systems). Indeed, given no clear model of text quality and no agreed objective criterion for comparing the "goodness of texts", the evaluation of NLG systems is inherently difficult. The present work addresses these problems: *i)* sample-efficient approaches to NLG that improve the fairness and transparency of text information systems by adapting their content to the literacy level of the target audience, *ii)* systematic analysis of evaluation metrics for NLG models informed by theory and empirical evidence.

In particular, we show that text simplification can be decomposed into a compact pipeline of tasks to ensure the transparency and explainability of the process; low-resource text simplification can be framed from a task and domain adaptation perspective which can be decomposed into multiple adaptation steps via meta-learning and transfer learning; and evaluators for NLG can be evaluated at scale and compared with human judgements. Beyond the problem of low-resource text simplification, the methodology proposed in this dissertation (explainable decomposition, chain of adaptations to new tasks and domains, and meta-evaluation) may benefit other research areas related to generative artificial intelligence (AI).

CHAPTER 1

Introduction

1.1 Motivation

Over the past decade, advances in training large artificial neural networks have significantly changed the field of Natural Language Processing (NLP), enabling superior performance in a wide range of tasks and application scenarios. Notably, large scale pre-trained large language models based on the Transformer [463] architecture have shown outstanding text generation capabilities, producing realistic, coherent, and fluent texts in many downstream tasks such as dialogue generation, question answering, text summarization, machine translation, text simplification, story/poem generation, etc. Scaling up attention-based architectures in terms of model size, training data and training compute has been one of the key ingredients behind their predictably improved performance, and has given rise to emergent abilities such as few-shot in-context learning [36], zero-shot problem solving [221], chain-of-thought reasoning [477], instruction following [338], or instruction induction [170] from few demonstrations. This shows that with sufficient data, model capacity, and computational resource, text generative models possess remarkable capabilities and an unprecedented level of generality, even referred to as “sparks of artificial general intelligence (AGI)” [39]. Nevertheless, in stark contrast with their impressive abilities to understand and generate human-like language, the same models simultaneously fail on surprisingly simple, naive and intuitive tasks [100], [472]. Given the rapid development of these models, a rigorous scientific understanding of their potential, mixed capabilities, reliability, risks and limitations remains elusive.

Despite changing the landscape of NLP in recent years and their widespread deployment as foundational models for many tasks [29], the effectiveness of current large-scale neural language models is challenged by series of open questions related to the inner workings of these models and their ability to actually understand and process human language. Given that language modeling is only relying on surface form as training data through self-supervised pre-training objectives such as masked language modeling [90] or masked span infilling [249], it is conceivably difficult for such an approach to result in learning the meaning of language from form alone or to further lead to the ability to ground language in the real world [22]. Besides, aspects which require implicit reasoning such as commonsense knowledge, important for building generalizable models that can reason about events, causes and effects, are hard to acquire from raw text only [530]. Another challenge is related to the fact that current large-scale language models can be readily used to perform unconditional text generation, however they provide limited predictable control over attributes of the generated texts [128]; unlike conventional text generation methods which allow fine-grained control over many aspects of the system output including incorporating domain-specific dictionaries, terminology or certain words in the generated output, neural end-to-end approaches remove many of these knobs and switches [361]. Many natural language generation applications not only require text that is fluent, but which also satisfies desirable attributes that govern the syntax, semantics or style of the generated texts; this is particularly important in scenarios focused on the use of NLP for social good with real world constraints, given the ultimate goal of language technologies is to assist humans. In addition, current systems thrive in settings where plenty of data is available, however they struggle in low-resource settings with only a handful of examples.

This raises the question of how to efficiently adapt these large language models (LLMs) to new resource-constrained tasks and domains. By far the most common approach is to leverage the transfer learning paradigm [51], [343], which consists in first pre-training on web-scale data for a surrogate task, then specializing the pre-trained model to the downstream task of interest via fine-tuning [90]. Moreover, the opacity of deep neural networks that underpin current models is a limiting factor in better understanding these models and their capabilities; methods designed for explaining and interpreting their behaviour are much needed to elucidate important aspects of learned models, ensure safety and compliance with regulatory requirements, debug these models, or reveal biases and unintended effects learnt by the model [2]. Another aspect that has been a core research challenge is related to the evaluation of machine-generated text, in particular the lack of consensus on how to measure the "goodness" of generated texts in the absence of a clear model of text quality [78], [160]. Given the lack of well established evaluation measures, natural language evaluations are carried in a rather ad-hoc manner with a lot of variability across the proposed models and tasks on inconsistent benchmarks, resulting in misleading performance measures. This is detrimental to accurately assessing the performance of current natural language generation (NLG) systems, correctly identifying state-of-the-art methods and making research advances.

Given the exciting advances as well as present challenges in the field, in this thesis we broadly focus on enhancing our common understanding as far as the following themes are concerned:

- **Constrained Neural Language Generation:** Controlling the output of current neural network-based models for text generation to account for specific user and task needs is crucial not only for customizing the content and style of the generated language, but also for their safe and reliable deployment in the real world. We present an extensive survey on the emerging topic of constrained neural language generation in which we formally define and categorize the problems of natural language generation (NLG) by distinguishing between conditions and constraints, present constrained text generation tasks, and review existing literature on methods and evaluation metrics for constrained text generation. Our goal is to highlight recent progress and trends, informing on the most promising directions and limitations towards advancing the state-of-the-art of constrained NLG research.

- **Explaining Model Predictions for Accessibility Purposes:** The explainability/ interpretability of a machine learning model has been widely believed to be an indispensable factor to its fidelity and fairness when applied to the real world [232]. Nevertheless, the explainability aspect of text complexity prediction has been overlooked in the literature. We argue it is crucial for improving the transparency and accessibility of text information systems, and to this end we present a promising direction towards a transparent and explainable solution to the problem of text complexity prediction in various domains.

- **Adapting Large Language Models to Low-Resource Tasks and Domains:** In recent years scaling up attention-based architectures in terms of training data and compute has led to considerably improved performance for current NLG systems. Neverthless, such conditions are unrealistic in many NLP tasks and application scenarios where abundant training examples either do not exist or are costly to label, and the computational resource required to train large neural language models from the scratch is a luxury. We investigate how neural language models pre-trained on large-scale corpora can be adapted to new tasks in different domains with limited training examples. We assess the performance of two popular vehicles for few-shot adaptive learning in low-resource tasks and domains, namely meta-learning and transfer learning, and how they can be used to complement each other. Our findings serve as a novel step for bridging the two popular paradigms of few-shot adaptive learning and towards developing more structured solutions to task and domain adaptation.

- **Evaluation of Natural Language Generation:** The variety of NLG models are evaluated with different approaches. By far, human assessments are considered the gold-standard for the evaluation of NLG systems. However, rigorously evaluating NLG systems with real users can be expensive and time consuming, and it does not scale well. Automated evaluations are widely employed for comparing natural language generations with human-written texts, allowing developers to make rapid changes to their systems and automatically tune parameters without human intervention. Despite the benefits, the use of automated metrics in the field of NLG is controversial and their results are often criticized as not meaningful [379]. We present a large-scale, systematic experiment that evaluates the evaluators for NLG. We compare three types of evaluators including human evaluators, automated adversarial evaluators trained to distinguish human-written from machine-generated product reviews, and word overlap metrics in a particular scenario. Results have intriguing implications to both the evaluation and the construction of natural language generators.

In addressing these challenges, we focus on the particular NLG task of **text simplification**, designed to improve the fairness, accessibility and transparency of text-based information systems, and make information accessible to broad audiences. Text simplification has many important practical applications in the real world. In healthcare, the mismatch in writing style and terminology can be mitigated using content adaptation to enhance a layperson's understanding of health information conveyed in professional language by a doctor [1]. In education, adaptive language systems improve student outcomes and bring equal opportunities for learners of all levels in teaching, learning and assessment [302]. In the literature, text simplification refers to reducing the complexity of text at various linguistic levels, ranging all the way through replacing individual words in the text to generating a simplified document completely through a computer agent via lexical [91], syntactic [427] or semantic [197] changes. More recent approaches simplify texts in an end-to-end fashion, employing machine translation models in a monolingual setting regardless of the type of simplifications [516, 153, 460]. Nevertheless, these models are limited on the one hand due to the absence of large-scale parallel (complex $\rightarrow$ simple) monolingual training data, and on the other hand due to the lack of interpretability of their black-box procedures [4]. Given the ambiguity in problem definition, there also lacks consensus on how to measure the goodness of text simplification systems, and automatic evaluation measures are perceived ineffective and sometimes detrimental to the specific procedure, in particular when they favor shorter but not necessarily simpler sentences [324]. While end-to-end simplification models demonstrate superior performance on benchmark datasets, their success is often compromised in out-of-sample, real-world scenarios [79]. In this thesis we aim to make progress towards addressing and advancing these research challenges.

1.2 Thesis Outline

Following the central research challenges outlined above, in this thesis we address topics related to the accessibility, steerability, explainability, adaptability and evaluation of NLG systems. First, we survey the literature on the emerging topic of constrained natural language generation, highlighting recent progress and open problems. Second, we decompose the problem of text simplification into several carefully designed sub-problems, and focus on the explainability aspect of text complexity prediction. Third, we analyze ways in which large pre-trained language models can be adapted to the low-resource tasks and domains of text simplification. Finally, we evaluate existing evaluators for natural language generation at scale, with the goal of establishing which automated evaluation metrics reasonably correlate with human judgements of text quality in the context of online review generation. In what follows we present an overview of each chapter.

- **Chapter 2 – Literature Review: Why is constrained natural language generation particularly challenging?** We focus on the emerging problem of neural natural language generation with constraints. We first define the problem and differentiate between the ambiguous use of *conditions* and *constraints* in natural language generation, including examples that represent instantiations of the constrained neural text generation problem. We then survey approaches, learning methodologies and model architectures employed for generating texts with desirable attributes, and corresponding evaluation metrics. We identify open research problems and limitations of current models. Our goal is to draw clear boundaries between the confusing terminology used in the neural language generation literature, highlight current approaches and discuss how they suffer from the general challenges of constrained text generation. This serves as an informative guide towards advancing meaningful, useful, and safe constrained NLG research.

- **Chapter 3 – Explainable Prediction of Text Complexity: The Missing Preliminaries for Text Simplification** We show that the ambiguous notion of text simplification can be decomposed into a compact, transparent, and logically dependent pipeline of modular subtask that increase the transparency and explainability of text simplification systems, while also improving the generalization of state-of-the-art text simplification models in out-of-distribution settings. We present a systematic analysis of the first two steps in this pipeline, which are commonly overlooked: 1) to predict whether a given piece of text needs to be simplified at all, and 2) to identify which part of the text needs to be simplified. The second task can also be interpreted as an explanation of the first task: why a piece of text is considered complex. We demonstrate that by simply applying explainable complexity prediction as a preliminary step, the out-of-sample text simplification performance of the state-of-the-art, black-box models can be improved by a large margin.

- **Chapter 4 – Adapting Pre-trained Language Models to Low-Resource Text Simplification: The Path Matters** We frame the problem of low-resource text simplification from a task and domain adaptation perspective. We consider the everyday use of text simplification in a wide variety of domains, including news and scientific articles (which naturally contain many subject areas), and view parallel complex-simple English language examples in different domains as samples drawn from a distribution over text generation tasks with varying constraints on the level of text complexity and readability. Once such a distribution is learned from large-scale, general purpose corpora (i.e., a pre-trained language model), it is fast adapted to new tasks and domains (in our case different text simplification scenarios) with few training examples. We consider two approaches to this problem: 1) a standard transfer learning practice that fine-tunes the general language model to the new domains of text simplification with limited in-domain data, and 2) simulate many domain adaptation tasks and use gradient based meta-learning to learn model parameters that can generalize to new tasks, again with few examples. We investigate the performance of two popular vehicles for task and domain adaptation: meta-learning and transfer learning (fine-tuning), in the context of low-resource text simplification that involves a diversity of tasks and domains. Our findings serve as a novel step bridging the two popular paradigms of few-shot adaptive learning and towards developing more structured solutions to task and domain adaptation.

- **Chapter 5 – Judge the Judges: A Large-Scale Evaluation Study of Neural Language Models for Online Review Generation** We present a large-scale, systematic experiment that evaluates the *evaluators* for NLG. We compare three types of evaluators including human evaluators, automated adversarial evaluators trained to distinguish human-written from machine-generated product reviews, and word overlap metrics in a particular scenario, generating online product reviews. The preferences of different evaluators on a dozen representative deep-learning based NLG algorithms are compared with human assessments of the quality of the generated reviews. Our findings reveal significant differences among the evaluators and shed light on the potential factors that contribute to these differences. The analysis of a post experiment survey also provides important implications on how to guide the development of new NLG algorithms.

- **Chapter 6 – Conclusion** We finally conclude this thesis with a summary of our contributions and insights we gathered, as well as open challenges and perspectives for future work.

1.3 Contributions

The contributions of this thesis are summarized as follows:

- We present an extensive survey on constrained NLG. We first define the problem and differentiate between the ambiguous use of conditions and constraints in the literature, including examples that represent instantiations of the constrained neural text generation problem. We then survey approaches, learning methodologies and model architectures employed for generating texts with desirable attributes, and corresponding evaluation metrics.

- We formally decompose the ambiguous notion of text simplification into a compact, transparent, and logically dependent pipeline of sub-tasks, where explainable prediction of text complexity is identified as the preliminary step. We conduct a systematic analysis of its two sub-tasks, namely complexity prediction and complexity explanation, and show that they can be either solved separately or jointly through an extractive adversarial network. Using complexity prediction as a preliminary step reduces the error of the state-of-the-art text simplification models by a large margin.

- We learn how to adapt a pre-trained neural language model to new text simplification contexts that involve a wide diversity of real-world tasks and domains, for which only few in-domain training examples are available. For this purpose, we use widely popular adaptation strategies, namely gradient-based meta-learning and fine-tuning-based transfer learning. We investigate the robustness and efficacy of meta-learning and transfer learning methods in a direct adaptation setting, as well as in a two-stage adaptation setting via a pseudo-stop. We extensively compare these two approaches to task and domain adaptation in our low-resource adaptation settings, and demonstrate that they complement each other. Moreover, performance on out-of-distribution text simplification tasks and domains significantly increases when consecutive stages of adaptation are employed in the correct order.

- We evaluate the evaluators for NLG at scale in the context of generating online product reviews. We compare human-based evaluators with a variety of automated evaluation procedures, including discriminative evaluators and word overlap metrics. We find that generators that fool machine judges easily are less likely to confuse human judges, and vice versa. Word-overlap evaluators tend to have a positive correlation with the human evaluators in ranking the generators. Our results also suggest that when adversarial evaluation is used, the training examples must be carefully selected to avoid false-positives. We also find that when humans are distinguishing fake reviews from real ones, they tend to focus more on the usage of words, expressions, emotions, and other details. This may affect the design of objectives for the next generation of NLG models.

Beyond the problem of low-resource text simplification we are focusing on in this thesis, we believe the methodology proposed in this dissertation (explainable decomposition, chain of adaptations to new tasks and domains, and meta-evaluation) may benefit other areas related to generative AI. These decomposition practices are well aligned with the philosophy of ”chain-of-thoughts” [477] that is a corner stone of large language models, although the latter appears after our work.

CHAPTER 2

Literature Review: Why Is Constrained Neural Language Generation Particularly Challenging?

Recent advances in deep neural language models combined with the capacity of large scale datasets have accelerated the development of natural language generation systems that produce fluent and coherent texts (to various degrees of success) in a multitude of tasks and application contexts. However, controlling the output of these models for desired user and task needs is still an open challenge. This is crucial not only to customizing the content and style of the generated language, but also to their safe and reliable deployment in the real world. We present an extensive survey on the emerging topic of *constrained* neural language generation in which we formally define and categorize the problems of natural language generation by distinguishing between *conditions* and *constraints* (the latter being testable conditions on the output text instead of the input), present constrained text generation tasks, and review existing methods and evaluation metrics for constrained text generation. Our aim is to highlight recent progress and trends in this emerging field, informing on the most promising directions and limitations towards advancing the state-of-the-art of constrained neural language generation research.

2.1 Introduction

Recent advances in the field of natural language generation (NLG) [129] have resulted in models able to produce realistic, coherent, and fluent texts in a multitude of natural language processing tasks. Powerful large scale language models can be readily used to perform unconditional language generation, however these models provide little control over attributes of the *generated* texts. Unlike conventional methods which were able to provide fine-grained control over many aspects of the system output including incorporating domain-specific dictionaries, terminology or certain words in the generated output, neural end-to-end approaches remove many of these knobs and switches [361]. However, imposing constraints on the output generated by these models is crucial for achieving useful and safe language generation in a multitude of real world application scenarios. For example, it can help avoid generic and meaningless responses in dialogue systems [403], personalize dialogue agents based on user features that lead to more engaging and meaningful conversations [513], ensure non-offensive sentence completion and friendly communication [279], intervene on the system output in interactive scenarios where domain specific terminology must be included in the generated texts [75], or aid in creative applications such as poetry generation or assisted story writing [354]. Moreover, controlling a generic pretrained language model in order to satisfy certain desiderata helps avoid generating toxic content, prevents demographic biases, can steer generations towards a desired topic or style [206], and helps communicate intentions in suitable manners for different situations, target audiences and environments [235], [257]. Incorporating prior knowledge and target side constraints in text generative models has numerous applications in many natural language processing areas, including dialogue systems, machine translation, question answering, text summarization, text simplification, image captioning, etc. Unquestionably, constrained text generation is important in many real-world applications, but compared to other instances of natural language generation, constrained text generation using neural networks remains an open challenge.

We identify the following reasons that explain why constrained neural text generation represents a much harder problem compared to other instances of neural text generation: *i) lack of model expressiveness*: current models are not expressive enough to incorporate arbitrary constraints, defined as testable conditions on the *output* text, into the objective function at training time; *ii) lack of suitable evaluation metrics*: while one can verify whether an output satisfies a constraint or not, it is usually hard to measure *to what extent* an output satisfies a constraint, and it is even harder to jointly evaluate this with other properties of the generated text (such as relevance or coherence); *iii) difficulty in constrained optimization*: even if constraints can be expressed and added to the objective function, they are usually non-differentiable, especially at the token level. This is bad as most methods model and generate text as a sequence of tokens; *iv) lack of constrained text generation datasets* that are diverse and representative enough of the variety of practical constraints.

For example, commonly used sequential text generation methods and architectures assume a rigid modeling of the output sequence based on an ordering of words, in which tokens are generated progressively one at a time in a standard left-to-right manner [54]. Such autoregressive models cannot easily express constraints at arbitrary positions in the generated sequence or satisfy constraints involving multiple input objects. In addition to these issues, it is generally more challenging to incorporate multiple and heterogeneous constraints, which conform to given rules, topics, sentiments, lexical constraints, or pre-defined stylistic and content attributes.

Our work focuses on the emerging problem of neural natural language generation with constraints. We first define the problem and differentiate between the ambiguous use of *conditions* and *constraints* in natural language generation, including examples that represent instantiations of the constrained neural text generation problem. We then survey approaches, learning methodologies and model architectures employed for generating texts with desirable attributes, and corresponding evaluation metrics. We conclude with open research problems and limitations of current models. The scope of our work is draw clear boundaries between the confusing terminology used in the neural language generation literature, highlight the main approaches and discuss how they suffer from the general challenges of constrained text generation, and serve as an informative guide and an advocate for solving these general challenges and advancing meaningful, useful, and safe constrained NLG research.

2.2 Problem Definitions

We formally define the problem of natural language generation, accounting for context, conditions, and constraints placed on text generative models. First, we aim to articulate the key difference between condition and constraint since the distinction between these concepts is rather blurred in the natural language processing literature. Given a text generation task defined as $g(X) \rightarrow X'$, we define *condition* as a testable statement of the *input* X, and *constraint* as a testable statement of the *output* X'.

Accounting for the distinction above, we divide the text generation problem into three categories: *i) generic or free-text generation* which we present in Section 2.2.1, *ii) conditional text generation* which we introduce in Section 2.2.2, and *iii) constrained text generation* which we outline in Section 2.2.3. The focus of our work is on the particular problem of *constrained* text-to-text generation, leaving aside text generation tasks from other types of inputs such as data-to-text generation or image-to-text generation which are *conditional* in nature according to our definitions.

2.2.1 Generic/Free-Text Generation

The problem of generic text generation considers the intrinsic history of words generated until the current timestep in the sequence as context, and does not place any external user-defined conditions or constraints on the model output.

Given a discrete sequence of text tokens $x = (x_1, x_2, \ldots, x_n)$ as input where each x_i is drawn from a fixed set of symbols, generic text generation aims to learn the unconditional probability distribution $p(x)$ of sequence x. This distribution can be auto-regressively factorized using the chain rule of probability [24] into a product of conditional probabilities $p(x) = \prod_{i=1}^{n} p(x_i|x_{<i})$ to perform density estimation and generation of language data. When $p(x)$ is modeled by a neural network with parameters θ, the neural network is trained to minimize the negative log-likelihood

$\mathcal{L}(D) = -\sum_{k=1}^{|D|} \log p_\theta(x_i^k | x_{<i}^k)$ over a collection of samples $D = \{x^1, \ldots, x^{|D|}\}$. To generate new samples, each token x_i is iteratively sampled from $p_\theta(x_i | x_{<i})$ and is fed back into the model as the input for the next timestep.

Large scale models for generic text generation show promising abilities to imitate the distribution of natural language and generate long-term realistic and coherent texts, however such free-text generation models place a lot of burden on the generative model to capture complex semantic and structural features underlying the data distribution; this can often result in repetitive, contradictory, and largely randomized generated texts [169]. Notably, the content generated by free-text generative models cannot be controlled with respect to particular attributes and modes of the data distribution. This inability to control which regions of the data distribution are generated is particularly problematic considering there is significant toxicity, hate, bias, and negativity present in the large-scale web crawled datasets text generation models are commonly trained on. Imposing conditions or constraints on the generation process results in safer and more useful generated texts for downstream application tasks [224].

2.2.2 Conditional Text Generation

Conditional text generation manipulates attributes of the generated content depending on specific contexts or user needs, and allows the data generation process to focus on specific modes of the data. Conditioning the generative model on additional information makes it possible to generate texts which satisfy given *input* conditions and meet desired attributes. In the literature conditional text generation is sometimes referred to as context-dependent text generation. While the word context may carry different semantics for different readers, in this survey we consider as context only attributes which are inherently external to the model itself; model intrinsic attributes such as for example, the history of past generated words, is already included in the formulation of generic text generation. For example, context attributes used for conditioning generated texts are the *source sentence* in machine translation, the *conversational history* in dialogue systems, the *input document* in text summarization and text simplification, the *input question* in question answering systems, or contextual information such as *product, time*, and *location* in review generation.

Conditional text generation models add a contextual variable or attribute code c to the probabilistic model $p(x)$ transforming it into a conditional probability model $p(x|c)$, which can be auto-regressively decomposed using the chain rule of probability $p(x|c) = \prod_{i=1}^{n} p(x_i | x_{<i}, c)$. When $p(x|c)$ is modeled by a neural network with parameters θ, the model minimizes the negative log-likelihood loss function accounting for the attribute code c: $\mathcal{L}(D) = -\sum_{k=1}^{|D|} \log p_\theta(x_i^k | x_{<i}^k, c^k)$. Besides generation, conditional models can also be used as generative classifiers to compute $p(c|x_{<i})$ by applying Bayes rule.

2.2.3 Constrained Text Generation

The problem of constrained text generation is focusing on generating coherent and logical texts that do (not) cover lexical concepts (for eg., pre-defined nouns, verbs, entities, phrases or sentence fragments) desired to be (not) present in the *output*, as well as generate outputs that abide to specific format, semantic, syntactic or utility rules to reflect the particular interests of the system user. Constraints impose restrictions on the generative model that must be satisfied by any solution to the optimization problem and their fulfillment can be tested accordingly. In the literature the distinction between conditional, controlled, and constrained text generation is not clearly defined, and these terms are often used interchangeably. In fact, the first work that proposed generating constrained text is actually referring to the task as "controlled" generation [178]. In what follows we formally define the problem of constrained text generation.

Let us consider we are (optionally) given an unordered or ordered set of n concepts $x = \{c_1, c_2, \ldots, c_n\} \in \mathcal{X}$, where $\mathcal{X}$ denotes the space of all concepts, and $c_i \in C$ is a concept belonging to the concept vocabulary C. In addition, let us assume we are also (optionally) given a set of m rules $y = \{y_1, y_2, \ldots, y_m\} \in \mathcal{Y}$, with $y_i \in \mathcal{R}$, where $\mathcal{R}$ denotes the space of all rules, and each y_i is a text generation constraint expressed in logical form. We formulate constrained text generation as learning the structured predictive function $f : \mathcal{X} \cup \mathcal{Y} \to \mathcal{Z}$, where $\mathcal{X} \cup \mathcal{Y} \neq \phi$ which maps a set of concepts and/or constraint rules to a generated sentence. Therefore, constrained text generation methods impose constraints on the generated sentences and produce output in the form of grammatical sentence $z \in \mathcal{Z}$ which contains all concepts present in x and all constraint rules specified in y. The probability $p(z|f)$ can still be modeled autoregressively $p(z|f) = \prod_{i=1}^{n} p(z_i|z_{<i}, f)$; when $p(z|f)$ is modeled by a neural network with parameters θ, the negative log likelihood function can be minimized while leveraging f for constraint satisfaction $\mathcal{L}(D) = -\sum_{k=1}^{|D|} \log p_\theta(z_i^k|z_{<i}^k, f)$.

The matching function f manipulates the probability distribution and indicates to which extent the constraints are satisfied. In the literature, constrained text generation methods can be either *i) Soft-constrained (priming)*, when the matching function f is a soft measure of semantic similarity and only requires the generated sentences to be semantically related to the given constraints, or *ii) Hard-constrained*, when the matching function f is a binary indicator which rules out the possibility of generating infeasible sentences that do not meet the given constraints. Hard-constrained text generation is notably a more challenging task compared to soft-constrained text generation, and it requires designing specialized approaches and architectures to ensure the constraints in the output sentence. In contrast, soft-constrained text generation models are usually easier to design, e.g., with the use of existing copy and attention mechanisms for soft enforcing constraints and annotated keyword-text pairs; nevertheless, some of these soft constraints are likely to be lost during generation, especially if multiple weakly correlated (lexical) constraints must be included [520].

Compared to generic text generation which assumes no conditions on input or output other than existing context, and compared to conditional text generation which places conditions on the input which can be considered at training time, constrained text generation places conditions on the output which is a considerably more difficult and challenging problem to solve. Unlike input conditions, output conditions cannot be considered at training time and their satisfaction is assessed after training has completed by sampling and inspecting the generated outputs. In addition, standard sequence generation architectures are not designed to easily accommodate or incorporate output constraints. Given the model structure itself cannot express output conditions, it becomes challenging to evaluate the extent to which constraints are satisfied by a model, objectively compare and contrast the performance of different models, and measure overall success to inform on progress in constrained natural language generation. Due to these limitations, current methods proposed to address constrained text generation are neither satisfactory nor sufficient. The main machine learning challenge is that it is hard to evaluate the objective function for constrained text generation, and very few works have approached the problem from the prism of editing the objective function to incorporate constraints at training time. Even if constraints were to be added to the objective function itself, constrained optimization would be another challenge. In general, reinforcement learning approaches are used in the context of text generation to optimize non-differentiable reward functions computed at the token level, for eg., BLEU in machine translation or ROUGE in text summarization. However, optimizing such automatic measures that focus on local n-gram patterns often results in deteriorated textual outputs despite increased automatic scores [32], [347]. Moreover, applying reinforcement learning to text generation at the word level leads to difficulty in proper temporal credit assignment for long-term textual rewards [389]. Given that the environment provides only delayed rewards as the agent executes a sequence of actions, it is impossible to know whether the agent succeeds in achieving a task until the end of the episode, at which point the agent needs to determine which of the actions in the sequence are to be credited with producing the resulting reward [125]. Adding constraints on top of existing reinforcement learning issues would be detrimental to the learning process, if not make learning close to impossible: the objective function would be even harder to optimize, rewards would be delayed, sparse and non-informative. Despite these open problems and limitations, we argue neural constrained text generation is an important research area which deserves a lot more attention.

Constrained text generation is useful in many scenarios, such as incorporating in-domain terminology in machine translation [361], improving semantic corectness [14], avoiding generic and meaningless responses in dialogue systems using grounding facts [316], paraphrase generation in monolingual text rewriting [177], [195], incorporating ground-truth text fragments (such as semantic attributes, object annotations) in image caption generation [6], creating a story [109] or poem [135] using a pre-defined set of keywords, or re-writing a user search query as a fluent sentence. Typical attributes used to generate constrained natural language are the *tense* and the *length* of the summaries in text summarization [108], the *sentiment* of the generated content in review generation [318], *language complexity* in text simplification or the *style* in text style transfer applications. In addition, constrained text generation is used to overcome limitations of neural text generation models for dialogue such as genericness and repetitiveness of responses [403], [407].

Nevertheless, generating text under specific lexical constraints is challenging. Common models and architectures employed for natural language generation are autoregressive in nature, generating tokens one by one in a sequential manner from left to right; by design, these models lack fine control over the generated sequence and cannot easily support constraints at arbitrary positions in the output or constraints involving multiple input objects [520], [176]. While for humans it is straightforward to generate sentences that cover a given set of concepts or abide to pre-defined rules by making use of their commonsense reasoning ability, generative commonsense reasoning with a constrained text generation task is more challenging for machine learning models [264].

2.3 NLG Constraints

Natural language generation models place restrictions on the generated output to produce texts that reflect certain user preferences. In Table 2.1 we present NLG tasks distinguishing between conditions and constraints. We broadly group existing constraints into the following categories:

Lexical constraints Lexical constraints serve with the inclusion of specific keywords, phrases or entities at arbitrary positions in the output, and can be specified as a word (a single token) or phrasal constraint (a multi-word phrase). They are useful in tasks such as *dialogue generation*, *machine translation*, *story telling* or *poetry generation*.

Format constraints Format constraints such as number of sentences, length of sentences, order of words, number of syllables, etc. serve to denote preferences on the form and appearance of the generated output. Format constraints are particularly useful in tasks such as *poetry generation* to specify the form of the generated poem, for eg. quatrain or regulated verse, length of the poem, rhyme and rhythm. In *text summarization* or *text simplification*, length constraints define the length of the generated output to be strictly less than the length of the input document, while in *dialogue generation* they help define the level of verbosity of the dialogue agent.

Semantic constraints Semantic constraints are used to define the topic and sentiment of the generated content, or control fine-grained aspects such as removing toxicity. Topic constraints are particularly useful in *dialogue generation*, where the goal is to generate on-topic responses that are safe, non-harmful, unbiased, relevant to the dialogue context and particular user needs; in

Table 2.1: Overview of constrained NLG tasks, differentiating between conditions and constraints.

Task	**Condition**	**Constraint**				
		Lexical	*Format*	*Semantic*	*Syntactic*	*Utility*
Machine Translation	source input	words phrases entities	–	topic sentiment	paraphrase tense gender pronouns	target language politeness factuality/faithfulness
Dialogue Generation	past utterance(s)	words phrases entities	length verbosity	topic sentiment toxicity	paraphrase gender pronouns	politeness personality traits factuality/faithfulness
Text Summarization	input document(s)	words phrases entities	length	topic	paraphrase	factuality/faithfulness
Text Simplification	input text	words phrases entities	length	topic	paraphrase	simpler vocabulary readability factuality/faithfulness
Text Style Transfer	source text	words phrases entities	length	topic sentiment	paraphrase tense gender pronouns	style factuality/faithfulness
Question Answering	input question	words phrases entities	length	topic	paraphrase tense gender pronouns	factuality/faithfulness politeness
Narrative Generation/ Story telling	–	words phrases entities	length	topic sentiment	paraphrase tense gender pronouns	readability factuality/faithfulness style
Poetry Generation	–	words phrases entities	length rhyme rhythm	topic sentiment	paraphrase tense gender pronouns	readability factuality/faithfulness style

story telling or *poetry generation*, topic constraints help define the theme. Generating language that conveys particular positive, neutral or negative sentiment aims to endow artificial agents with human-like traits such as compassion, empathy, and enables agents to react with appropriate emotion in diverse social situations; constraining on a specific sentiment is important in many tasks such as *dialogue generation*, *review generation*, *story telling*, *poetry generation* or *text style transfer*. Furthermore, increasing politeness of a dialogue system or reducing toxicity of generated language are important aspects with respect to human-centered metrics of conversation quality.

Syntactic constraints Syntactically constrained text generation produces sentences with desired syntax by incorporating syntactic templates and rules in the training of the text generative model. Syntactic constraints are useful in paraphrase generation, where given a sentence and a target syntactic form (e.g., a constituency parse), a system must produce a paraphrase of the sentence whose syntax conforms to the target [183]. Generating texts that convey the same meaning but with different expressions has numerous applications in many natural language generation tasks, including monolingual transduction tasks such as *text simplification*, *text compression*, or *text style transfer*, as well as in tasks like *text summarization*, *machine translation* or *question answering* where alternative ways of expressing the same information capture the inherent language variations.

Utility constraints Utility constraints capture holistic properties of the generated output, for eg., stylistic, readability, faithfulness and politeness aspects. Preserving the information content of texts while manipulating attributes such as style, readability level, personality traits of the user or specific gender pronouns allows to customize generated texts to different audiences and make them relevant in a wide variety of end-user applications. Stylistic constraints are immediately relevant to the task of *text style transfer*, with applicability in many tasks, including *dialogue generation*, *machine translation*, *text simplification*, *story telling*, *poetry generation*, *review generation*.

Constraining text generation on attributes such as readability and level of text complexity serves to adapt the generated output to users of different age, backgrounds and educational levels. Reducing complexity of texts while preserving the information content is the main goal of *text simplification*; in addition, in tasks such as *dialogue generation*, *text summarization*, *story telling*, *poetry generation*, *question answering* it is important to customize texts for various literacy levels.

In many languages the degree of politeness is an important aspect of inter-personal communication, and honorifics are used to express courtesy, social distance, or the relative social status between the speaker and their addressee(s) [406]. Politeness constraints on the output are used in *machine translation*, *dialogue generation*, *story telling*, and *text style transfer*.

Faithfulness constraints enforce similarity between a generated text sequence and its corresponding input, requiring models to generate texts that are faithful, factual and preserve the original information content. Such constraints are important in many tasks, including *text summarization*, *machine translation*, *text simplification* or *dialogue generation*, where models are vulnerable to producing hallucinated content.

Language constraints are useful when translating texts between different languages such as in *machine translation*, or from complex language into simple language such as in *text simplification*.

2.4 Constrained Natural Language Tasks

In what follows we briefly describe natural language generation tasks, differentiating between conditions and constraints.

Machine Translation Machine translation is focusing on the automatic translation of textual content from one language into another language, and is a typical example of both *conditional and constrained text generation*, as it conditions on the input text in the source language and constraints the model to generate fluent and faithful output in the target language. Additional constraints can be placed on the degree of formality and politeness, the use of gender-specific pronouns, the inclusion in the target sentence of named entities or specific concepts from the source sentence.

Dialogue Systems A dialogue system, also known as a conversational agent, is a computer system designed to converse with humans using natural language. Dialogue generation is an instance of *conditional text generation* where the system response is conditioned on the previous user utterance and frequently on the overall conversational context. Dialogue generation can also be an instance of *constrained text generation* - it is desirable generated dialogues incorporate explicit personality traits [525], control the sentiment [222], topic, degree of formality and politeness of the generated response to resemble human-to-human conversations. In addition, dialogue responses may need to incorporate text excerpts from past dialogue history or entities such as locations, persons, institutions, etc. From an application point of view, dialogue systems can be categorized into: *i) task-oriented dialogue agents*, designed to help users complete a particular task, or *ii) non-task oriented dialogue agents (chat-bots)* designed to carry entertaining conversations with their users on a wide range of open domains. A common problem in dialogue generation systems is that they tend to generate safe, universally relevant responses that carry little meaning [407], [253], [316]. Moreover, they can fail to take turns asking questions and balance specificity with genericness of the output [403].

Text Summarization Text summarization facilitates a quick grasp of the essence of a document and produces a condensed version of its content, by copy-pasting the relevant portions from the input as in extractive summarization [321], or by generating novel content as in abstractive summarization [386], [322], [402], or via hybrid approaches [275] that combine both techniques. Text summarization is a *conditional text generation task* where the condition is represented by the given document(s); additional conditions are used in remainder summarization to flexibly define which parts of the document(s) are of interest, for eg., remaining paragraphs the user has not read yet, or in source-specific summarization to condition summaries on the specific input source and style of writing, for eg., newspapers, books or news articles. Text summarization is also a *constrained text generation* task considering that the length of the summary is fixed, pre-determined, and strictly less than the original document; this allows to digest information at different levels of granularity and detail according to user needs and time budgets. Moreover, constraints can be placed on specific concepts to include in the summary, such as named entities, or on explicitly picking sentences from the original document as in extractive summarization.

Text Simplification Text simplification is designed to reduce the text complexity, while preserving its original meaning. In the literature, simplification has been addressed at multiple levels: *i) lexical simplification* focused on replacing complex words or phrases with simpler alternatives; *ii) syntactic simplification* alters the syntactic structure of the sentence; *iii) semantic simplification* paraphrases portions of the text into simpler and clearer variants. End-to-end models attempt to combine all these steps. Text simplification is both *conditional and constrained text generation*; we are conditioning on the input complex text to generate a simpler version, accounting for constraints such as higher readability, simpler vocabulary, and shorter sentence length than the complex input.

Text Style Transfer Style transfer has its origins in computer vision applications for image-to-image translation and more recently has been used in natural language processing applications for machine translation, sentiment modification to change the sentiment of a sentence from positive to negative and vice versa, word substitution decipherment and word order recovery [178]. Text style transfer is designed to preserve the information content of a source sentence while altering the way it is delivered to meet desired presentation constraints. Textual content is disentangled from the style in which it is presented, and manipulating stylistic attributes can be done without parallel aligned data between source and target styles. Text style transfer is an instance of both *conditional and constrained text generation* given that we condition on the given source text and constrain the transferred sentences to stylistically match target examples.

Question Answering Question answering systems are designed to find and integrate information from various sources to provide responses to user questions [122]. While traditionally candidate answers consist of words, phrases or sentence snippets retrieved and ranked appropriately from knowledge bases and textual documents [223], answer generation aims to produce more natural answers by using neural models to generate the answer sentence. Question answering is both *conditional and constrained text generation* task; the system conditions on the user question, and simultaneously ensures that concepts needed to answer the question are present in the generated output. Diverse question answering systems are proposed in the literature addressing for eg., medical information needs [479], mathematical questions [400], quiz bowl questions [182], cross-lingual and multi-lingual questions [281]. Notably, in practical applications users are not only interested in learning the exact answer word or phrase, but also in how it relates to background information and to previously asked questions and answers [122].

Narrative Generation/Story Telling Neural narrative generation is an important step towards computational creativity [132] and represents a long-form open-ended text generation task which simultaneously addresses the selection of appropriate content (*"what to say"*) and the surface realization of the generation (*"how to say it"*)[484]. Narrative generation is a *constrained text generation* task that places explicit constraints on concepts to steer the narrative in particular topic directions and expands the few keywords specified as the story title, beginning or ending. While existing models can generate stories with good local coherence, generating long stories is challenging. Difficulties in coalescing individual phrases into coherent plots and in maintaining character consistency throughout the story lead to a rapid decrease in coherence as the output length increases [462]. Hierarchical models for story generation break down the generation process into multiple steps: first modelling the action sequence, then the story narrative, and finally entities such as story characters [110]. Neural narrative generation combining story-writing with human collaboration in an interactive way improves both story quality and human engagement [142].

Poetry Generation The poem generator operates in an interactive context where the user supplies the model with a set of ordered concepts that reflect her writing intent, as well as the format of the poem, for eg. quatrain or regulated verse. Poetry generation is a *constrained text generation* problem since user defined concepts need to be included in the generated poem, and a *conditional text generation* problem given the explicit conditioning on stylistic attributes. For a detailed overview of poetry generation please see [331].

2.5 Constrained NLG Methods

Accounting for the different types of constraints introduced in Section 2.3, we distinguish five methodologies commonly employed in the constrained text generation literature: *i)* decoding approaches, *ii)* fine-tuning approaches, *iii)* discriminative approaches, *iv)* edit-based approaches, and *v)* adapting existing models and architectures to accommodate constraints on the generated output. In what follows we present each approach in detail, outlining the main associated challenges.

2.5.1 Decoding approaches

The most popular approach to text generation in the literature has been supervised learning with task-specific datasets; nevertheless since many real-world applications require diverse and potentially evolving constraints, it is infeasible to annotate task-specific training data for every combination of constraints [366]. Furthermore, even if collecting the data was not a bottleneck, re-training large language models that are extreme in scale for each new constraint or combination of constraints is undesirable. The alternative to fine-tuning language models with task-specific datasets is to enrich decoding algorithms so as to accommodate constraints on the fly. We present decoding approaches to constrained text generation below.

Lexical constraints ***Lexically constrained (guided) decoding*** aims to restrict the search space at decoding time to sequences which contain pre-defined lexical constraints only. These lexical constraints can be specified in the form of a word constraint (a single token) or a phrasal constraint (a multi-word phrase, i.e. a sequence of two or more contiguous tokens). To this end, the beam search decoding algorithm is modified to enforce the inclusion of pre-specified words and phrases in the generated output by allowing the model distribution to not only account for the given lexical constraints, but also to generate parts of the output sequence not covered by the constraints. In general, the decoder can more easily place multiple sequential tokens in a phrasal constraint (where the permutation order is fixed) on the generated output as opposed to placing multiple separate, independent constraints. In addition, the lexically constrained decoding approach assumes lexical constraints are pre-determined, which may not always be the case; if so, the open question is where to get lexical constraints from.

Early work on constrained decoding in machine translation relies on the ***placeholder approach*** designed to recognize identifiable elements (numbers and named entities) in the source sentence, temporarily replace these with corresponding placeholders during preprocessing, and then substitute the assigned placeholders with the original source-language strings during beam search decoding [75]. Nevertheless, such an approach is limited and unable to model the source tokens in target language specific terminology or the vocabulary from a new out-of-distribution domain. ***Prefix decoding*** represents a modification of beam search to first ensure that a user defined target prefix is generated first, and only after build hypotheses for the suffix that maximize the coverage of the remaining source-side tokens. As decoding progresses from left to right, the decoder transitions from a constrained prefix decoding mode to unconstrained beam search. For example, the start of the sentence symbol $<s>$ can be easily included as the first word of a constraint [220], [491]. In the context of text summarization, an essential property of a summarization system is the ability to generate a summary with desired length. ***Grid beam search*** [167] extends beam search decoding to allow for the inclusion of arbitrary target side hard lexical constraints at any position in the generated sequence. Given C input constraints, the algorithm maintains $C + 1$ separate beams $B_0, B_1, \ldots, B_c$ that group together hypotheses which meet the same number of satisfied constraints. Decoding runs similar to beam search, with an additional dimension added to keep track of how many constraints are met by each hypothesis at every timestep; the highest scoring hypothesis in beam B_c is ultimately generated. However, grid beam search is impractical as decoding complexity is linear in the number of constraints, i.e. beam size increases proportionally to the amount of constraints and changes for every sentence. ***Constrained beam search*** [6] guarantees the inclusion of input constraints in the generated sentences by extending beam search with a finite state machine whose states mark completed subsets of the input set of constraints; however, decoding complexity has an exponential cost in the number of constraints, making it infeasible in many applications. ***Dynamic beam allocation*** [361] improves upon the runtime complexity of grid beam search and constrained beam search by decoding with constant complexity $O(1)$ in the number of constraints. The algorithm still groups together hypotheses that have met the same number of constraints by using a single fixed-size beam which is dynamically divided at each time-step according to how many constraints have been met. Despite being more efficient, dynamic beam allocation does not necessarily outperform conventional beam search [264]. In addition, the generation of hypotheses that only partially satisfy a phrasal constraint needs to be aborted to unwind to the tokens in the constraint. ***Neurologic decoding*** [286] modifies beam search to enforce the satisfaction of lexical constraints expressed under predicate logic in conjunctive normal form (CNF). Given the intractability of exhaustive beam search to optimize CNF constraints, the algorithm searches for approximately-optimal output sequences in which all clauses are satisfied, including both positive and negative constraints (i.e. words that must be generated, respectively omitted in the output sequence). The method is applied to cooking recipe generation, where the task is to generate cooking instructions given a dish name and a list of ingredients, and to data-grounded dialogue response generation where a response is generated given a query and a list of facts to convey.

In general, lexically constrained decoding methods have high computational complexity and force the inclusion of specific words in the generated sentence at every timestep of the generation process with no prior examination of these specific words before generation begins [236]; this unnatural way of generating sentences can impact the quality and naturalness of the generated output [271], [361]. In lack of suitable evaluation metrics, there is no commonly agreed criteria for objectively assessing the quality of the generated sentences and conducting comparisons across text generation models.

Format constraints ***Fixed length decoding*** [211] constrains the length of generated summaries in two ways: *i)* by preventing the decoder from generating the end-of-sentence tag until the length of the generated sequence exceeds the desired length, and *ii)* by defining the minimum and maximum length range of the sequence and discarding out-of-range sequences. ***Non-monotonic decoding*** approaches allow tokens to be inserted at any position in the generated sequence during decoding, therefore accommodating flexible orderings of the output. Unlike left-to-right autoregressive generation that produces a single word at a time, non-monotonic decoding can satisfy lexical constraints at multiple locations in the output sequence allowing for highly parallel generation and faster decoding times. Nevertheless, such approaches assume the generated sequence length is known a priori, preventing it from being dynamically adjusted as generation proceeds. Moreover, such models assume conditional independence between output tokens, i.e. tokens are generated independently, may be inconsistent and agnostic to each other. Consequently, this approach may hurt the expressiveness of the model and lead to potential performance degradation, impacting the fluency and naturalness of the output. In addition, non-monotonic sequence decoding approaches can terminate prematurely before constraints are satisfied in the output sequence [520], [176]. The main limitation of this approach is the lack of model expressiveness in accommodating constraints.

Insertion Transformer [438] proposes a flexible sequence generation framework based on repeated insertion operations into an initially empty output sequence until a termination condition is met. The model adopts a progressive masking approach based on token importance in the original text and is trained to generate a missing token between every two tokens in the input. To this end, the original Transformer [463] decoder is modified to allow insertions not just at the end but anywhere in the output sequence. The model can decode sequences serially one token at a time, or it can decode sequences in parallel with simultaneous insertions at multiple locations. A similar approach is considered in InDIGO [150] which extends Transformer for insertion-based decoding with inferred generation order. Token generation order for the output sequence is modeled as a latent variable, and at each decoding step the model predicts both the generated word and its position in the output sequence; nevertheless, strong conditional independence is assumed between the output tokens which hurts output quality. An iterative refinement step based on latent variables is added to the Transformer decoder to refine a target sequence gradually over multiple steps until a predefined stopping criterion is met [241]. Progressive Insertion Transformer [520] uses non-autoregressive modeling based on a top-down progressive structure for lexical hard-constrained text generation. Given lexical constraints as input, the model inserts tokens progressively according to word importance to generate the target sequence, as follows: first it generates high-level words in a sentence such as nouns, adjectives and verbs, then uses these as pivoting points to insert details of finer granularity and finally completes the sentence by adding connecting words which carry less information, such as pronouns and prepositions. Entity Constrained Insertion

Transformer [176] builds upon previous models considering hard lexical constraints in the form of entities in the output sequence. Similar approaches train the Transformer decoder to insert missing tokens in a partially complete sequence without relying on a pre-specified factorization of tokens [54], [151]; based on the information available in the sequence, the insertion-based generative model is able to dynamically infer the remaining parts irrespective of their arbitrary order.

Syntactic and Semantic constraints ***Distributional constraints*** [12] on topic and semantic similarity are used to incorporate source side-information at decoding time in neural conversational systems and encourage the generation of more diverse responses. Moreover, constraints over topics and syntax are used to generate matching or semantically similar statements in response to the user input [329]. Lexically constrained decoding from pre-trained language models aims to steer language models in useful and safe directions so as to minimize the risks associated with these models generating biased, offensive and toxic content [417], [169]. **Energy-based constrained decoding** allows the specification of style and lexical constraints through an energy function and performs differentiable reasoning through gradient-based sampling [366]. The sampling process uses gradients of the energy function to update a continuous relaxation of text data, which is then mapped back to the discrete space of natural language via a discretization approach. For streering the generation towards desired constraints, biases are applied to the logits of the pre-trained model output layer, which is also found to improve the speed of the decoding process [276]. Nevertheless, sampling from energy-based models requires many iterations to converge to plausible text.

2.5.2 Fine-tuning approaches

Semantic and Utility constraints Controlling the output of pre-trained language models is crucial in a wide-range of safety-critical applications, including mental health support chatbots, sentiment controlled text generation, language detoxification, etc. To this end, fine-tuning approaches are used for fine-grained control over individual stylistic aspects (for eg., length, professional and descriptive style, tense, personal voice, gender) and content aspects (for eg., sentiment and topic) of the generated texts [113], [235]. Typically, the pre-trained model is fine-tuned separately for each attribute of interest, which poses the challenge of how to learn disentangled latent representations of style and content in neural language models [191] and isolate the desired attribute from the distribution shift between the generative model and the fine-tuned dataset. The lack of datasets that are diverse and representative of constrained criteria encountered in practice represents an open challenge for fine-tuning pre-trained models.

CTRL [205] uses control codes to trigger the generation of texts that meets user-defined constraints on domain, style, topics, dates, entities, relationships between entities, plot points, and task-related behavior. The pre-defined codes are appended to the beginning of raw text sequences to define task-specific data at training time and create controllable task-specific behaviour at sampling time. Decoding Experts (DExperts) [269] is a decoding-time method for constrained text generation which combines a pre-trained language model with both an "expert" and "anti-expert" language model in a product of experts. The "expert" models desirable aspects of the generated text (for eg., positive sentiment), while the "anti-expert" plays the antagonistic role of modeling undesirable attributes to be avoided (for eg., toxicity); each one of the three language models is conditioned on the same user prompt. While the method highlights the promise of customizing decoding from pre-trained language models in safe and efficient ways, gathering large amounts of toxic data to model undesirable attributes may be challenging. In general, adding negativity to a positive prompt is a much easier task than adding a positive turn to a negative prompt [296].

Fine-tuning approaches in a reinforcement learning setting based on human preferences is used to generate texts with desired attributes from pre-trained language models [537]. Aligning AI models with human preferences is considered crucial for safely deploying artificial systems in the real-world [338], [333]. The reward model is derived from human preferences on text continuations with positive sentiment or vividly descriptive language. Importantly, a KL constraint is used to prevent the fine-tuned model from drifting too far from the pre-trained model and encourage the new policy to remain close to the prior policy. Similar KL control has been used in dialogue systems to retain prior information and penalize divergence from the pre-trained model during RL fine-tuning [187]. Controlled text generation from pre-trained language models is formalized as a constraint satisfaction problem, where pointwise constraints focus on the quality of each individual output while distributional constraints enforce collective statistical properties desirable over the set of all generations [206]. Similar to prior work, a KL penalty term is used to discourage large deviations from the pre-trained language model as a proxy for sample quality. The lack of suitable evaluation metrics is an outstanding challenge in generating high quality outputs.

Pre-trained OpenAI-GPT2 [369] model is used to re-write a story through counterfactual reasoning and generate a narrative consistent with the imposed constraints [365]. In abstractive summarization, OpenAI-GPT2 is used in a reinforcement learning setting which trains the summarization agent to maximize coverage and fluency of the generated content constrained on a pre-defined length [230]. RecipeGPT [157] fine-tunes the GPT-2 pre-trained language model for generating cooking instructions when hard constraints are placed on the recipe title and ingredients; the model can also generate the list of ingredients for a recipe when constrained on the recipe title and specific cooking instructions. Infilling by language modeling is used to complete variable length text spans (e.g. words, n-grams and sentences) by fine-tuning a pre-trained language model on sentence pairs that contain both artificially-masked text and the corresponding original text [95].

While fine-tuning models on task specific datasets has become the dominant paradigm for constrained text generation from large pre-trained language models, these models generally fail to reliably incorporate the underlying constraints in the generated texts even when supervised with large amounts of task-specific examples [286]. Notably, fine-grained constrained text generation is limited even with large scale pre-trained neural networks. The main challenges are the lack of model expressiveness to incorporate constraints and the lack of constrained text generation datasets for fine-tuning these models. Prompt-based text generation from pre-trained language models is a recent trend in the literature, nevertheless this is beyond the scope of our work. For an overview of constrained text generation focused particularly on pre-trained language models, please see [510].

2.5.3 Discriminative approaches

Utility constraints One of the early works to propose constrained generation and manipulation of the generated text learns disentangled latent representations by combining variational autoencoders with attribute discriminators [178]. Semantic structure is imposed on the latent codes by using global discriminators, one for each attribute, to guide the learning of the discrete text generator and force it to allocate one latent dimension per attribute code. The model is used to generate sentences with constrained sentiment and tense.

Weighted decoding [168] relies on a mixture of discriminative models to guide a recurrent generator towards incorporating attributes that enhance the overall coherence, style, and information content of the generated text. The discriminators complement each other and their weighted contributions form the final decoding objective from the generator. Similarly, stylistic configurations are revised and polished for generated poems by adding additional weights during decoding to control the style of generated poem, including the repetition, alliteration, word length, cursing, sentiment, and concreteness [135]. Nevertheless, modifying the scoring function used for generation as in weighted decoding often leads to sacrificing fluency and coherence of the generated text [403]. Selective sampling [471] relies on a sample selector (multilayer perceptron for binary classification) which outputs whether the current sample should be accepted or rejected based on the presence of desired target words that define the output style and topic in the generated sequence. The robustness of evaluation metrics is directly correlated with model performance, therefore it is crucial to focus on developing metrics that capture diverse aspects of text quality during training and sampling time.

Generating texts with desirable attributes from a pre-trained unconditional language model $P(X)$ is a non-trivial task. Most approaches resort to either training from scratch a new conditional model $P(X|a)$ for desired attribute a, or fine-tuning $P(X)$ on additional data representative for the attribute a. Theoretically, rejection sampling could also be used to sample $P(X|a)$ from $P(x)$, but this approach is highly inefficient in practice. Fudge [499] generates text conditioned on a desired attribute a (for eg., topic control in language generation, degree of formality in machine translation, poetry couplet completion) while only accessing the output probabilities $P(X)$ of generative model G. Given an incomplete sequence prefix, the model trains binary discriminative models for one or multiple desired attributes to predict whether the attribute(s) will be fulfilled in the future complete sequence, therefore evaluation is an important challenge. The output probabilities of the discriminator(s) are then multiplied with the output logits of the generator G to adjust the original probabilities of G accounting for desired attribute(s) a and model $P(X|a)$ via a Bayesian decomposition.

PPLM [83] combines a pre-trained language model with attribute classifiers that guide generation towards specific topics and sentiment styles. These classifiers are trained on top of the last hidden layer of the pre-trained language model, and gradients from the classifiers are backpropagated to update the hidden representations of the language model and steer generation in desirable directions. While PPLM achieves fine-grained control of content and style attributes via a simple gradient-based sampling mechanism, the approach is computationally intensive and inefficient as it requires multiple forward and backward passes for each generation step. Plug-and-play methods have been used to control large pre-trained conversational models such as GPT-2 [369] using a variety of styles (positive and negative sentiment) and topics (Question, Sport, Business, Finance) [296]. Undoubtedly, more effort needs to be focused on collecting datasets for constrained text generation that capture many possible real-world constraints.

GeDi [224] guides language generation from large language models towards desired attributes by using generative discriminators to compute classification likelihoods for all candidate next tokens on the fly at generation time. Given a class-conditional language model conditioned both on a desired attribute c^+ and an undesired attribute c^-, GeDi-guided contrastive generation uses the two instances of the model as discriminative classifiers to contrast and filter out common attributes between the two classes c^+ and c^-; then aspects of the desired attribute c^+ are transferred across domains via weighted decoding and filtering. The contrast between a positive and a negative class conditional distribution is employed both at training and inference time to control the bias, toxicity and negativity of GPT-2 [369] and GPT-3 [36].

2.5.4 Edit based approaches

Utility constraints Edit based approaches rely on the key idea that changing only a few words or phrases which are indicative of a particular attribute are sufficient to alter the style of a given piece of text. For example, the sentiment of a sentence can be altered from negative to positive by first identifying negative attribute markers (”bad”, ”worst”, ”disappointed”), deleting these negative attributes while keeping other content words fixed, and then generating the final output via a recurrent decoder which conditions on the extracted content words and the target attribute [257]. Leaving from the observation that humans write text in incremental passes with multiple revisions, a prototype-then-edit model first samples a prototype sentence from the training corpus and then edits it conditioned on an edit vector [156]. Noticeably, text generation based on editing a prototype is much easier compared to generating text from scratch. Also building upon the ”Delete Retrieve Generate” framework, the Generative Style Transformer [440] incorporates a neural mechanism to delete style attributes from the source sentence based on the attention weights of a Transformer model (Delete Transformer), and then generates sentences in the desired target style by decoding with a pre-trained GPT-2 [367] model.

Lexical constraints Constrained sentence generation by Metropolis-Hastings sampling [307] first inserts all constraint keywords in a template in random order, then samples local edit operations (word replacement, deletion or insertion) to perform at specific positions for improving sentence fluency. The probability of each edit operation being accepted or rejected is determined by a language model, however individually sampling each token results in slow convergence. Instead of randomly sampling edit operations, the gradient of a differentiable objective function is used to determine where and how to edit [410].

2.5.5 Adapting existing models and architectures to accommodate constraints

It is non-trivial to impose constraints on existing deep learning models while maintaining high generation quality since their model architecture is designed to generate sentences sequentially from left to right. While current deep learning models are lacking the expressiveness to incorporate constraints at training time and at arbitrary positions in the generated sequence, well known models and architectures are adapted to accommodate constraints through a set of custom engineered approaches. We present these methods below.

Lexical constraints Current architectures used for language generation produce texts sequentially from the first word to the last word, and it is non-trivial to impose lexical constraints on left-to-right generation while maintaining high output quality for natural and fluent texts. Current workarounds for hard lexically constrained text generation address this limitation by generating texts in a non-monotonic fashion when employing ***forward-backward language models***. The backward language model takes a lexical constraint as input, considers it as the starting point and generates the first half of the sentence backwards conditioned on the topic word, while the forward language model takes as input the sequence generated by the backward generator and produces its sentence completion in normal order conditioned on the backward generated sequence. While the topic word can occur at any position in the sentence, this approach can only generate output constrained on at most one lexical constraint; generating sequences with multiple lexical constraints is an open research problem. Moreover, these approaches adapt existing frameworks for constrained text generation by splitting a sentence into two parts, which is unnatural and also hurts fluency when generating half of the sequence in reverse order.

Given a topic word at an arbitrary position in a scientific paper title, a recurrent language model is tasked with generating both past and future words in the title conditioned on the given topic [317]. Similarly, on-topic dialogue responses that satisfy hard lexical constraints are generated with a "sequence to backward and forward sequences" (seq2bf) model [316] which first predicts a keyword noun that reflects the gist of the response, then decodes the response backward and forward starting from the given word. BFGAN [271] employs GANs for lexically constrained text generation using GANs. The model incorporates three modules, namely a backward generator and a forward generator which collaborate on generating lexically constrained sentences, and a discriminator which guides the joint training with policy gradient of the two generators. BFGAN is used to generate Amazon product reviews and conversational responses with lexical constraints.

Generating a fluent sequence which simultaneously satisfies multiple lexical constraints employs a backward-forward LSTM language model to first generate the sequence from a user-defined verb constraint and then satisfy other lexical constraints by word embedding substitution based on cosine similarity between generated tokens and desired constraints [236]. Nevertheless, the approach assumes a verb constraint is always specified in the set of lexical constraints.

Semantic and Utility constraints Steering neural models in specific directions is achieved by: *i)* ***adding special tokens at the beginning or end of the source text***, *ii)* ***incorporating additional conditions into the decoder hidden states*** and *iii)* ***connecting the conditions directly to the decoder output layer***. A topic aware sequence-to-sequence model is used to generate on-topic conversational responses by conditioning the decoder on specific topic words [494]. Imposing conversational goals on dialogue agents aims to guide the conversation towards a designated target subject by combining coarse-grained topic constraints with discourse-level rules [451]. Generating emotional responses in neural conversational systems is achieved by feeding the emotion category embedding to a sequence-to-sequence decoder [529]. Personalized chit-chat dialogue agents that display consistent personalities, viewpoints and are configurable depending on attributes of the system user are used to produce more personal, specific and engaging dialogue responses [471], [32], [513]. Nevertheless, finding the proper balance between fluency, engagement, consistency and a persistent personality remains an open challenge for current dialogue models due to lack of a measurable objective function and correspondingly suitable evaluation metrics. While we can easily judge whether or not an output satisfies one constraint, it is hard to judge the extent to which (or “how much”) it actually satisfies the constraint, and it is even harder to jointly model/measure multiple constraints. Moreover, accounting for repetition and diversity is important as these models often get stuck in an infinite loop of redundant, dull, generic and universally relevant responses that carry little meaning [255], [403], [316].

For integrating factual knowledge into open-ended conversational systems, factoid and entity-rich web documents are encoded altogether with the conversation history into the same representation which is passed to an attentional neural decoder that generates the response tokens. Similarly, speaker-level representations are integrated into seq2seq conversational models for generating personalized conversation responses [254]. Fact-guided sentence modification for dynamically rewriting, updating or correcting articles according to changing information is an instance of constrained text generation which presents the particular challenge that the rewritten sentence needs to be consistent with an input claim while at the same time preserve non-contradicting content [411]. Given the claim and an old sentence, an updated sentence is produced by first identifying contradictory components in the input sentence, masking these, then using the residual sentence and the claim as input into a two encoder sequence-to-sequence model with copy attention to produce the update sentence consistent with the claim. Syntactically controlled paraphrase generation produces paraphrases of an input sentence by constraining the system on the target syntactic form [183], however not many syntactically constrained datasets to learn from are available.

Controllable story generation based on RNNs is used to influence the story ending valence (whether happy or sad) and the storyline (specified as a sequence of words) [354]. Story-telling methods commonly use a hierarchical approach to thematically consistent story generation, by first generating a prompt describing the topic for the story, and then constraining on the prompt for generating the story content [109]; additionally, constraints on the presence of entities are included as well [70]. Open-domain story generation requires composing coherent natural language texts that describe plausible sequence of events and is more challenging compared to generating stories in a narrow domain given an existing plot.

Unsupervised machine translation methods are adapted for the task of text-style transfer by incorporating stylistic constraints in a neural seq2seq model with attention and ***using a style classifier to guarantee the accuracy of style transfer*** [521], or for control over multiple style attributes, including gender, sentiment or product type [235]. In machine translation, honorifics constraints are important for producing socially appropriate forms of address and controling the level of courtesy [406]; the system user defines the desired level of politeness of the translation, however these user-defined constraints are only soft constraints and can be overridden by the attentional encoder-decoder machine translation system whenever the source text provides strong politeness clues.

For effective imposition of semantic structure in constrained text generation, latent space representations need to be disentangled [191], such that varying an individual latent code will only change a single desired attribute. VAEs can achieve meaningful latent representations with designated semantics when combined with ***attribute discriminators*** and optimized end-to-end with differentiable softmax approximation [178]; this allows to generate sentences with constraints on sentiment and tense. Given an input sequence and a set of labels, sequence transduction with multi-space variational autoencoders [527] generates an output sequence that alters the content of the input sequence according to the constraints specified by the labels; the method is used for morphological inflection in multiple languages. In general, constrained text generation approaches assume that constraints need to be known a priori; however, this is not always possible, for eg., when suggesting alternative phrases for search queries in real-time, or when generating responses in dialogue systems according to the dynamics of the conversational context. Recent constrained text generation approaches control attributes of a generated sequence based on another sentence example: given two sentences X and Y, the goal is to generate a new sentence Z that follows the semantics of X and the syntax of Y. To this end, a VAE model with two latent variables is used to achieve disentanglement in the continuous latent space between syntax and semantics [58], [18]. Topic guided VAEs [474] use a Gaussian mixture model prior where each mixture component corresponds to a latent topic extracted from data as opposed to using pre-defined parameter settings which do not incorporate semantic meaning into the latent codes; the model is used for text summarization with designated topic guidance. Abstractive and extractive sentence compression with VAEs assumes the existence of a background language model from which a latent summary sentence is drawn first, and then the observed sentence is generated conditioned on the latent summary [308]; the model is able to balance copying a word from the source sentence with generating it from the background distribution. Iterative refinement of a sequence to transform it into another sequence with desired attributes exploits geometry of the latent space to produce incremental higher-quality revisions with theoretical guarantees in the combinatorial space of sequence elements [318], [416]. Such latent variable manipulations allow to rewrite modern text in the language of Shakespeare, improve sentence positivity, address word substitution and word order recovery tasks without need for any revision examples. Constraints on the use of metaphor and personification in poems are incorporated in a conditional VAE with a rhetorically controlled decoder trained to emit meaningful and diverse rhetoric and overcome generic sentences [279]. Variational neural machine translation [509] incorporates a continuous latent variable to model the underlying semantics of sentence pairs. Nevertheless, efficiently performing posterior inference and large-scale training during the incorporation of latent variables remains an open challenge for constrained VAEs.

Modifying textual attributes of sentences including sentiment, style, tense, voice, mood and negation is achieved by ***incorporating conditioning information into a neural encoder-decoder model***, and optimizing a reconstruction loss which interpolates between auto-encoding and back-translation components to encourage content compatibility, as well as an adversarial loss which encourages sentence-level stylistic attribute compatibility [280]. The model allows simultaneous conditioning on multiple textual attributes, however the extent to which the generated sentences match the conditioning information requires new objective evaluation metrics for attribute accuracy and content compatibility/preservation.

Style transfer between scientific papers and newspapers is performed with ***separate style decoders***, or by generating both content and style from the same decoder [123]. In poetry generation, it is common to impose hard constraints on rhyme, rhythm, and topic [134], [135]. Given a user-supplied topic, the poetry generation algorithm first generates a large set of on-topic words and phrases, assigns rhyming words and phrases to specific lines, and then combines finite-state machinery with an RNN language model to score plausible poems that meet the desired constraints. While ***augmenting an RNN with a working memory*** to explicitly maintain a limited history of generated topics and context, coherence in meaning and topics across the overall poem remains an important challenge [515]. Constrained recurrent models are also used to generate online product reviews of certain topic, sentiment, style and length [113], affective dialogue responses [136], or for modeling participant roles and topics in conversational systems [305].

Alternative non-autoregressive architectures based on continuous diffusion models are adapted for text generation with semantic and syntactic constraints [500], [8], [144]. Diffusion-LM [258] gradually denoises a sequence of Gaussian noise vectors into word vectors, resulting in a hierarchy of continuous latent representations which enables gradient-based methods to steer the text generation process. Nevertheless, training of diffusion models is slower to converge and decoding from these models takes longer time. To speed up the inference process, adaptive sampling strategies are applied for different generation stages in the context of story generation [452].

Format and Utility constraints Text simplification models parameterized on constraints such as length, amount of paraphrasing, degree of lexical and syntactic complexity are used for generating texts easier to read and understand with simpler grammar and structure [298]. Towards a similar goal of controlling the degree of lexical complexity, the ***training loss function is changed to assign weights to words based on their complexity level*** [328]. In text summarization, constraints on the output sequence length for neural encoder-decoder models are specified as ***length embeddings*** and are passed as additional input to the decoder [211].

Faithfulness in abstractive text summarization is enforced in a seq2seq model by conditioning on both the source text and extracted factual descriptions [46]; this helps avoid generating false facts in the output summary. Hybrid text summarization approaches combine an unsupervised sentence extractor which selects salient sentences from the input document with a sentence abstractor that paraphrases each extracted sentence to overcome limitations of parallel aligned datasets [327].

Reinforcement learning is used in the context of constrained natural language generation to directly optimize non-differentiable reward functions and evaluation metrics. While any user-defined reward function can be employed for training, most frequently optimized metrics with RL are BLEU for machine translation [375], ROUGE for text summarization [375], [351], [488], [127], or human-defined conversation metrics focused on coherence, informativeness, sentiment, politeness, toxicity, question, repetition or semantic similarity [255], [389], [488]. However, manually defined reward functions based on heuristics cannot cover all crucial aspects of a natural realistic conversation [32], [127]. In addition, rewards are commonly modeled at the word level accounting for the probability of generating each word in a sentence [375], [187]; such low-level control makes credit assignment challenging since the number of actions available to the RL agent is equivalent to the number of words in the vocabulary. Defining a global score that measures complex aspects of text quality beyond local n-gram patterns and which can reliably approximate human judgments of text quality remains an open challenge [32].

In the RL framework the generative model is seen as an agent with parameters that define a policy and which interacts with an external environment by taking actions, receives a reward once it reaches the end of a sequence and updates its internal state consequently. To this end, policy gradient methods are used to train text generative models and alleviate issues such as exposure bias and loss functions which do not operate at the sequence level. However, policy gradient algorithms present large variance and generally struggle in settings with large action spaces such as natural language generation. In addition, they take very long time to converge [64] and the improvement in the optimized metrics is not always reflected in human evaluations of text quality. Training RL models to optimize n-gram evaluation measures based on local patterns provides only a limited and myopic perspective of overall text quality and does not necessarily lead to better text quality, overall coherence or discourse structure [32]. Moreover, fine-tuning on such measures may yield deteriorated outputs despite increased automatic scores, while difficulty in constrained optimization with RL often leads to sparse, non-informative and delayed reward signals.

Learning RL rewards from human preferences aims to incorporate human feedback in text generation and teach models to follow human instructions [338], [333], [334]. Neural reward learning schemes train neural teachers that learn to score an ordered sequence of sentences and formulate rewards that guide coherent long text generation [32]; the approach is used for generating cooking recipes given the dish title and the set of ingredients as constraints. Learning-to-rank algorithms are used to approximate ground-truth oracle rewards in extractive multi-document summarization to indicate the quality of a summary or preferences over summary pairs [127]. Machine learnability of human rewards in neural machine translation models is approached by first training reward estimators on rewards collected from offline logs, then integrating these reward estimators in an off-policy RL setting [225]. Similarly, implicit human reactions such as sentiment or length of a conversation are used to learn rewards for fine-tuning off-policy RL models for dialog [187]. Nevertheless, human feedback is noisy, not well-defined, complex and inconsistent. Using RL to improve system outputs with respect to human-centered metrics of conversation quality is highly dependent on developing robust metrics tailored to the particular application domain, for eg. increasing politeness of a technical-support system or reducing toxicity of generated language.

Hard-constrained text generation in a non-monotonic order relies on a tree-based text generation scheme, where a word is generated at an arbitrary position in the sentence, then binary trees of words to its left and right are recursively generated [478]. Learning proceeds in an incremental fashion in an imitation learning framework, where the policy gradually moves from imitating the oracle to reinforcing its own preferences and generating texts without a pre-specified word order. Nevertheless, the time complexity of the approach is $\mathcal{O}(n)$, same as for autoregressive models and the constructed tree does not reflect a high-level to low-level hierarchy of concepts.

2.5.6 Prompting Large Language Models

The paradigm of prompt-based learning, which became popular with the release of OpenAI GPT-3 [36] model, demonstrates it is possible to elicit factual and commonsense knowledge from large language models and steer them towards desired behaviours via a textual prompt. Instead of adapting models to downstream tasks via objective engineering as it is common during fine-tuning, prompt-based learning reformulates downstream tasks to resemble those encountered during the language model pre-training phase where a fill-in-the-blanks objective is used [274]. While prompting allows for manipulating the model behaviour to predict desired output, sometimes even without additional task-specific training, model performance on a given task is highly dependent on the quality of the prompt used to steer the model and how much conditioning text can fit into the model's input. In general, identifying the most appropriate prompt for a task is a challenge in itself. While prompting provides a natural interface for humans to communicate with machines, human users have little knowledge of which instructions are compatible with a given model and need to experiment with a wide range of discrete prompts to find suitable ones that elicit desired behaviours [531]. Given that plain language prompts do not always produce the intended results, automated methods for prompt design are proposed in the literature, including searching over the discrete space of words guided by training data [425], prefix tuning which optimizes a task-specific continuous vector [259], [158], prompt tuning which learns soft prompts via backpropagation [246], [364], natural language prompt engineering where large language models themselves generate meta-prompts for solving a wide range of tasks [382], [531] or inverse prompting which uses the generated text to inversely predict the prompt [538]. Directional Stimulus Prompting [260] guides black-box language models such as ChatGPT [333] towards desired outputs by optimizing a policy model trained to maximize rewards that measure the alignment between the generated text and desired topics and keywords on tasks such as text summarization and dialogue response generation.

While there is not much theoretical understanding behind the reasons why and how prompting works, it is assumed that prompting provides a way to steer large language models in particular directions by helping locate a specific task in the pre-trained model's existing space of learned tasks, phenomenon evidenced by the superior performance of some prompts over others [382]. Nevertheless, prompting large language models is far from sufficient for robust and reliable constrained text generation. Prompting approaches must be employed with caution, as models can deviate from the original prompt, fail to maintain the coherence and produce texts on unrelated topics [538], and even degenerate into toxic text from seemingly innocuous prompts [130]. In improving prompting reliability, it is important to account for generalization outside of distribution, reducing social biases, ensuring fairness to different demographic groups, calibrating output probabilities and updating the model's factual knowledge and reasoning chains [426].

Prompting Considerations Large language models leverage vast amounts of information they learn from web-scale pre-training datasets which they store in their parameters, resulting in improved performance on many knowledge-intensive tasks [36], [333], [334]. Nevertheless, it is important to understand what kind of knowledge LLMs actually capture. In factuality assessments of LLMs, it is found that current systems tend to hallucinate and make up facts [303], [449], [528], [267], and this behaviour becomes more predominant as the rarity of entities increases [312]. There is a strong correlation between the correctness of answering factoid questions and the number of pre-training documents relevant to that question [196]; models are more accurate on instances whose terms are more prevalent in the training data, and struggle on questions containing long-tail terms with low document count. Similarly, mathematical reasoning capabilities are correlated with training data frequency, and the selection of the training corpus does impact the few-shot performance of LLMs [377], [424]. These findings suggest that low-order co-occurrence statistics in the pre-training dataset have a significant impact on model performance, leaving the open question of the extent to which current models generalize beyond the training data. Ideally, a general purpose language model is able to generalize not only to unseen instances of known tasks but to new tasks as well, however current LLMs are found to rely on narrow, non-transferable procedures for task solving specialized to tasks seen during pre-training [489]; in counterfactual settings their performance degrades considerably, indicating overfitting to training tasks.

2.6 Constrained NLG Evaluation

Evaluation of constrained text generation is performed using the same evaluation approaches and methodologies available in the natural language generation literature. In general, evaluation of the generated text is largely an unsolved and notoriously difficult problem [31]. Currently, there is no well-established consensus on how NLG systems should be evaluated, [461], [139], and the lack of meaningful quantitative evaluation metrics to accurately assess the quality of trained models is detrimental to the progress of the field. In the absence of well established evaluation measures, natural language evaluations are carried in a rather ad-hoc manner with a lot of variability across the proposed models and tasks on inconsistent benchmarks, resulting in misleading performance measures. Subjective evaluations based on visual inspection of the generated samples often lack scientific rigour, making it difficult to quantify and judge precisely the quality of a generative model [162]. In what follows we review the main methods for constrained text generation evaluation.

Lexical constraints Measuring how many of the given lexical constraints are included in the generated outputs is done using ***concept coverage*** [264], [286]; the metric is computed as the the average percentage of input concepts that are present in the lemmatized outputs.

Semantic and syntactic constraints Surface similarity based on ***n-gram overlap metrics***, such as BLEU [345], ROUGE [265], METEOR [16] measure to what extent the generative model can preserve content by retaining words commonly shared between the generated output and ground-truth references. Such metrics are commonly used to measure response relevance in dialogue systems [124], [254], translation quality in neural machine translation [406], assess summary quality in text summarization [402]. In general, the correlation between word overlap metrics and true text quality is a widely debated topic [255]. Evaluation metrics based on local n-gram patterns only provide a limited and myopic perspective of overall text quality and are notoriously poor at evaluating dialogue systems [270], [403], [32].

Perplexity [189] based evaluation metrics are used to evaluate and compare language models, and measure the fluency and diversity of the generated samples [296], [32], [254]. Reverse Perplexity [522] and Forward Perplexity [213] scores are calculated by training language models on synthetic samples, respectively real samples, and then using these trained models to measure perplexity real samples, respectively generated samples. Nevertheless, perplexity is a model dependent metric, and "how likely a sentence is generated by a given model" is not directly comparable across different models. Moreover, numerous studies find perplexity to be an inadequate measure of text quality [455], [111], since models with high likelihood can generate low-quality samples, while samples of good quality can present low likelihood. In addition, infinite perplexity can still be obtained from a perfect model even when its ability to generate test sentences is removed [162].

P, R, F1 are used to measure the distance of the generated samples to the real data manifold [290]. When precision is high, the generated samples are close to the data manifold, and when recall is high, the generator outputs samples that cover the manifold well. Metrics that aggregate precision and recall such as F_β, a generalization of the F_1 score, are used to quantify the relative importance of precision and recall [388]. Nevertheless, the data manifold of non-synthetic data is unknown and therefore impossible to compute in practice.

Content diversity measures how different the generated sentences are from each other, by either considering word choice, topic and meaning [465], [138], [181], or by looking at the level of sentence interestingness or unlikeliness [162]. Perplexity on a reference set, n-gram diversity [253] and Self-BLEU [534] are commonly used measures of the diversity of the generated samples. In addition, Backward-BLEU [421] evaluates test data using the generated samples as reference; the higher the score the more diverse the generator output. Lexical diversity [9] calculates the ratio of unique tokens to the total number of generated tokens. Similarly, Distinct-k or Dist-k [253] measures the total number of unique k-grams normalized by the total number of generated k-gram tokens to avoid favoring long sentences. Nevertheless, the Dist-k metric ignores the fact that infrequent k-grams contribute more to diversity than frequent ones and assign same weight to all k-grams that appear at least once. Distinct-1 and Distinct-2 are used to measure the diversity of constrained conversational responses [12], [518] and rhetoric constrained generated poems [279]. Entropy based metrics such as Ent-k [518] reflect the frequency difference of k-grams and to analyze the information content of the generated responses in dialogue systems [409], [316].

Unlike traditional evaluation metrics based on heuristics, learnable metrics train machine learning models on human annotated datasets to learn a scoring function that reproduces human judgements. ***Fully-learnt metrics*** leverage existing datasets of human ratings to learn automated evaluation metrics that fit the human data distribution, and can be tuned to measure specific properties of the generated texts, such as fluency, style, grammaticality, fidelity, etc. Linear regression based on human judgements is used to learn a model for scoring system summaries [356]. RUSE [423] combines sentence embeddings in a multi-layer perceptron regressor model. ESIM [59], [301] feeds the encoded representations of the candidate and the reference sentence into a feedforward regressor. BLEURT [404] fine-tunes BERT [89] on human ratings datasets for similarity score prediction. MAUDE [428] is proposed for the evaluation of online dialogue conversations and leverages sentence representations from pre-trained BERT to train text encoders which can distinguish between valid dialogue responses and fake examples. BARTScore [506] formulates the evaluation of generated text as a text generation task from pre-trained language models and measures the weighted probability of the generated text given another text as input or output. GPT Judge [267] fine-tunes GPT3 [36] model on human annotated data to clasify answers of QA systems as true or false, evaluating factuality and truthfulness. The same evaluation metric, this time based on GPT-4 [334], is used to establish via prompting whether texts generated by GPT-4 are more similar to human-written reference answers or GPT-3 machine-generated texts. GPTScore [121] computes the conditional probability of generating the target text given specific context. FactScore [312] breaks generation into atomic pieces of information and evaluates the factual precision of long-form text by measuring the percentage of atomic facts supported by a reliable knowledge source. Other evaluation metrics based on probabilities inferred from pre-trained masked language models include InfoLM [73], CTRLEval [203], MaskEval [278]. ***Hybrid metrics*** combine learnt elements with human-defined logical rules, for example, contextual embeddings with token alignment rules. BERTscore [514] evaluates generated text against gold standard references using soft-string similarity matches (i.e. cosine similarity) computed on pre-trained contextualized BERT [89] token embeddings. MoverScore [523] combines contextualized representations of system and reference texts with semantic measures of distance computed using Word Mover's Distance [229]; the metric is extended to evaluate multi-sentence texts [69]. Human and statistical evaluation are combined in HUSE [162], an evaluation framework which estimates the optimal error rate of predicting whether a piece of text is human-written or machine-generated. However, a limitation of learned evaluation metrics is that they generally fail to generalize well across different systems [53].

Utility constraints A commonly used approach in the literature to assess whether generated texts have desirable attributes is to rely on an attribute classifier and measure the ***classification score***, i.e. the fraction of outputs generated by the model having the desired attribute [178], [416], [257]. ***Adversarial evaluation*** [33], [199] employs an evaluator trained to distinguish machine-generated from human-written texts, analogous to the discriminator in GANs [146]. On this note, ***pre-trained attribute classifiers*** and ***class-specific discriminators*** measure how well the generated samples match the conditioning labels on attributes such as sentiment, tense, voice, mood and negation [280], [256], [38], guarantee the accuracy of stylistic text transfer [521], [416], or are used to evaluate biases against certain demographics and quantify model fairness in downstream settings [45], [299], [228]. ***GLEU*** [323] was originally proposed for grammatical error correction, and later adopted for the evaluation of text style transfer since both tasks require localized edits to the input sentence; GLEU is found to present a reasonable balance between target style match and content retention [440].

Readability metrics such as Flesch-Kincaid Grade Level [214] and Flesch Reading Ease [118] are used to account for simplicity and measure the reading difficulty of a piece of text. Both metrics are computed as linear combinations of the number of words per sentence and number of syllables per word with different weighting factors. Even though these metrics are frequently used to measure readability, are fast and easy to compute, they should not be used on their own but in combination with metrics that capture the grammaticality and meaning preservation of the generated output [490]. In addition, they were not designed for measuring text readability in scientific or specialized domains, and are only available for the English language.

All constraints While automated evaluation helps assess generated texts quickly and cheaply, the use of automated evaluation metrics is dependent upon their correlation with human judgements of quality [119]. ***Human evaluations*** remain the gold-standard in natural language generation and automated evaluation metrics can only be used as a proxy for human judgements only when there is reasonable correlation with human decisions. Ideally, automated evaluations are carried simultaneously with human annotation studies, and not as a replacement of human evaluations. In text style transfer, human evaluations are conducted to determine how accurately constrained text generation methods identify stylistic textual attributes in the source input and replace these with desired target attributes in generated sentences [440]. In conversational systems, responses generated by open-domain chatbots are evaluated across two dimensions: *i)* humanness, as a proxy for the fluency and coherence of the generated responses, and *ii)* attribute consistency, to determine whether the style and topic enforced by the generation model are well captured [296]. Human evaluations are also carried to determine the plausability of the generated response, as well as to measure its content richness and how much new information it adds to the conversation) [12]. Outputs generated by neural conversational systems are also assessed for quality, style and topic to determine whether the acquisition of styles of famous personalities, characters, or professionals is achievable, and whether the conversational topic can be steered in particular directions [471].

Limitations of current evaluation metrics Given the wide diversity of evaluation paradigms in the field of NLG, it becomes challenging to objectively compare models and research progress when different evaluation metrics are employed in each work. By far, human-quality texts are considered the ground-truth for evaluating the output of NLG systems, serving as an upper bound measure of their performance. However, collecting human-quality texts and/or soliciting human judgements of text quality is a costly and time-consuming process which requires careful design choices. Often times automated metrics that present reasonable correlation with human evaluations of text quality are used as a proxy for human judgements, however these metrics come with their own limitations. A common complaint is the lack of good ways to encode what constitutes human-quality output in an automated metric [68]. In addition, shortcomings of current evaluation metrics include poor correlations with human judgements, lack of interpretability of their scores, the presence of bias in evaluations, poor adaptability across tasks and inability to capture all nuances in a task [209]. In what follows we discuss limitations of existing evaluation metrics hoping to inform on the development of more robust evaluations for NLG systems.

Word Overlap metrics measure the lexical overlap between the model generated text and a set of human-written references. Metrics such as BLEU [345], ROUGE [265] and METEOR [16] allow for fast and inexpensive development cycles and have been widely adopted for evaluating the output of natural language generation systems based on their correlation with human judgements at the time they were introduced, nevertheless their use is not without problems. On the one hand, the choice and quality of references is critical for improving the correlation between human and automated evaluation [120]. Current evaluation metrics are biased towards assigning higher scores to outputs that share a similar style with the reference, therefore collecting only a single style of references fails to reward systems that produce alternative but equally accurate outputs [360]; besides, collecting human-written references for new tasks is costly. On the other hand, these metrics assume that valid machine-generated responses present a significant degree of overlap with ground-truth references; this is problematic for open-ended text generation tasks that require diversity and creativity (for eg., dialogue generation), and in such cases their correlation with human judgements is relatively low [277], [270], [148], [404]. For the evaluation of text simplification, BLEU presents weak or no correlation with grammaticality and meaning preservation for sentence splitting operations, therefore penalizing simpler sentences [441]. In addition, improvements in BLEU do not necessarily reflect an improvement in machine translation quality and there is a huge amount of variation for identically scored hypotheses [43] (i.e. a wide variety of candidate outputs receive the same score when they present the same degree of overlap with the reference although they greatly vary). Word overlap metrics are also insufficient for measuring factual correctness of text summarization and fail to correlate with human judgements of factuality [107], [226], [342]. Even more concerning is that the great majority of automated metrics, and in particular conventional reference-based metrics such as BLEU [345] and CIDER [464], are found to overrate machine-generated text over human-written text even though the machine text falls short of humans [200]. In addition, BLEU and ROUGE fail to accurately measure content quality, capture syntactic errors and do not reflect the reliability of NLG systems [379], [437]. Using such evaluation metrics to compare systems may lead to drawing inaccurate conclusions, gives the false impression of progress to the research community and actively discourages the development of stronger human-like generative models.

Since BLEU is based on n-gram precision, lexical differences between the hypothesis and references are aggressively penalized even when they are similar or synonymous to the reference. Given that no partial credit is given if an n-gram does not exactly match a sub-sequence of the reference, BLEU is also hard to optimize due to the fact that learning objective is flat and cannot hill-climb through intermediate hypotheses that have high semantic similarity or synonymy, but low n-gram overlap [480]. Alternative metrics based on word embeddings are easier to optimize as they output continuous values and capture fine-grained distinctions between similar outputs [480]. When used for measuring the quality of back-translations for data augmentation, BLEU only shows significant improvements for test examples if the source itself is a translation [101]; whenever references are translations and the source itself is natural text, BLEU fails to capture human preference for source original sentences. While the use of multiple references substantially improves reference-based metrics, often times evaluations are conducted using a single human-written reference per instance; in such cases strong referenceless metrics frequently achieve higher correlation with human judgements [378]. Developing evaluation metrics that correlate well with human judgements on an instance level could serve to augment and validate human annotations.

To overcome the limitations of reference-based evaluation metrics, reference-free natural language evaluators are proposed [121], [473]. Simultaneously, evaluating the quality of generated texts based on a form-filling paradigm leverages large language models with chain-of-thoughts [277]: given a prompt that defines the evaluation task and desired evaluation criteria, the language model generates a chain-of-thought with detailed evaluation instructions based on which it will then score the generated text according to the defined criteria. While LLM-based metrics seem to outperform reference-based and reference-free evaluation metrics in terms of correlation with human judgements for open-ended and creative NLG tasks, they are very sensitive to the instructions and prompts given. Moreover, they tend to prefer LLM-generated texts over high quality human-written texts, which leads to biased predictions especially when used as reward signal for improving themselves. Using language models for "self-evaluation" indicates their predictions are well calibrated for token probabilites in-distribution, but they struggle with calibration in settings outside of the data distribution [194].

Model-Based Evaluation metrics are becoming increasingly popular for NLG evaluation due to powerful representations learnt by pre-trained language models and high correlations with human judgements of text quality. However, current language models have well-known flaws and limitations, for example they assign high likelihood to degenerate texts, i.e. output that is bland, incoherent, or repetitive [169], can be insensitive to perturbations such as word order randomization [357], negation [105] or named entity replacements [15], exploit superficial cues through the use of the self-attention mechanism [357] and exhibit naive understanding of the meaning of sentences without complex reasoning [384]. To investigate the extent to which model-based evaluation metrics suffer from the same limitations as black-box pre-trained language models, stress tests are used to complement human correlation tests and detect the blind spots of evaluation metrics [163]. The authors construct a noised hypothesis set by applying different synthetic errors to ground-truth human-written references; if this noised hypothesis set is not scored worse than the original unperturbed set, it means the evaluation metric fails the corresponding stress test. Stress tests reveal that model-based evaluation metrics can be insensitive to errors at the start and middle of the generations when based on pre-trained models that do not encode long-range context, their judgement can be mislead by simply injecting valueless text spans into the hypotheses, are biased towards frequent n-grams, present a self-evaluation bias by unfairly ranking generations from their underlying base pre-trained language model higher than better quality generations from larger models, fail fluency tests (lemmatizing verbs, removing articles, prepositions or tokens at the end of the hypothesis) and consistency tests (sentence switching, replacement or negation). While some model-based metrics perform better than others, it is important to recognize their limitations and use each metric with awareness of its blind spots. To mitigate the risks of drawing inaccurate conclusions based on a single metric, using combinations of evaluation metrics that cover each other's blind spots is recommended. For example, evaluation metrics based on pre-trained language models that encode long range context could be more robust to errors in the beginning or middle of the generations, valueless text span injections can be identified by word-overlap based metrics such as ROUGE [265], biases towards frequent n-grams can be detected by using diversity metrics, and truncation errors can be recognized via precision, recall and F1 scores. In addition, to mitigate unfair biases it is desirable to avoid using the same pre-trained language model for generation as well as base for the evaluation metric, or comparing different pre-trained models using an evaluation metric that relies on one of these models. Finally, adding explainability on top of black-box evaluation metrics can help identify system quality issues and increase trust in the evaluation of NLG systems [244].

Human Evaluation is considered the gold standard for the evaluation of NLG systems, however there is no consensus on how these human studies should be conducted [139], [461]. The large variability in the design of human evaluations leads to difficulty in comparing results across different studies and also impacts the reliability of the inferred conclusions. The lack of consistency in human evaluation can be attributed to different factors such as the level of expertise of human annotators, their cognitive biases, ambiguity of the annotation task itself, or the actual wording of questions and instructions presented to participants, i.e. "how something is asked as opposed to what is asked" [399]. Untrained human evaluators may provide inconsistent results and contradictory reasons behind their judgments, leading researchers to state that "all that's human is not gold" [68]. Unsurprisingly, selecting a different subset of annotators can lead to different conclusions due to variations in individual annotators' understanding of the annotation scheme [5]; in general, it is hard to decompose, interpret and validate crowdworker evaluations [200]. Depending on the evaluation setup, it may be sensible to use qualified evaluators who have gone through an extended training and can provide more reliable annotations. Moreover, improving the robustness and transparency of human evaluation guidelines is essential for increasing the reliability of human annotations. As the fluency of generated texts is improving, it is important to not only focus on surface-level aspects of text quality when conducting human evaluations, but to also assess the informativeness and usefulness of generated texts in downstream settings [68].

Future Outlook While so far we have reviewed limitations of existing evaluation metrics, we would also like to note the metrics that are missing or are under-represented in the literature, particularly metrics for measuring the trustworthiness, factuality, fairness, bias, toxicity, efficiency, diversity, uncertainty quantification, calibration and robustness of text information systems. In addition, it is important for the community to focus on the interpretability aspect of evaluation metrics, particularly for model-based evaluations that currently function as a black-box [244]. In the era of large language models, aspects such as knowledge, reasoning, memorization/copyright and disinformation are becoming increasingly important to quantify and analyze for NLG systems [261]. Special attention also needs to be paid to existing datasets used to evaluate the generalization abilities of state-of-the-art methods to ensure there is no overlap between the train set and the test set; in such cases, evaluations inadvertently measure memorization instead of the model's ability to generalize, giving the false impression of improvements in performance [102]. Large language models in particular are known to memorize parts of their training data, phenomenon which becomes more predominant with increasing the model capacity and the repetition of training examples [47], [377]. On top of this, given that LLMs are trained on web-scale datasets, evaluations are subject to potential data contamination issues [489], [94], [297]. Therefore, interpreting evaluation results must be done with caution accounting for the source pre-training data in determining to what extent current models generalize vs simply memorize training examples [377]; this also highlights the need to reconsider and redefine evaluation schemes for LLMs. Finally, it is important to consider how advances in generative models can benefit and inform the development of more suitable evaluation techniques, and vice versa. Bidimensional leaderboards [200] that simultaneously track progress in language generation models and evaluation metrics can bridge the gap between generation modeling and evaluation research. As generation models continue to improve, it is important to keep reassessing and updating evaluation metrics so that they accurately reflect the target objectives and correlate with human language use in the real world [507].

2.7 Constrained NLG Benchmarks and Datasets

The collection of datasets that capture a wide diversity of constraints and are representative of many real world situations are critical for advancing safe and robust constrained text generation. Existing benchmarks focused on politeness [294], formality [376], sentiment [416], writing style [190] are rather limited in nature and do not offer fine-grained control over stylistic attributes. StylePTB [292] aims to allow compositional transfer over a wider range of fine-grained stylistic constructs, including lexical, semantic, stylistic and thematic transfers.

CommonGen [264] benchmark proposes the task of constrained text generation with generative commonsense reasoning, where given a set of concepts the task is to generate a coherent sentence describing an everyday scenario using the given concepts. To do this successfully, the generative model must reason over commonsense relations between the given concepts (relational reasoning), and infer novel combinations of familiar concepts (compositional generalization). Preliminary analysis shows that current state-of-the-art pre-trained models struggle at the task and generate implausible sentences by a large margin. Other benchmarks proposed in the literature focus on avoiding model hallucinations and assessing the veracity and factuality of current models [164], [27], [448]. TruthfulQA [267] benchmark is proposed for measuring the factual accuracy and truthfulness of QA systems. Surprisingly, in their preliminary experiments the authors find that larger language models are less truthful than smaller language models from the same family, neverthleless they are more informative. RealToxicityPrompts [130] aims to measure the extent to which toxic degeneration of large language models can be avoided, and the effectiveness of steering text generation algorithms away from producing racist, sexist and toxic content. Ideally, we want to have NLG models that are controllable, truthful, informative and perform well in the real world, however current pre-trained large language models can degenerate into toxic texts even from seemingly innocuous prompts. Moreover, performance of current models on existing benchmarks is not necessarily representative of their performance in practice. The research community not only needs better evaluation metrics (as outlined in Section 2.6), but also better benchmarks. Given the fragility of current NLG benchmarking practices, fallacious interpretations can be derived [86]. To minimize the discrepancy between model performance on a given benchmark and its actual usefulness when deployed in real-life situations, benchmarks used for assessing the capabilities of current NLG systems should accurately reflect the end task of interest, as well as the wide diversity of scenarios and constraints encountered in practice. Motivated by the observation that new advances in metrics and models should more directly inform and benefit each other, bidimensional leaderboards [200] are proposed to track progress in both generative models and evaluation metrics for constrained text generation tasks such as machine translation, text summarization and image captioning. Nevertheless, a larger issue in terms of natural language evaluation is the gap between how humans use language in the real world, and what current benchmarks can measure [507]. In addition, many datasets are not an effective indicator of model generalization and real world performance, particularly in the presence of overlap between the train and test sets, leading to inflated evaluation results [102], [94]. Besides, since massive web-based datasets used to train large language models are often "contaminated" with downstream test sets, it is important to conduct in-depth analyses to disentangle genuine progress in natural language understanding/generalization from rote memorization [297]; overlooking the impact of pre-training data can result in misleading interpretations of model performance [377]. Finally, we would like to draw attention on the lack of resources (datasets and evaluation metrics) for many languages other than English.

2.8 Discussion and Open Challenges

In what follows we review the main challenges associated with constrained NLG outlining why these challenges have not been solved yet, and present the most promising research directions.

In our view, constrained text generation is a more difficult problem compared to other instances of text generation. The difficulty arises from a multitude of factors, including lack of model expressiveness which makes it difficult for current models to incorporate constraints into the objective function, lack of suitable evaluation metrics to assess the extent to which constraints are satisfied (which becomes even more challenging in the presence of multiple constraints), difficulty in the constrained optimization of non-differentiable reward functions, and finally lack of constrained text generation datasets that are illustrative of a wide diversity of constraints. Due to these pressing issues, constrained text generation remains an open challenge in the research community. Advancing the state-of-the-art requires considerable more collective and focused effort. Below we identify the most promising directions for advancing the state-of-the-art for safe and robust constrained NLG.

Multiple constraint satisfaction Most approaches proposed for constrained text satisfaction focus on generating sentences that meet one single desired constraint, nevertheless generating sequences that simultaneously satisfy multiple lexical constraints is an important open research problem in text generative models [271], [236], [176]. While incorporating one constraint is already hard enough due to lack of model expressiveness, incorporating multiple constraints poses significant challenges in terms of defining the loss function accounting for all the desired constraints, difficulty in optimizing it and evaluating whether each constraint is satisfied. Approaches that convert the multiple constraint satisfaction problem into allowing the inclusion of pre-specified lexical constraints at decoding time are not optimal either: on the one hand, decoding complexity increases exponentially or linearly in the number of constraints, and on the other hand forcing constraints at every step of the generation process impacts the quality and naturalness of generated texts [361]. Moreover, many model architectures are designed for sequential sentence generation only (vs. non-monotonic text generation) and it is non-trivial to impose decoding time constraints while maintaining optimal text generation quality [307].

Dynamically defined constraints Current approaches to constrained text generation assume there is prior knowledge of the constrained textual attributes and the finite set of values these attributes can take on. Nevertheless, there are situations when it may be desirable to impose constraints dynamically, for eg. in conversational systems depending on the system user's statements, reactions and emotions. When dynamically defining constraints, the main challenges are the lack of model expressiveness and robust ways to evaluate whether these constraints are satisfied. In the literature, controling the realization of a sentence based on another's sentence syntax and semantics is a less explored setting for constrained text generation with dynamic constraints which does not require prior knowledge of all the values the control variable might take on [58]. Disentangled latent space representations of syntax and semantics are essential for the manipulation sentence attributes in tasks such as unsupervised paraphrase generation and syntax-transfer generation [18].

Generative reasoning Current large-scale text generation models display impressive ability to generate fluent texts, nevertheless composing realistically plausible sentences in the presence of constraints remains a significant open challenge. This is illustrative of all challenges associated with constrained text generation, including lack of model expressiveness, lack of suitable evaluation metrics, difficulty in constrained optimization and lack of constrained text generation datasets. Nevertheless, endowing generative models with commonsense reasoning abilities is an important milestone towards advancing machine understanding and intelligence. Generally, the great majority of models proposed in the literature only exploit superficial cues via self-attention to solve NLP tasks, without relying on syntactic information or complex reasoning [357].

Attribute specific datasets The lack of annotated datasets for attribute specific text generation constitutes a bottleneck in the development and adaptation of models for tasks that require fine-grained control over style and topics. For example, in dialogue systems the absence of attribute annotated conversational datasets that can be used for fine-tuning large scale pre-trained models limits control over the generated responses for a desired attribute [296]. Moreover, such attribute annotated datasets can help with the personalization of dialogue systems, make dialogues safe, supportive and engaging [408], [513]. Personalized dialogue agents that display consistent personalities and viewpoints overcome the unsatisfying experience of a persona-free chit-chat model. Nevertheless, imposing conversational goals on a dialogue agent for learning target-guided strategies requires keyword-augmented conversation datasets for learning how to steer the conversation towards a designated target subject [451].

Rule constraints While most research that is currently trying to address constrained text generation is focusing on the incorporation of pre-defined utility or lexical constraints, the satisfaction of rule based constraints is equally relevant, particularly when used to define format and syntactic conditions on the output. However, the lack of model expressiveness makes it challenging to incorporate rule based constraints into the loss function at training time. We encourage more effort in this direction likely to open a plethora of new possibilities in how constraints are specified, incorporated and satisfied in models particularly designed for constrained neural text generation.

Evaluation of constrained text generation In general, evaluation of text generative models is an open challenge. The field is missing robust automated evaluation metrics that correlate with human judgements across multiple dimensions of text quality. Evaluation of models for constrained text generation is currently done using the same flawed existing metrics commonly used in unconditional and conditional text generation evaluation, or in an informal way often times in the absence of a rigorous evaluation procedure. Human evaluation remains the gold standard way to assess text quality, however designing evaluation metrics tailored specifically at assessing whether generated texts meet desired constraints altogether with new benchmark datasets for the evaluation of constrained sequence generation are important next steps [236].

Adversarial Attacks Adversarial examples exploit vulnerabilities in text generation models and represent an active research area. Adversarial triggers in the form of input-agnostic sequences of tokens concatenated to any input dataset can trigger a pre-trained language models to produce biased, racist and discriminatory outputs even when these models are carefully fine-tuned and optimized against adversarial triggers [468]. Gradient-based adversarial trigger phrase search techniques are used to generate input prompts to a pre-language model that induce biases in the generated output and allows to study strategies for bias mitigation [418]. Constrained text generation models that are robust to adversarial attacks are needed for the beneficial use of machine learning and artificial intelligence technology in real world applications, as well as to mitigate any potential societal harms and biases associated with the deployment of large pre-trained language models.

While the above directions outline some of the most pressing research challenges associated with constrained text generation, it is nevertheless a non-exhaustive list of all research problems that need increased attention. Other important open challenges include the use of constrained text generation for personalized agents in a wide variety of contexts, such as in dialogue settings [513], and new benchmark datasets that are reflective of real-world constraints for both training/fine-tuning and evaluating constrained text generation models.

2.9 Contributions/novelty of this dissertation

In this work, we have presented the reasons why constrained natural language generation is an important, yet highly challenging and largely unsolved research problem. Our first contribution consists in clarifying the difference between the ambiguous use of unconditional, conditional and constrained terms in the natural language generation literature, and draw clear boundaries between these concepts by exemplifying instances of natural language generation tasks with their associated conditions and constraints. Among different paradigms of text generation, we consider constrained text generation to be particularly challenging (if not the most challenging), yet also extremely useful. We identify general reasons why constrained natural language generation deserves significant more attention in the research community, including the lack of model expressiveness in incorporating constraints into the objective function at training time, difficulty in constrained optimization algorithms, the lack of suitable evaluation metrics for robustly assessing, comparing model outputs and claiming success in constrained natural language generation, as well as the lack of constrained text generation datasets that are representative of a wide range of real-world constraints for training and fine-tuning these models. We then survey a representative body of recent literature on constrained text generation using neural networks, presenting the main approaches and methods used, as well as their limitations. Our work serves as an informative guide for both researchers and practitioners to become familiar with the current methodology and main challenges, as well as an advocate for advancing the state-of-the-art in constrained NLG. We invite future work in solving the outlined challenges for better, useful, safer, robust constrained text generation and evaluation.

CHAPTER 3

Explainable Prediction of Text Complexity: The Missing Preliminaries for Text Simplification

In this chapter, we focus on the challenge of lack of interpretability of black-box procedures when applied to the task of text simplification. Our main contribution is that we show that the ambiguous notion of text simplification can be decomposed into a compact, transparent, and logically dependent pipeline of modular sub-task that increase the transparency and explainability of text simplification systems, while also improving the generalization of state-of-the-art text simplification models in out-of-distribution settings. This work is presented in:

- Gârbacea, Cristina, Mengtian Guo, Samuel Carton, and Qiaozhu Mei. *"Explainable Prediction of Text Complexity: The Missing Preliminaries for Text Simplification."* In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, vol. 1. 2021.

Text simplification reduces the language complexity of professional content for accessibility purposes. End-to-end neural network models have been widely adopted to directly generate the simplified version of input text, usually functioning as a blackbox. We show that text simplification can be decomposed into a compact pipeline of tasks to ensure the transparency and explainability of the process. The first two steps in this pipeline are often neglected: 1) to predict whether a given piece of text needs to be simplified, and 2) if yes, to identify complex parts of the text. The two tasks can be solved separately using either lexical or deep learning methods, or solved jointly. Simply applying explainable complexity prediction as a preliminary step, the out-of-sample text simplification performance of the state-of-the-art, black-box simplification models can be improved by a large margin.

3.1 Introduction

Text simplification aims to reduce the language complexity of highly specialized textual content so that it is accessible for readers who lack adequate literacy skills, such as children, people with low education, people who have reading disorders or dyslexia, and non-native speakers of the language.

Mismatch between language complexity and literacy skills is identified as a critical source of bias and inequality in the consumers of systems built upon processing and analyzing professional text content. Research has found that it requires on average 18 years of education for a reader to properly understand the clinical trial descriptions on ClinicalTrials.gov, and this introduces a potential self-selection bias to those trials [486].

Text simplification has considerable potential to improve the fairness and transparency of text information systems. Indeed, the Simple English Wikipedia (`simple.wikipedia.org`) has been constructed to disseminate Wikipedia articles to kids and English learners. In healthcare, consumer vocabulary are used to replace professional medical terms to better explain medical concepts to the public [1]. In education, natural language processing and simplified text generation technologies are believed to have the potential to improve student outcomes and bring equal opportunities for learners of all levels in teaching, learning and assessment [302].

Ironically, the definition of "text simplification" in literature has never been transparent. The term may refer to reducing the complexity of text at various linguistic levels, ranging all the way through replacing individual words in the text to generating a simplified document completely through a computer agent. In particular, *lexical simplification* [91] is concerned with replacing complex words or phrases with simpler alternatives; *syntactic simplification* [427] alters the syntactic structure of the sentence; *semantic simplification* [197] paraphrases portions of the text into simpler and clearer variants. More recent approaches simplify texts in an end-to-end fashion, employing machine translation models in a monolingual setting regardless of the type of simplifications [516, 153, 460]. Nevertheless, these models are limited on the one hand due to the absence of large-scale parallel (complex $\rightarrow$ simple) monolingual training data, and on the other hand due to the lack of interpretability of their black-box procedures [4].

Given the ambiguity in problem definition, there also lacks consensus on how to measure the goodness of text simplification systems, and automatic evaluation measures are perceived ineffective and sometimes detrimental to the specific procedure, in particular when they favor shorter but not necessarily simpler sentences [324]. While end-to-end simplification models demonstrate superior performance on benchmark datasets, their success is often compromised in out-of-sample, real-world scenarios [79].

Our work is motivated by the aspiration that increasing the transparency and explainability of a machine learning procedure may help its generalization into unseen scenarios [98]. We show that the general problem of text simplification can be formally decomposed into a compact and transparent pipeline of modular tasks. We present a systematic analysis of the first two steps in this pipeline, which are commonly overlooked: 1) *to predict whether a given piece of text needs to be simplified at all*, and 2) *to identify which part of the text needs to be simplified.* The second task can also be interpreted as an explanation of the first task: why a piece of text is considered complex. These two tasks can be solved separately, using either lexical or deep learning methods, or they can be solved jointly through an end-to-end, explainable predictor. Based on the formal definitions, we propose general evaluation metrics for both tasks and empirically compare a diverse portfolio of methods using multiple datasets from different domains, including news, Wikipedia, and scientific papers. We demonstrate that by simply applying explainable complexity prediction as a preliminary step, the out-of-sample text simplification performance of the state-of-the-art, black-box models can be improved by a large margin.

Our work presents a promising direction towards a transparent and explainable solution to text simplification in various domains.

3.2 Related Work

3.2.1 Text Simplification

3.2.1.1 Identifying complex words

Text simplification at word level has been done through 1) **lexicon based** approaches, which match words to lexicons of complex/simple words [88, 103], 2) **threshold based** approaches, which apply a threshold over word lengths or certain statistics [245], 3) **human driven** approaches, which solicit the user's input on which words need simplification [381], and 4) **classification** methods, which train machine learning models to distinguish complex words from simple words [413]. Complex word identification is also the main topic of SemEval 2016 Task 11 [340], aiming to determine whether a non-native English speaker can understand the meaning of a word in a given sentence. Significant differences exist between simple and complex words, and the latter on average are shorter, less ambiguous, less frequent, and more technical in nature. Interestingly, the frequency of a word is identified as a reliable indicator of its simplicity [245].

While the above techniques have been widely employed for complex word identification, the results reported in the literature are rather controversial and it is not clear to what extent one technique outperforms the other in the absence of standardized high quality parallel corpora for text simplification [339]. Pre-constructed lexicons are often limited and do not generalize to different domains. It is intriguing that classification methods reported in the literature are not any better than a "simplify-all" baseline [414].

3.2.1.2 Readability assessment

Traditionally, measuring the level of reading difficulty is done through lexicon and rule-based metrics such as the age of acquisition lexicon (AoA) [227] and the Flesch-Kincaid Grade Level [214]. A machine learning based approach in [401] extracts lexical, syntactic, and discourse features and train logistic regression classifiers to predict the relative complexity of a single sentence in a pairwise setting. The most predictive features are simple representations based on AoA norms. The perceived difficulty of a sentence is highly influenced by properties of the surrounding passage. Similar methods are used for fine-grained classification of text readability [3] and complexity [436].

3.2.1.3 Computer-assisted paraphrasing

Simplification rules are learnt by finding words from a complex sentence that correspond to different words in a simple sentence [4]. Identifying simplification operations such as copies, deletions, and substitutions for words from parallel complex vs. simple corpora helps understand how human experts simplify text [4]. Machine translation has been employed to learn phrase-level alignments for sentence simplification [490]. Lexical and phrasal paraphrase rules are extracted in [352]. These methods are often evaluated by comparing their output to gold-standard, human-generated simplifications, using standard metrics (e.g., token-level precision, recall, F1), machine translation metrics (e.g., BLEU [345]), text simplification metrics (e.g. SARI [497] which rewards copying words from the original sentence), and readability metrics (among which Flesch-Kincaid Grade Level [214] and Flesch Reading Ease [214] are most commonly used). It is desirable that the output of the computational models is ultimately validated by human judges [414].

3.2.1.4 End-to-end simplification

Neural encoder-decoder models are used to learn simplification rewrites from monolingual corpora of complex and simple sentences [394, 460, 516, 153]. On one hand, these models often obtain superior performance on particular evaluation metrics, as the neural network directly optimizes these metrics in training. On the other hand, it is hard to interpret what exactly are learned in the hidden layers, and without this transparency it is difficult to adapt these models to new data, constraints, or domains. For example, these end-to-end simplification models tend not to distinguish whether the input text should or should not be simplified at all, making the whole process less transparent. When the input is already simple, the models tend to oversimplify it and deviate from its original meaning (see Section 3.5.3).

3.2.2 Explanatory Machine Learning

Various approaches are proposed in the literature to address the explainability and interpretability of machine learning agents. The task of providing explanations for black-box models has been tackled either at a local level by explaining individual predictions of a classifier [383], or at a global level by providing explanations for the model behavior as a whole [247]. More recently, differential explanations are proposed to describe how the logic of a model varies across different subspaces of interest [232]. Layer-wise relevance propagation [7] is used to trace backwards text classification decisions to individual words, which are assigned scores to reflect their separate contribution to the overall prediction.

LIME [383] is a model-agnostic explanation technique which can approximate any machine learning model locally with another sparse linear interpretable model. SHAP [291] evaluates Shapley values as the average marginal contribution of a feature value across all possible coalitions by considering all possible combinations of inputs and all possible predictions for an instance. Explainable classification can also be solved simultaneously through a neural network, using hard attentions to select individual words into the "rationale" behind a classification decision [242]. Extractive adversarial networks employs a three-player adversarial game which addresses high recall of the rationale [50]. The model consists of a generator which extracts an attention mask for each token in the input text, a predictor that cooperates with the generator and makes prediction from the rationale (words attended to), and an adversarial predictor that makes predictions from the remaining words in the inverse rationale. The minimax game between the two predictors and the generator is designed to ensure all predictive signals are included into the rationale.

No prior work has addressed the explainability of text complexity prediction. We fill in this gap.

3.3 An Explainable Pipeline for Text Simplification

We propose a unified view of text simplification which is decomposed into several carefully designed sub-problems. These sub-problems generalize over many approaches, and they are logically dependent on and integratable with one another so that they can be organized into a compact pipeline.

The first conceptual block in the pipeline (Figure 3.1) is concerned with explainable prediction of the complexity of text. It consists of two sub-tasks: 1) *prediction*: classifying a given piece of text into two categories, needing simplification or not; and 2) *explanation*: highlighting the part of the text that needs to be simplified. The second conceptual block is concerned with simplification generation, the goal of which is to generate a new, simplified version of the text that needs

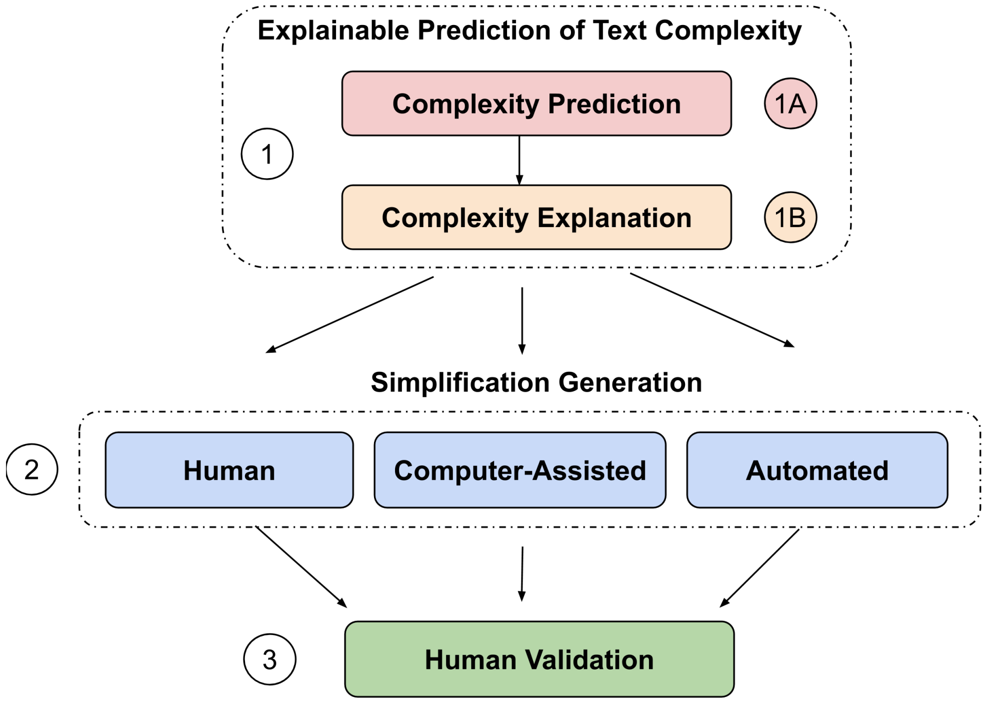


Figure 3.1: A text simplification pipeline. Explainable prediction of text complexity is the preliminary of any human-based, computer assisted, or automated system.

to be simplified. This step could be achieved through completely manual effort, or a computer-assisted approach (e.g., by suggesting alternative words and expressions), or a completely automated method (e.g., by self-translating into a simplified version). The second building block is piped into a step of human judgment, where the generated simplification is tested, approved, and evaluated by human practitioners.

One could argue that for an automated simplification generation system the first block (complexity prediction) is not necessary. We show that it is not the case. Indeed, it is unlikely that every piece of text needs to be simplified in reality, and instead the system should first decide whether a sentence needs to be simplified or not. Unfortunately such a step is often neglected by existing end-to-end simplifiers, thus their performance is often biased towards the complex sentences that are selected into their training datasets at the first place and doesn't generalize well to simple inputs. Empirically, when these models are applied to out-of-sample text which shouldn't be simplified at all, they tend to oversimplify the input and result in a deviation from its original meaning (see Section 3.5.3).

One could also argue that an explanation component (1B) is not mandatory in certain text simplification practices, in particular in an end-to-end neural generative model that does not explicitly identify the complex parts of the input sentence. In reality, however, it is often necessary to highlight the differences between the original sentence and the simplified sentence (which is essentially a variation of 1B) to facilitate the validation and evaluation of these black-boxes. More generally, the explainability/interpretability of a machine learning model has been widely believed to be an indispensable factor to its fidelity and fairness when applied to the real world [232]. Since the major motivation of text simplification is to improve the fairness and transparency of text information systems, it is critical to explain the rationale behind the simplification decisions, even if they are made through a black-box model.

Without loss of generality, we can formally define the sub-tasks 1A, 1B, and 2- in the pipeline:

Definition 3.3.1. (Complexity Prediction). Let text $d \in D$ be a sequence of tokens $w_1 w_2 ... w_n$. The task of complexity prediction is to find a function $f : D \to \{0, 1\}$ such that $f(d) = 1$ if d needs to be simplified, and $f(d) = 0$ otherwise.

Definition 3.3.2. (Complexity Explanation). Let d be a sequence of tokens $w_1 w_2 ... w_n$ and $f(d) = 1$. The task of complexity explanation/highlighting is to find a function $h : D \to \{0, 1\}^n$ s.t. $h(d) = c_1 c_2 ... c_n$, where $c_i = 1$ means w_i will be highlighted as a complex portion of d and $c_i = 0$ otherwise. $d|h(d)$ denotes the highlighted part of d and $d|\neg h(d)$ is the unhighlighted part of d.

Definition 3.3.3. (Simplification Generation). Let d be a sequence of tokens $w_1 w_2 ... w_n$ and $f(d) = 1$. The task of simplification generation is to find a function $g : D \to D'$ s.t. $g(d, f(d), h(d)) = d'$, where $d' = w'_1 w'_2 ... w'_m$ and $f(d') = 0$, subject to the constraint that d' preserves the meaning of d.

In this paper, we focus on an empirical analysis of the first two sub-tasks of explainable prediction of text complexity (1A and 1B), which are the preliminaries of any reasonable text simplification practice. We leave aside the detailed analysis of simplification generation (2-) for now, as there are many viable designs of $g(\cdot)$ in practice, spanning the spectrum between completely manual and completely automated. Since this step is not the focus of this paper, we intend to leave the definition of simplification generation highly general.

Note that the definitions of complexity prediction and complexity explanation can be naturally extended to a continuous output, where $f(\cdot)$ predicts the complexity level of d and $h(\cdot)$ predicts the complexity weight of w_i. The continuous output would align the problem more closely to readability measures [214]. In this paper, we stick to the binary output because a binary action (to simplify or not) is almost always necessary in reality even if a numerical score is available.

Note that the definition of complexity explanation is general enough for existing approaches. In lexical simplification where certain words in a complex vocabulary V are identified to explain the complexity of a sentence, it is equivalent to highlighting every appearance of these words in d, or $\forall w_i \in V, c_i = 1$. In automated simplification where there is a self-translation function $g(d) = d'$, $h(d)$ can be simply instantiated as a function that returns a sequence alignment of d and d'. Such reformulation helps us define unified evaluation metrics for complexity explanation (Section 3.4).

It is also important to note that the dependency between the components, especially complexity prediction and explanation, does not restrict them to be done in isolation. These sub-tasks can be done either separately, or jointly with an end-to-end approach as long as the outputs of f, h, g are all obtained (so that transparency and explainability are preserved). In Section 3.4, we include both separate models and end-to-end models for explanatory complexity predication in one shot.

3.4 Empirical Analysis of Complexity Prediction and Explanation

With the pipeline formulation, we are able to compare a wide range of methods and metrics for the sub-tasks of text simplification. We aim to understand how difficult they are in real-world settings and which method performs the best for which task.

3.4.1 Complexity Prediction

3.4.1.1 Candidate Models

We examine a wide portfolio of deep and shallow binary classifiers to distinguish complex sentences from simple ones. Among the shallow models we use Naive Bayes (NB), Logistic Regression (LR), Support Vector Machines (SVM) and Random Forests (RF) classifiers trained with unigrams, bigrams and trigrams as features. We also train the classifiers using the lexical and syntactic features proposed in [401] combined with the n-gram features (denoted as "enriched features"). We include neural network models such as word and char-level Long Short-Term Memory Network (LSTM) and Convolutional Neural Networks (CNN). We also employ a set of state-of-the-art pre-trained neural language models, fine-tuned for complexity prediction; we introduce them below.

ULMFiT [174] a language model on a large general corpus such as WikiText-103 and then fine-tunes it on the target task using slanted triangular rates, and gradual unfreezing. We use the publicly available implementation[1] of the model with two fine-tuning epochs for each dataset and the model quickly adapts to a new task.

BERT [90] trains deep bidirectional language representations and has greatly advanced the state-of-the-art for many natural language processing tasks. The model is pre-trained on the English Wikipedia as well as the Google Book Corpus. Due to computational constraints, we use the 12 layer BERT base pre-trained model and fine-tune it on our three datasets. We select the best hyperparameters based on each validation set.

[1] https://docs.fast.ai/tutorial.text.html, retrieved on 5/31/2021.

XLNeT [501] overcomes the limitations of BERT (mainly the use of masks) with a permutation-based objective which considers bidirectional contextual information from all positions without data corruption. We use the 12 layer XLNeT base pre-trained model on the English Wikipedia, the Books corpus (similar to BERT), Giga5, ClueWeb 2012-B, and Common Crawl.

3.4.1.2 Evaluation Metric

We evaluate the performance of complexity prediction models using *classification accuracy* on balanced training, validation, and testing datasets.

3.4.2 Complexity Explanation

3.4.2.1 Candidate Models

We use *LIME* in combination with LR and LSTM classifiers, *SHAP* on top of LR, and the *extractive adversarial networks* which jointly conducts complexity prediction and explanation. We feed each test complex sentence as input to these explanatory models and compare their performance at identifying tokens (words and punctuation) that need to be removed or replaced from the input sentence.

We compare these explanatory models with three baseline methods: 1) *Random highlighting*: randomly draw the size and the positions of tokens to highlight; 2) *Lexicon based highlighting*: highlight words that appear in the Age-of-Acquisition (AoA) lexicon [227], which contains ratings for 30,121 English content words (nouns, verbs, and adjectives) indicating the age at which a word is acquired; and 3) *Feature highlighting*: highlight the most important features of the best performing LR models for complexity prediction.

3.4.2.2 Evaluation Metrics

Evaluation of explanatory machine learning is an open problem. In the context of complexity explanation, when the ground truth of highlighted tokens ($y_c(d) = c_1c_2...c_n$, $c_i \in \{0, 1\}$) in each complex sentence d is available, we can compare the output of complexity explanation $h(d)$ with $y_c(d)$. Such per-token annotations are usually not available in scale. To overcome this, given a complex sentence d and its simplified version d', we assume that all tokens w_i in d which are absent in d' are candidate words for deletion or substitution during the text simplification process and should therefore be highlighted in complexity explanation (i.e., $c_i = 1$).

In particular, we use the following evaluation metrics for complexity explanation: 1) *Tokenwise Precision (P)*, which measures the proportion of highlighted tokens in d that are truly removed in d'; 2) *Tokenwise Recall (R)*, which measures the proportion of tokens removed in d' that are actually highlighted in d; 3) *Tokenwise F1*, the harmonic mean of P and R; 4) word-level *Edit distance (ED)* [248]: between the **unhighlighted** part of d and the simplified document d'. Intuitively, a more successful complexity explanation would highlight most of the tokens that need to be simplified, thus the remaining parts in the complex sentences will be closer to the simplified version, achieving a lower edit distance (we also explore ED with a higher penalty cost for the substitution operation, namely values of 1, 1.5 and 2); and 5) *Translation Edit Rate (TER)* [430], which measures the minimum number of edits needed to change a hypothesis (the unhighlighted part of d) so that it exactly matches the closest references (the simplified document d'). Note these metrics are all proxies of the real editing process from d to d'. When token-level edit history is available (e.g., through track changes), it is better to compare the highlighted evaluation with these true changes made. We compute all the metrics at sentence level and macro-average them.

3.4.3 Experiment Setup

3.4.3.1 Datasets

We use three different datasets (Table 3.1) which cover different domains and application scenarios of text simplification. Our first dataset is *Newsela* [496], a corpus of news articles simplified by professional news editors. In our experiments we use the parallel Newsela corpus with the training, validation, and test splits made available in [516]. Second, we use the *WikiLarge* corpus introduced in [516]. The training subset of WikiLarge is created by assembling datasets of parallel aligned Wikipedia - Simple Wikipedia sentence pairs available in the literature [202]. While this training set is obtained through automatic alignment procedures which can be noisy, the validation and test subsets of WikiLarge contain complex sentences with simplifications provided by Amazon Mechanical Turk workers [497]; we increase the size of validation and test on top of the splits made available in [516]. Third, we use the dataset released by the *Biendata* competition[2], which asks participants to match research papers from various scientific disciplines with press releases that describe them. Arguably, rewriting scientific papers into press releases has mixed objectives that are not simply text simplification. We include this task to test the generalizability of our explainable pipeline (over various definitions of simplification). We use alignments at title level. On average, a complex sentence in Newsela, WikiLarge, Biendata contains 23.07, 25.14, 13.43 tokens, and the corresponding simplified version is shorter, with 12.75, 18.56, 10.10 tokens.

[2] https://www.biendata.com/competition/hackathon, retrieved on 5/31/2021.

Table 3.1: Aligned complex-simple sentence pairs.

Dataset	**Training**	**Validation**	**Test**
Newsela	94,208 pairs	1,129 pairs	1,077 pairs
WikiLarge	208,384 pairs	29,760 pairs	59,546 pairs
Biendata	29,700 pairs	4,242 pairs	8,486 pairs

3.4.3.2 Ground Truth Labels

The original datasets contain aligned complex-simple sentence pairs instead of classification labels for complexity prediction. We infer ground-truth complexity labels for each sentence such that: *label 1* is assigned to every sentence for which there is an aligned simpler version not identical to itself (the sentence is complex and needs to be simplified); *label 0* is assigned to all simple counterparts of complex sentences, as well as to those sentences that have corresponding "simple" versions identical to themselves (i.e., these sentences do not need to be simplified). For complex sentences that have label 1, we further identify which tokens are not present in corresponding simple versions.

3.4.3.3 Model Training

For all shallow and deep classifiers we find the best hyperparameters using random search on validation, with early stopping. We use grid search on validation to fine-tune hyperparameters of the pre-trained models, such as maximum sequence length, batch size, learning rate, and number of epochs. For ULMFit on Newsela, we set batch size to 128 and learning rate to 1e-3. For BERT on WikiLarge, batch size is 32, learning rate is 2e-5, and maximum sequence length is 128. For XLNeT on Biendata, batch size is 32, learning rate is 2e-5, and maximum sequence length is 32.

We use grid search on validation to fine-tune the complexity explanation models, including the extractive adversarial network. For LR and LIME we determine the maximum number of words to highlight based on TER score on validation (please see Table 3.2); for SHAP we highlight all features with positive assigned weights, all based on TER.

Table 3.2: Maximum numbers of most important LR features and features highlighted by LIME.

Model	**Newsela**	**WikiLarge**	**Biendata**
LR	200 features	20,000 features	200 features
LIME & LR	10 features	50 features	10 features
LIME & LSTM	60 features	20 features	40 features

For extractive adversarial networks batch size is set to 256, learning rate is 1e-4, and adversarial weight loss equals 1; in addition, sparsity weight is 1 for Newsela and Biendata, and 0.6 for WikiLarge; lastly, coherence weight is 0.05 for Newsela, 0.012 for WikiLarge, and 0.0001 for Biendata.

3.5 Results

3.5.1 Complexity Prediction

In Table 3.3, we evaluate how well the representative shallow, deep, and pre-trained classification models can determine whether a sentence needs to be simplified at all. We test for statistical significance of the best classification results compared to all other models using a two-tailed z-test.

Table 3.3: Accuracy of representative shallow*, deep, and pre-trained models for complexity prediction. **BOLD**: best performing models. * Shallow models perform similarly and some are omitted for space; Difference between the best performing model and other models is statistically significant: $p < 0.05$ (*), $p < 0.01$ (**), except for †: difference between this model and the best performing model is not statistically significant.

Classifier	Newsela	WikiLarge	Biendata
NB n-grams	73.10 %	62.70 %	84.30 %
NB enriched features	73.10 %	63.10 %	86.00 %
LR n-grams	75.30 %	71.90 %	89.60 %
LR enriched features	76.30 %	72.60 %	91.70 %
SVM n-grams	75.20 %	71.90 %	89.50 %
SVM enriched features	77.39 %	70.16 %	88.60 %
RF n-grams	71.50 %	71.50 %	84.60 %
RF enriched features	74.40 %	73.40 %	87.00 %
LSTM (word-level)	73.31 %	71.62 %	89.87 %
CNN (word-level)	70.71 %	69.27 %	89.05 %
CNN (char-level)	$78.83\%^{\dagger}$	74.88 %	88.00 %
CNN (word & char-level)	75.90 %	74.00 %	92.30 %
Extractive Adversarial Networks	72.76 %	71.50 %	88.64 %
ULMFiT	**80.83%****	74.80 %	94.17 %
BERT	77.15 %	**81.45%****	94.43 %
XLNeT	$78.83\%^{\dagger}$	73.49 %	**95.48%****

In general, the best performing models can achieve around 80% accuracy on two datasets (Newsela and WikiLarge) and a very high performance on the Biendata ($> 95\%$). This difference presents the difficulty of complexity prediction in different domains, and distinguishing highly specialized scientific content from public facing press releases is relatively easy (Biendata).

Deep classification models in general outperform shallow ones, however with carefully designed handcrafted features and proper hyperparameter optimization shallow models tend to approach to the results of the deep classifiers. Overall models pre-trained on large datasets and fine-tuned for text simplification yield superior classification performance. For Newsela the best performing classification model is ULMFiT (accuracy = 80.83%, recall = 76.87%), which significantly ($p < 0.01$) surpasses all other classifiers except for XLNeT and CNN (char-level). On WikiLarge, BERT presents the highest accuracy ($81.45\%, p < 0.01$), and recall = 83.30%. On Biendata, XLNeT yields the highest accuracy ($95.48\%, p < 0.01$) with recall = 94.93%, although the numerical difference to other pre-trained language models is small. This is consistent with recent findings in other natural language processing tasks [72].

3.5.2 Complexity Explanation

We evaluate how well complexity classification can be explained, or how accurately the complex parts of a sentence can be highlighted.

Results (Table 3.4) show that highlighting words in the AoA lexicon or LR features are rather strong baselines, indicating that most complexity of a sentence still comes from word usage. Highlighting more LR features leads to a slight drop in precision and a better recall. Although LSTM and LR perform comparably on complexity classification, using LIME to explain LSTM presents better recall, F1, and TER (at similar precision) compared to using LIME to explain LR. The LIME & LSTM combination is reasonably strong on all datasets, as is SHAP & LR. TER is a reliable indicator of the difficulty of the remainder (unhighlighted part) of the complex sentence. ED with a substitution penalty of 1.5 efficiently captures the variations among the explanations. On Newsela and Biendata, the extractive adversarial networks yield solid performances (especially TER and ED 1.5), indicating that jointly making predictions and generating explanations reinforces each other. Table 3.5 provides examples of highlighted complex sentences by each explanatory model.

3.5.3 Benefit of Complexity Prediction

One may question whether explainable prediction of text complexity is still a necessary preliminary step in the pipeline if a strong, end-to-end simplification generator is used. We show that it is. We consider the scenario where a pre-trained, end-to-end text simplification model is blindly applied to texts regardless of their complexity level, compared to only simplifying those considered complex by the best performing complexity predictor in Table 3.3. Such a comparison demonstrates whether adding complexity prediction as a preliminary step is beneficial to a text simplification process when a state-of-the-art, end-to-end simplifier is already in place. From literature we select the current best text simplification models on WikiLarge and Newsela which have publicly released

Table 3.4: Results for complexity explanation. P, R and F1 - the higher the better; TER and ED 1.5 - the lower the better. **BOLD** & Underlined: best & second best.

Dataset	Explanation Model	P	R	F1	TER	ED 1.5
Newsela	Random	0.515	0.487	0.439	0.985	13.825
	AoA lexicon	**0.556**	0.550	0.520	0.867	12.899
	LR Features	0.522	0.250	0.321	0.871	12.103
	LIME & LR	0.535	0.285	0.343	0.924	12.459
	LIME & LSTM	0.543	**0.818**	**0.621**	0.852	11.991
	SHAP & LR	0.553	0.604	0.546	0.848	12.656
	Extractive Networks	0.530	0.567	0.518	**0.781**	**11.406**
WikiLarge	Random	0.412	0.439	0.341	1.546	17.028
	AoA lexicon	0.427	0.409	0.357	1.516	16.731
	LR Features	0.442	0.525	0.413	0.993	17.933
	LIME & LR	0.461	0.509	0.415	**0.988**	18.162
	LIME & LSTM	**0.880**	0.470	0.595	1.961	25.051
	SHAP & LR	0.842	**0.531**	**0.633**	1.693	22.811
	Extractive Networks	0.452	0.429	0.359	1.434	**16.407**
Biendata	Random	0.743	0.436	0.504	1.065	12.921
	AoA lexicon	0.763	0.383	0.475	1.064	13.247
	LR Features	0.796	0.257	0.374	0.979	10.851
	LIME & LR	**0.837**	0.466	0.577	0.982	**10.397**
	LIME & LSTM	0.828	0.657	0.713	**0.952**	16.568
	SHAP & LR	0.825	0.561	0.647	0.979	11.908
	Extractive Networks	0.784	**0.773**	**0.758**	0.972	10.678

Table 3.5: Explanations of complexity predictions (in red). Extractive network obtains a higher recall.

Explanatory Model	Complexity Explanation
LIME & LR	Their fatigue changes their voices , but they 're still on the freedom highway .
LIME & LSTM	Their fatigue changes their voices , but they 're still on the freedom highway .
SHAP & LR	Their fatigue changes their voices , but they 're still on the freedom highway .
Extractive Networks	Their fatigue changes their voices , but they 're still on the freedom highway .
Simple sentence	Still , they are fighting for their rights .
LIME & LR	Digitizing physically preserves these fragile papers and allows people to see them , he said .
LIME & LSTM	Digitizing physically preserves these fragile papers and allows people to see them , he said .
SHAP & LR	Digitizing physically preserves these fragile papers and allows people to see them , he said .
Extractive Networks	Digitizing physically preserves these fragile papers and allows people to see them , he said .
Simple sentence	The papers are old and fragile , he said .

pre-trained models:

- ACCESS [298], a controllable sequence-to-sequence simplification model that reported the highest performance (41.87 SARI) on WikiLarge.

- Dynamic Multi-Level Multi-Task Learning for Sentence Simplification (DMLMTL) [153], which reported the highest performance (33.22 SARI) on Newsela.

We apply the author-released, pre-trained ACCESS and DMLMTL on all sentences from the validation and test sets of all three datasets. We do not use the training examples as the pre-trained models may have already seen them. Presumably, a smart model should **not** further simplify an input sentence if it is already simple enough. However, to our surprise, a majority of the *out-of-sample simple* sentences are still changed by both models (above 90% by DMLMTL and above 70% by ACCESS, please see Table 4.10).

Table 3.6: Percentage of out-of-sample simple sentences changed by pre-trained, end-to-end simplification models. Ideal value is 0%.

Dataset	**Pre-trained Model**	**Validation**	**Testing**
Newsela	ACCESS	72.73 %	75.50 %
	DMLMTL	90.48 %	91.69 %
WikiLarge	ACCESS	70.83 %	71.12 %
	DMLMTL	95.20 %	95.61 %
Biendata	ACCESS	94.25 %	93.66 %
	DMLMTL	98.88 %	98.73 %

We further quantify the difference with vs. without complexity prediction as a preliminary step. Intuitively, without complexity prediction, an already simple sentence is likely to be overly simplified and result in a loss in text simplification metrics. In contrast, an imperfect complexity predictor may mistaken a complex sentence as simple, which misses the opportunity of simplification and results in a loss as well. The empirical question is which loss is higher. From Table 3.7, we see that after directly adding a complexity prediction step before either of the state-of-the-art simplification models, there is a considerable drop of errors in three text simplification metrics: Edit Distance (ED), TER, and Fréchet Embedding Distance (FED) that measures the difference of a simplified text and the ground-truth in a semantic space [85]. For ED alone, the improvements are between 30% to 50%. This result is very encouraging: considering that the complexity predictors are only 80% accurate and the complexity predictor and the simplification models don't depend on each other, there is considerable room to optimize this gain. Indeed, the benefit is higher on Biendata where the complexity predictor is more accurate.

Qualitatively, one could frequently observe syntactic, semantic, and logical mistakes in the model-simplified version of *simple* sentences. We give a few examples below.

- In Ethiopia, HIV disclosure is low → In Ethiopia , HIV is low (ACCESS)

Table 3.7: Out-of-sample performance of simplification models. ED, TER, FED metrics: the lower the better. Adding complexity prediction as preliminary step reduces simplification error by a wide margin.

Dataset	Sentence Pairs	Metric	ACCESS	DMLMTL
Newsela	No complexity prediction (simplify everything)	ED	4.044	12.212
		TER	0.175	1.611
		FED	0.016	0.170
	With complexity prediction (predicted simple: no change)	ED	**2.631** (-35%)	**8.677** (-29%)
		TER	0.089 (-49%)	1.149 (-29%)
		FED	0.006 (-63%)	0.066 (-61%)
WikiLarge	No Complexity Prediction (simplify everything)	ED	5.857	16.920
		TER	0.208	2.328
		FED	0.004	0.143
	With Complexity Prediction (predicted simple: no change)	ED	**4.021** (-31%)	**10.566** (-38%)
		TER	0.132 (-37%)	1.452 (-38%)
		FED	0.002 (-50%)	0.049 (-66%)
Biendata	No Complexity Prediction (simplify everything)	ED	3.796	9.030
		TER	0.254	1.348
		FED	0.033	0.131
	With Complexity Prediction (predicted simple: no change)	ED	**1.887** (-50%)	**5.249** (-42%)
		TER	0.114 (-55%)	0.819 (-39%)
		FED	0.009 (-73%)	0.051 (-61%)

- Mustafa Shahbaz , 26 , was shopping for books about science . → Mustafa Shahbaz , 26 years old , was a group of books about science . (ACCESS)
- New biomarkers for the diagnosis of Alzheimer's → New biomarkers are diagnosed with Alzheimer (ACCESS)
- Healthy diet linked to lower risk of chronic lung disease → Healthy diet linked to lung disease (DMLMTL)
- Dramatic changes needed in farming practices to keep pace with climate change → changes needed to cause climate change (DMLMTL)
- Social workers can help patients recover from mild traumatic brain injuries → Social workers can cause better problems . (DMLMTL)

All these qualitative and quantitative results suggest that the state-of-the-art black-box models tend to oversimplify and distort the meanings of out-of-sample input that is already simple. Evidently, the lack of transparency and explainability has limited the application of these end-to-end black-box models in reality, especially to out-of-sample data, context, and domains. The pitfall can be avoided with the proposed pipeline and simply with explainable complexity prediction as a preliminary step. Even though this explainable preliminary does not necessarily reflect how a black-box simplification model "thinks", adding it to the model is able to yield better out-of-sample performance.

3.6 Conclusions

We formally decompose the ambiguous notion of text simplification into a compact, transparent, and logically dependent pipeline of sub-tasks, where explainable prediction of text complexity is identified as the preliminary step. We conduct a systematic analysis of its two sub-tasks, namely complexity prediction and complexity explanation, and show that they can be either solved separately or jointly through an extractive adversarial network. While pre-trained neural language models achieve significantly better performance on complexity prediction, an extractive adversarial network that solves the two tasks jointly presents promising advantage in complexity explanation. Using complexity prediction as a preliminary step reduces the error of the state-of-the-art text simplification models by a large margin. Future work should integrate rationale extractor into the pre-trained neural language models and extend it for simplification generation.

3.6.1 Future Outlook/Emerging Trends

The work presented in this chapter was conducted before language models became extremely large and widely popular. Given the remarkable performance improvements achieved when scaling up Transformer [463] based architectures, new interpretable approaches emerge to address the problem of explainable prediction of text complexity. In what follows we present the novel chain-of-thought [477] prompting approach for eliciting reasoning in large language models. We envision the use of this approach for generating natural language rationales that support text complexity decisions. More specifically, LLMs can be used for both making complexity predictions, i.e. identifying whether a piece of text needs simplification or not, and for explaining the rationale behind their prediction, i.e. identifying the complex words and phrases in the input and providing a step-by-step explanation of why these specific parts of the input are considered complex and need simplification; please see Figure 3.2 and Figure 3.3. While the explainability of black-box neural network models remains an important open research problem, such prompting techniques can offer a glimpse into the inner workings of the model by suggesting how it may have arrived at a particular answer. In addition, they also allow human users to better understand the knowledge captured by a large language model and debug where mistakes may occur in the reasoning chain.

Elicitive Prompting/Chain-of-thought More recent developments aimed at understanding reasoning processes in large language models propose chain-of-thought (CoT) [477] prompting techniques for providing an interpretable window into the model behaviour and eliciting explicit (instead of implicit) reasoning by letting the model "think things through". A chain-of-thought represents a series of intermediate reasoning steps expressed in natural language that illustrate step by step how the model reached its final output. CoT capabilities emerge in sufficiently large language models and lead to improved performance on challenging tasks where simply scaling up model size is not sufficient for achieving high performance, for example long-term/multi-step logic, arithmetic, commonsense, symbolic or compositional reasoning tasks [370], [330]. Such complex multi-step reasoning problems can be decomposed into sequential intermediate steps that are solved individually (by means of generating coherent natural language rationales) given a few examples of chain-of-thought sequences provided as part of the prompt. While this technique allows to elicit multi-step reasoning behaviour in large pre-trained language models (often times with no prompt engineering, by using a simple prompt template such as "Let's think step-by-step" [221]) and improves performance by a large margin for many reasoning tasks, it does not answer the important question of whether the neural-network based model is actually "reasoning". In addition, there is no guarantee that the reasoning paths generated are correct. Greedy search or beam search are frequently employed to decode the CoT reasoning chain [493]. Leaving from the observation that there may be multiple reasoning paths that lead to a correct answer, self-consistency [476] demonstrates that introducing diversity in the reasoning process can be beneficial. The method replaces greedy decoding in CoT prompting with a "sample-and-marginalize" decoding procedure which first samples a diverse set of reasoning paths from the language model decoder, then identifies the optimal path with the most consistent answer by marginalizing out reasoning paths (taking the majority vote over the produced answers). The intuition is that if there are multiple paths leading to the same answer, the greater the confidence the final answer is correct. Discriminative models trained to differentiate correct reasoning steps from invalid ones are used to guide the decoder towards valid inferences [207]. Other approaches for ensuring the correctness of the generated rationales train verifiers to evaluate the feasability of proposed solutions [71] (i.e. sample a fixed number of candidate solutions and select the solution ranked highest by the verifier), rerankers given human annotations to find the highest quality response [458], or finetune the model on annotated data for correcting reasoning errors [112], [444]. Nevertheless, creating high quality rationale-augmented sets for training or fine-tuning verifiers is expensive and time-consuming, and such approaches can easily overfit to ground-truth solutions and ignore alternative reasoning paths [207]. For more efficient verification, examining few local statements instead of the entire reasoning chain is proposed [268]. Human-in-the-loop systems improve reasoning performance by manual correction of sub-logic rationales [41].

Despite improvements in performance reported using CoT, it is not clear how the pre-trained model arrives at the correct answer, i.e. whether it relies on the chains-of-thought generated or rather on shallow simple heuristics [393]; recent work suggests that models mostly make predictions based on shallow rote learning instead of having a deep holistic task understanding [100]. It is also unclear how the series of reasoning steps provided in the demonstrations contribute to its performance. Interestingly, even when prompted with invalid demonstrations that contain incorrect reasoning steps, large language models can still generate coherent lines of reasoning at inference time [469]. In general, there is little understanding behind what makes CoT prompting effective, however relevance of demonstrations to the query and the correct ordering of the reasoning steps are found to be important factors. Moreover, while large language models are generally able to perform reasoning, how they acquire this ability and whether portions of the pre-training data teach the model to reason remain important open questions. It is also unclear how much of their performance is due to memorization of huge corpora as opposed to actual reasoning abilities [362]. Empirical analysis on compositional multi-hop reasoning problems finds that models can memorize single-step operations and achieve near-perfect performance on in-domain instances with low compositional complexity, however they fail dramatically on instances outside of the training distribution and are inherently limited in solving compositionally complex tasks [100]. An essential aspect of improving reasoning abilities for LLMs is detecting and mitigating hallucinations, given that in compositional reasoning tasks the overall solution depends on correctly composing the answers to sub-problems and a single logical error in a multi-step reasoning problem can derail the entire chain for arriving at the correct final answer. This is particularly true for more difficult reasoning tasks, where the complexity and length of reasoning chains is making them susceptible to errors and imperfections that accumulate across intermediate steps [61]. In general, refining intermediate representations in a meaningful way that would serve to improve final performance is an open challenge. Process supervision [263] is used for training reward models that provide feedback for each intermediate reasoning step and is found superior to outcome supervision models that only reward the final result. However, process supervision is costly as it relies on human annotators to label the correctness of each intermediate step; the cost can be reduced by employing an active learning strategy to solicit human feedback for the most promising completions only. An alternative approach is proposed by Self-ask [362], where the model itself is asking and answering follow-up questions before arriving at the final answer. Interaction-based frameworks where a generator is producing intermediate reasoning steps and a critic provides fine-grained feedback to the generator serve to improve the reasoning process through iterative feedback and refinement [350], [295]. To better explore the vast reasoning space and assess alternative reasoning paths, large language models are augmented with a world model which allows them to perform deliberate planning [159].

In general, the ability of large language models to answer compositional multi-hop questions does not necessarily improve with scale, even when models demonstrate knowledge of individual constituent facts. This is in contrast to their single-hop question answering performance which does improve as more powerful models memorize and recall more factual knowledge. Interestingly, the model confidence at recalling individual facts (as measured by perplexity) is indicative of its ability to compose them at a much higher rate [362]. This is in line with findings on self-evaluation [194] of large language models that measure the ability of models to propose answers to questions and then evaluate the probability that those answers are correct, suggesting that larger models are well-calibrated for in-distribution tasks as long as questions follow a specific format. Nevertheless, most CoT evaluations focus only on the correctness of the final answer, which does not always align well with the quality of the produced rationales [179]. Unsupervised metrics for the evaluation of step-by-step reasoning when no golden reference generation exists measure aspects such as semantic alignment to determine whether the generated reasoning is grounded within the source context, logical inference to check for hallucinations and logical fallacies, semantic similarity to quantify the relatedness between intermediate reasoning steps, and language coherence to establish the naturalness of the reasoning chain [143].

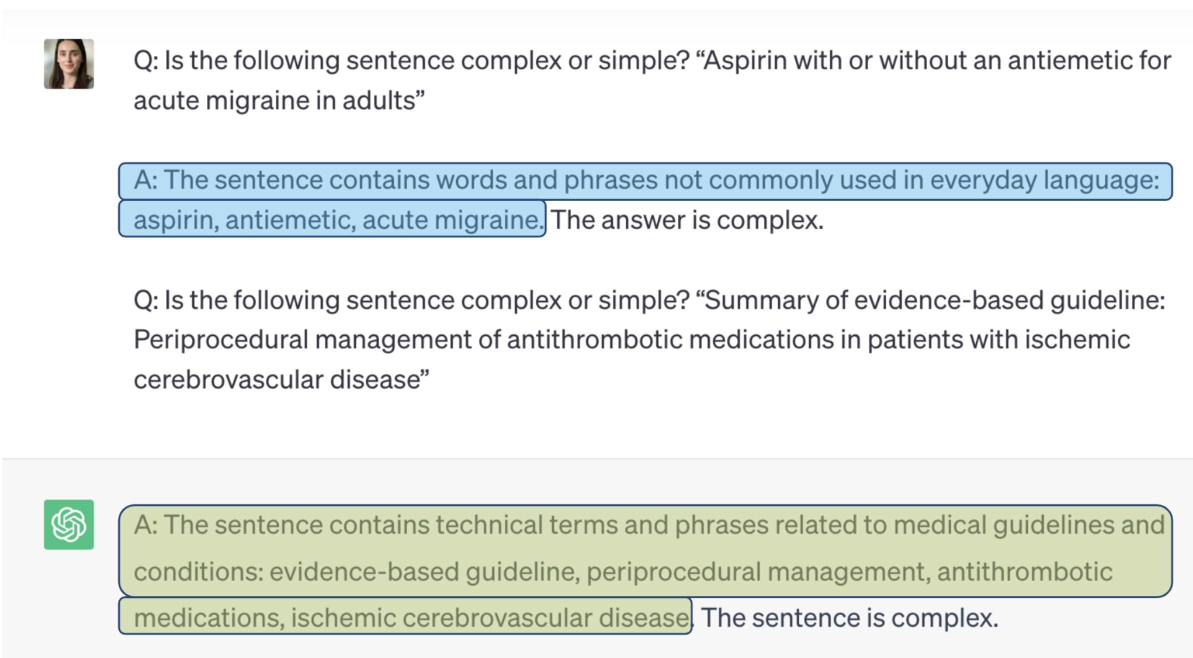


Figure 3.2: Chain-of-thought prompting for explainable prediction of text complexity.

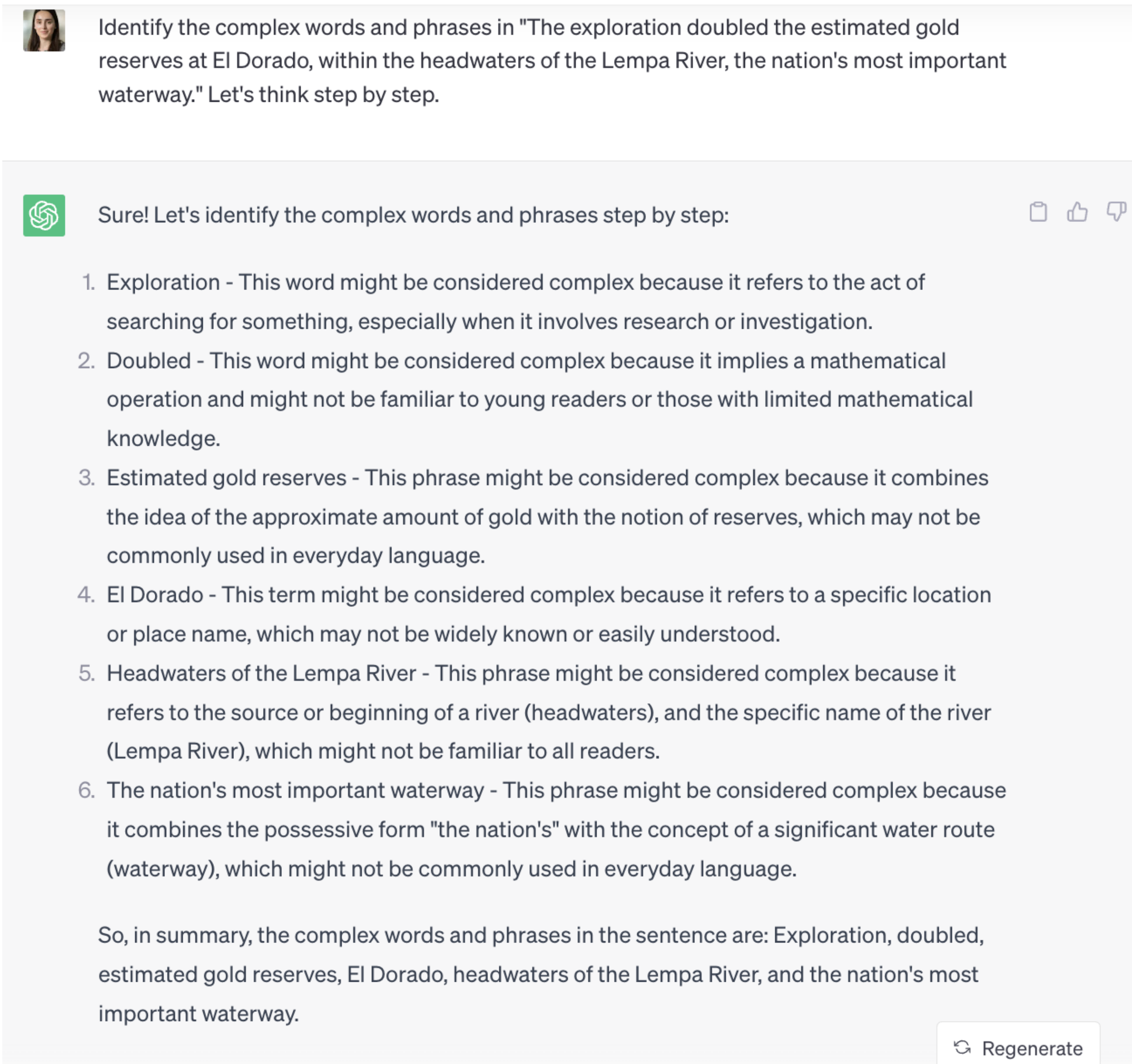


Figure 3.3: Chain-of-thought prompting for step-by-step explainable prediction of text complexity.

CHAPTER 4

Adapting Pre-trained Language Models to Low-Resource Text Simplification: The Path Matters

In this chapter, we focus on the challenge of absence of large-scale parallel (complex $\rightarrow$ simple) monolingual training data for the task of text simplification. Our main contribution is that we approach the problem of low-resource text simplification from a task and domain adaptation perspective and show there is benefit in bridging two popular paradigms of few-shot adaptive learning (transfer learning, meta-learning). This work is presented in:

- Gârbacea, Cristina, and Qiaozhu Mei. *"Adapting Pre-trained Language Models to Low-Resource Text Simplification: The Path Matters."* In Conference on Lifelong Learning Agents, pp. 1103-1119. PMLR, 2022.

We frame the problem of text simplification from a task and domain adaptation perspective, where neural language models are pre-trained on large-scale corpora and then adapted to new tasks in different domains through limited training examples. We investigate the performance of two popular vehicles of task and domain adaptation: meta-learning and transfer learning (in particular fine-tuning), in the context of low-resource text simplification that involves a diversity of tasks and domains. We find that when directly adapting a Web-scale pre-trained language model to low-resource text simplification tasks, fine-tuning based methods present a competitive advantage over meta-learning approaches. Surprisingly, adding an intermediate stop in the adaptation path between the source and target, an auxiliary dataset and task that allow for the decomposition of the adaptation process into multiple steps, significantly increases the performance of the target task. The performance is however sensitive to the selection and ordering of the adaptation strategy (task adaptation vs. domain adaptation) in the two steps. When such an intermediate dataset is not available, one can build a "pseudostop" using the target domain/task itself. Our extensive analysis serves as a preliminary step towards bridging these two popular paradigms of few-shot adaptive learning and towards developing more structured solutions to task/domain adaptation in a novel setting.

4.1 Introduction

Large-scale language models (such as those adopting the Transformer [463] architectures) have shown outstanding text generation capabilities. This demonstrates that with sufficient data, model capacity, and computational resource, generative models can learn distributions powerful enough to produce high-quality samples from complex domains. These conditions are however unrealistic in many NLP tasks and application scenarios, where abundant training examples either do not exist or are costly to label, and the computational resource required to train large neural language models from the scratch is a luxury. Text simplification, which aims to transform specialized/complex content into simpler text so that it is accessible to readers with low literacy skills, is such an scenario. Despite its difficulty, text simplification is critical for providing fairness and equitable information services to the broad population.

The predominant approach for building NLP solutions for low-resource scenarios relies on a transfer learning paradigm, which works by first training a language model on large general-domain datasets and then adapting or fine-tuning the pre-trained model to the downstream task in a specific domain and/or with a specific objective functions. Nevertheless, such transfer learning methods usually assume the source and target domains consist of the same feature space, which limits their performance in many practical situations where the target domain is qualitatively different from the generic corpora used to train the original language models [84]. Furthermore, the effectiveness of vanilla fine-tuning methods is still heavily dependent upon having adequate amounts of in-domain training data for the target task [62]; when this pre-requisite is not met, the generalization performance of deep models can be considerably limited, leading to model overfitting, catastrophic forgetting of general-domain knowledge, and negative transfer across tasks [218], [457], [498].

New approaches have been proposed in the literature to address these challenging issues, claiming various degrees of success on a diversity of benchmarks [99], [87]. Among these, *meta-learning* [459], [397], [171] has emerged as a promising general learning strategy suitable for few-shot learning and cross-domain generalization [252], [475]. A typical meta-learning approach frames the learning problem at two levels: *i) base learning*, where an inner/lower/base learning algorithm is focused on the quick acquisition of knowledge within each separate task it encounters, and *ii) meta-learning*, where an outer/upper/meta algorithm is focused on the slower extraction of information learned across all tasks and updates the inner learning algorithm such that the model it learns improves an outer learning objective. To solve a few-shot learning problem, meta-learning leverages a good number of similar few-shot *tasks* to learn how to adapt the base-learner to a *new* task for which also only a few labeled samples are available. Different approaches to meta-learning include metric learning [466], [429], [445], memory networks [392], [335], [313], [319] and gradient based learning [115], [512], [442]. Among them, gradient/optimization based meta-learning has emerged as an effective approach to addressing the few-shot learning problem [374]. Such learning settings lend themselves applicable to resource constrained problems where there is a distribution of tasks available.

In this work, we frame the problem of low-resource text simplification from a task and domain adaptation perspective. We consider the everyday use of text simplification in a wide variety of domains, including news and scientific articles (which naturally contain many subject areas), and view parallel complex-simple English language examples in different domains as samples drawn from a distribution over text generation tasks with varying constraints on the level of text complexity and readability. Once such a distribution is learned from large-scale, general purpose corpora (i.e., a pre-trained language model), it is fast adapted to new tasks and domains (in our case different text simplification scenarios) with few training examples. We consider two approaches to this problem: 1) a standard transfer learning practice that fine-tunes the general language model to the new domains of text simplification with limited in-domain data, and 2) simulate many domain adaptation tasks and use gradient based meta-learning to learn model parameters that can generalize to new tasks, again with few examples. We extensively compare these two approaches in our low-resource adaptation settings, and our experiments reveal that when directly adapting a general language model to the target tasks/domains, fine-tuning (i.e., domain-adaptation) remain competitive compared with meta-learning (i.e., task adaptation). Surprisingly, we find that adding an intermediate destination in between the source and target, i.e., first adapting the pre-trained language model to an auxiliary task/domain and then adapt the model to the target tasks/domains, significantly increases the performance of the target tasks. Adding a *stop* in the adaptation path allows each segment to use a different adaptation strategy (akin to a transportation method, or a "vehicle"), and the performance on target tasks is sensitive to which vehicle is taken in each segment. In particular, it is essential to perform domain adaptation through transfer learning (fine-tuning) in the second stage, and performing task adaptation via meta-learning in the first stage further improves the performance. Interestingly, when such an intermediate dataset is not available, one can build a "pseudostop" simply based on the target task/domain itself. Our findings serve as a novel step bridging the two popular paradigms of few-shot adaptive learning and towards developing more structured solutions to task/domain adaptation.

4.2 Related Work

The task of neural text simplification is similar in nature to neural machine translation, where transfer learning and meta-learning approaches have been widely applied to low-resource settings. Knowledge extracted from multilingual high-resource language pairs is leveraged for adapting machine translation systems to low-resource target languages, demonstrating the benefit of meta-learning over conventional multilingual translation approaches when limited in-domain training data is available [152]. Similarly, a meta-learning strategy is used to simulate many few-shot domain adaptation tasks to learn model parameters for fast adaptation to unseen language pairs in machine translation [412]. We consider a similar problem in a different application context.

Relevant to our work, a few-shot evaluation protocol is used to compare recent advances in transfer learning and meta-learning on standard visual classification benchmarks for task adaptation [99]. The authors find that meta-learning approaches struggle to generalize to out-of-distribution test tasks, and that their overall performance is inferior to transfer learning methods. While pre-training then fine-tuning remains a highly competitive baseline for few-shot classification tasks, simply scaling up the size of these pre-trained models does not result in any significant performance gain on out-of-distribution tasks. On the contrary, meta-learning methods are data efficient, but computational bottlenecks and implementation difficulties prohibit their use in combination with large scale backbones. Furthermore, having sufficient heterogeneous training tasks is a critical pre-requisite for meta-model training [198]; when source tasks present different characteristics from target tasks, the performance of meta-learning algorithms declines and results in poor generalization on unseen tasks. On the particular task of text classification, combining task-adaptive pre-training with domain adaptive pre-training results in performance gains [155]. This finding is in line with our work which confirms that multiple stages of adaptation result in improved performance on the end task/domain. However, unlike our work which is focused on text simplification in a multitude of domains (32 scientific domains and the news domain), [155] are focusing on the different task of text classification in four domains only and report that task-adaptive pre-training yields performance gains after domain adaptive pre-training; instead, our empirical results show the opposite order of adaptation is more effective.

Similarity between source and target (tasks and domains) represents an important predictive factor of successful adaptation [467], while increasing the size of the source dataset does not necessarily result in the largest transfer gains. When only scarce data is available for the target task, transfer learning remains beneficial.

Many recent approaches combine the strengths of meta-learning with pre-training. Meta-parameterized pre-training [372] "meta"-learns the pre-training hyperparameters, demonstrating that optimized meta-parameters improve the learnt representations and the predictive performance of the pre-trained model. Meta-finetuning [470] improves the fine-tuning of neural language models by meta-learning class prototypes and domain-invariant representations which are useful in solving groups of similar natural language tasks. Moreover, feature representations meta-learnt are clustered more tightly in the feature space than representations obtained through conventional training of neural networks/pre-train then fine-tune approaches, demonstrating that minimizing within-class feature variation is critical for robust few-shot performance on complex tasks [141].

Recent Trends in Low-Resource NLP Despite the great success of large-scale pre-trained language models on many NLP downstream tasks and benchmarks, they struggle in low-resource settings where only a handful of examples are available. Few-shot learning, the challenge of learning from a small number of examples, is critical for improving the robustness and efficiency of current models and for bridging the sample-efficiency gap between deep learning and human learning. It also alleviates the difficulty of collecting extensive expert-annotated datasets and reduces the costs associated with resource-intensive compute. In the literature, the goal of learning from few examples has been approached from different perspectives, including adaptation to new classes [19], adaptation to new tasks and domains [17], [44] or more recently as direct applications of large scale pre-trained language models [36], [334] via fine-tuning, in-context learning, one-shot or zero-shot prompting. Pre-training on a large general-purpose corpus followed by fine-tuning on a specific task requires task-specific fine-tuning datasets of (tens of) thousands examples, nevertheless recent advances in scaling up language models demonstrate great improvements that are competitive with state-of-the-art fine-tuning approaches when just leveraging their task-agnostic few-shot abilities. Unlike fine- tuning which involves updating the weights of a pre-trained model, few-shot requires no weight updates and works by providing the model with k examples of context and their corresponding completions, altogether with one final test example for which the model itself is expected to provide the completion. One-shot is similar to few-shot, however it only allows one demonstration and a natural language description of the task, while in zero-shot no demonstrations are allowed except for the instruction describing the task itself. Figure 4.1 and Figure 4.2 illustrate how few-shot learning improves dramatically with increasing the model size and the number of k-shot in-context examples provided [36]. The few-shot learning setup considerably diminishes the need for task-specific data, and is relevant for addressing the low-resource issue for many under-represented languages and resource-constrained settings [483].

In parallel with these advances that highlight emerging abilities of large-scale pre-trained models, other research directions explore the tradeoff between increasing the number of model parameters versus the number of labeled examples across a wide variety of NLP tasks [219], [239]. When hundreds of examples can be collected with relatively low cost and effort, the question is whether gains obtained from enlarging the dataset are comparable to gains obtained from using larger-sized models, since scaling up language models to hundreds of billions of parameters is not efficient and may restrict the ability to fine-tune (for eg., GPT3 [36] is only available as a service). To this end, the contribution of additional examples greatly varies depending on the task format [219]; please see Figure 4.3. In general, collecting more training data for tasks where data is sparse is often a more effective strategy than scaling to larger and more computationally demanding models. Tasks such as text classification, multiple choice or extractive question answering greatly benefit from

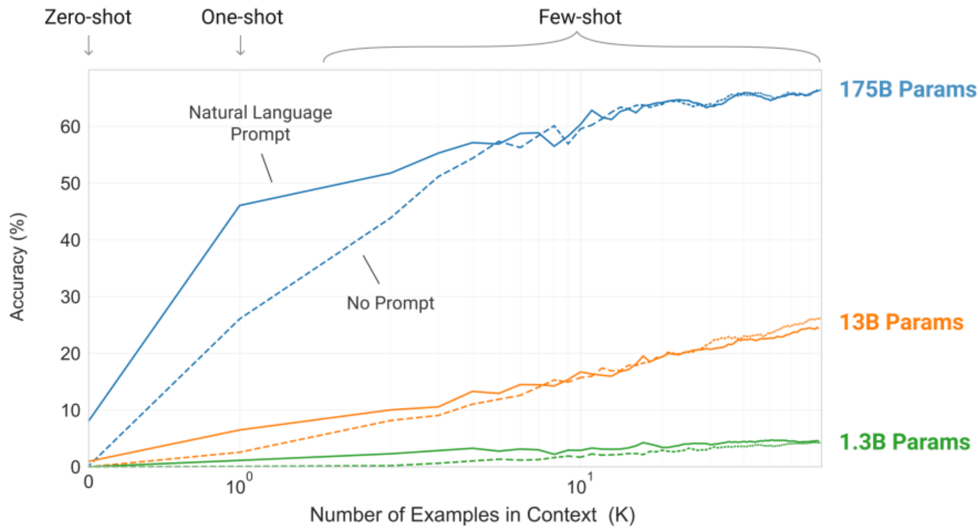


Figure 4.1: Larger models make increasingly efficient use of in-context information. The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information on a simple task requiring the model to remove random symbols from a word. Figure reproduced from (Brown et al, 2020)[36].

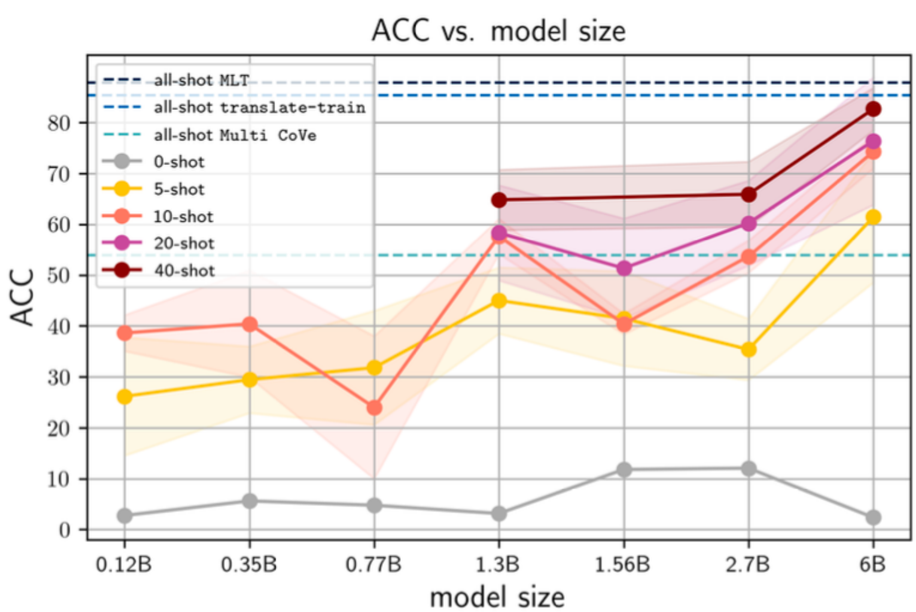


Figure 4.2: The average accuracy vs. model size on English-Spanish Multilingual Natural Language Understanding (NLU) dataset achieved by cross-lingual in-context learning using various GPT and T5 models. The shaded region represents the standard deviation of three runs. Figure reproduced from (Winata et al, 2021)[483].

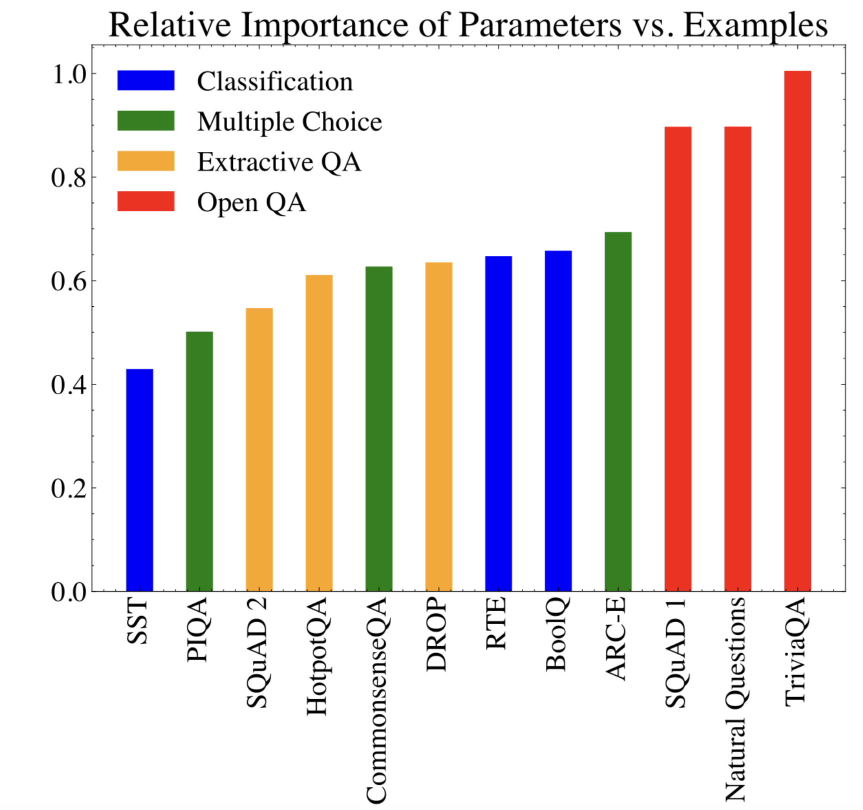


Figure 4.3: The relative importance of increasing the number of model parameters versus the number of training examples for various NLP tasks. Higher values indicate more dependence on parameters and less on labeled data. Figure reproduced from (Kirstain et al, 2021)[219].

additional training examples, which are considered to be "worth" billions of parameters (i.e. enlarging the dataset results in similar gains as increasing the number of parameters). However, for tasks such as open question answering which require the model to recall specific information seen during pre-training, enlarging the training set will not result in performance gains; in such cases additional model parameters are extremely important and cannot be replaced by more data.

For both fine-tuning and few-shot methods, prompting has been used in the low-data regime to introduce a task description in natural language. Prompt-based learning allows to customize the language model to predict output that reflects the downstream task of interest [36]. For the supervised task of text classification, prompting provides substantial advantage in terms of data efficiency and is found to be equivalent to hundreds of datapoints [239]. However, plain language prompts do not always produce the desired results and human users must experiment with a wide range of prompts to find those that elicit desired behaviours from black-box large language models [396], [382]; even if significant effort is invested, manual prompts are still likely to be sub-optimal. Instead of relying on human-designed prompts, approaches that automate the prompt generation and selection process by formulating it as a black-box optimization problem are proposed [531], [425], [126], [274]. Interestingly, single prompts such as *"Let's think step by step"* demonstrate zero-shot capabilities of large language models and can be used to elicit step-by-step reasoning in a zero-shot chain-of-thought manner for diverse logical tasks [221]. While zero-shot CoT yields less good results compared to few-shot CoT, the method outperforms few-shot and zero-shot prompting for reasoning tasks when used with large language models such as PaLM [65] with 540 billion parameters. Notably, the zero-shot and few-shot abilities of LLMs can be increased by fine-tuning models to follow instructions [391], [338], [396], [126], [526]. When the prompt is optionally augmented with few examples that reflect the target task of interest (demonstrations), the pre-trained model can solve a new test input by leveraging the in-context learning (ICL) capabilities learnt during pre-training; ICL does not require any gradient update to the model parameters. Nevertheless, model biases greatly influence ICL performance, causing it to vary from near random to near state-of-the-art [524]. LLM predictions appear more sensitive to demonstrations appearing closer to the test example, a phenomenon referred to as recency bias [524]. In addition, the order of demonstrations does matter: some orders of the training examples are "fantastic", while others cause a significant drop in performance [287]. In order to compare approaches for few-shot NLP, new benchmarks are proposed for evaluating the performance of current models [34].

4.3 Preliminaries to Few-shot Task/Domain Adaptation

We aim to learn how to adapt a pre-trained neural language model to text simplification contexts that involve new tasks and domains, with only few in-domain training examples available. For this purpose, we use widely popular adaptation strategies, namely gradient-based meta-learning and fine-tuning-based transfer learning. In what follows we formally introduce these two approaches for task and domain adaptation in the context of neural text simplification.

4.3.1 Gradient-based Meta-learning

In the context of few-shot learning, meta-learning models are designed to find parameters that can be fine-tuned in few optimization steps and with few labeled examples to achieve fast adaptation on a task not seen during training.

In typical machine learning settings, we are given a dataset $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, with a training split D_{train} and a testing split D_{test}. The goal is to train a model $\hat{y} = f_\theta(x)$ parameterized by θ such that model parameters θ are optimized on the training subset D_{train}: $\theta^* = \arg\min_\theta \mathcal{L}(\mathcal{D}; \theta, \omega)$, where $\mathcal{L}$ represents the loss function measuring the error between model predictions and ground-truth labels, and ω denotes assumptions such as the choice of the optimizer or function class for f. After training the model, we then evaluate its generalization performance on the testing subset D_{test}. The conventional assumption is that optimization is performed from scratch for every dataset D.

In meta-learning, we assume a distribution over tasks $p(\mathcal{T})$ we want our model to be able to adapt to, and that a set of tasks can be sampled from this distribution, $\{\mathcal{T}_i\}_{i=1}^n \sim p(\mathcal{T})$. Each task $\mathcal{T}_i = (\mathcal{T}_i^s, \mathcal{T}_i^q)$ consists of two small sets of labeled data, the support data $\mathcal{T}_i^s$ which is used for fine-tuning, and the query data $\mathcal{T}_i^q$ which is used for measuring the performance of the resulting fine-tuned model. Note that for different tasks $\mathcal{T}_i$ and $\mathcal{T}_j$, they may share the same data distribution (X) but just deal with different labels (Y). When the data distribution X_i and X_j are different across tasks, we can also describe them as sampled from different "domains". The task $\mathcal{T}_i$ is described as *n-way k-shot* if it consists of n classes, and there are k examples available for each class. In Algorithm 1 [140], we present the general gradient-based meta-learning framework, noting that variations of this approach exist in the literature.

Algorithm 1 The gradient-based meta-learning framework

Require: $p(\mathcal{T})$: distribution over tasks, F_θ: base model, $\mathcal{A}$: fine-tuning algorithm, γ: learning rate
Ensure: Initialize θ, the weights of F
  while not done **do**
    Sample a batch of tasks $\{\mathcal{T}_i\}_{i=1}^n \sim p(\mathcal{T})$, where $\mathcal{T}_i = (\mathcal{T}_i^s, \mathcal{T}_i^q)$
    for $i = 1, \ldots n$ **do**
      Fine-tune model on task $\mathcal{T}_i$ and obtain new network parameters $\theta_i = \mathcal{A}(\theta, \mathcal{T}_i^s)$ ▷ Inner Loop
      Compute gradient $g_i = \nabla_\theta \mathcal{L}(F_{\theta_i}, \mathcal{T}_i^q)$
    end for
    Update base model parameters ▷ Outer Loop
    $\theta \leftarrow \theta - \frac{\gamma}{n} \sum_i g_i$
  end while

In general, meta-learning algorithms employ a bi-level optimization scheme consisting of an "inner" loop and an "outer" loop, where the outer loop searches for the best global parameter initialization and the inner loop optimizes individual models that share a common parameter initialization for a range of tasks. A meta-learning iteration starts with the outer loop, where a batch of tasks are sampled from the distribution over tasks $p(\mathcal{T})$. Then in the inner loop, given as input a base model F_θ parameterized by network parameters θ, F_θ is in turn fine-tuned on the support data $\mathcal{T}_i^s$ of each task; the resulting fine-tuned model F_{θ_i} is used to make predictions on the query data $\mathcal{T}_i^q$ of each task. After the inner loop completes for all sampled tasks in the batch, the outer loop minimizes the loss on the query data with respect to the pre-finetuned weights; this outer optimization step is achieved by differentiating through the inner loop computation and updating the base model parameters θ such that the inner loop fine-tuning becomes as fast and efficient as possible. Importantly, meta-learning algorithms differentiate through the entire fine-tuning loop, unlike transfer learning approaches that only use simple first-order gradient information to update network parameters. Nevertheless, back-propagating meta-gradients through the inner loop comes with practical constraints such as high-order derivatives, big memory footprints, and the risk of vanishing or exploding gradients [184]. To alleviate these issues of hierarchical optimization where the outer optimization is constrained on the inner optimization, the number of inner fine-tuning steps k is often set to small value, for eg., $k = 1$; alternatively, approximations for higher order gradients are used when the outer/meta model and the inner/task-specific model lie in the same space.

4.3.2 Transfer learning

Transfer learning [51], [343] focuses on knowledge transfer across domains. It aims to improve the learning process of a target task with limited or no labeled training data by exploiting knowledge acquired from a different and related source domain/task which has sufficient data available. By enhancing the data in target domain with the additional data from the source domain, model performance on the target task can be considerably improved. Compared to meta-learning which includes an outer optimization loop to evaluate the benefit of prior knowledge when learning a new task, transfer learning extracts prior knowledge by learning on the source task directly (i.e. without the use of a meta-objective). Furthermore, while meta-learning seeks an “algorithmic” solution to the few-shot learning problem and does not necessarily focus on datasets and architectures, transfer learning approaches emphasize learning robust representations from large-scale datasets and models [99]. To this end, one of the most commonly employed approaches to transfer learning is *pretrain-then-finetune*, which first trains a model on massive datasets and then fine-tunes a pre-trained model on new tasks of interest that requires less data.

Numerous successes of large-scale pre-trained models on a wide variety of tasks and domains are reported in the literature [463], [36], [90]. Nevertheless, transferred knowledge does not always have a positive impact on new tasks. In the extremely data-scarce regime when only few samples are available in the target domain, transfer learning is less effective and performs subpar [140]. Furthermore, when there is little in common between domains or when domain similarities are misleading, the target learner is negatively impacted by the transferred knowledge and negative transfer occurs [536]. The brittleness of the fine-tuning process in settings where there is data distribution shift and different label space than seen during pre-training leads to poor out-of-domain generalization. Therefore, the main challenge in transfer learning becomes how to distinguish beneficial source knowledge from inherent cross-domain noise [84].

Relevant to our work, including a second stage of pre-training with intermediate supervised tasks is reported to improve the robustness and effectiveness of the resulting target task model in few-shot settings [358]. Crucially, a careful selection of source tasks to fine-tune on in the intermediate stage is still required, which is not always clear.

4.4 Experiment Setup

In our experiments we aim to investigate the robustness and efficacy of meta-learning and transfer learning methods when applied to the scenario of neural text simplification in a wide diversity of real-world data-constrained settings.

4.4.1 Datasets

We use three datasets which cover different domains and application scenarios of text simplification. In particular, we focus on generating simpler and more readable versions of news articles and scientific papers from a multitude of research fields that are disseminated to the general public. We use a third dataset, Wikipedia, as an auxiliary.

News Simplification. *Newsela* [496] is a corpus of news articles simplified by professional news editors for children of different age and grade levels for pre-college classroom use. Each article has been rewritten four times, resulting in a parallel sentence-aligned monolingual corpus with different reading levels. In our experiments we use the parallel Newsela dataset made available in [517], which we further divide into distinct subsets according to the ground-truth labels provided for complex-simple sentence pairs. In other words, we consider the different degrees (or difficulty level) of simplification as different *tasks* of new simplification. Table 4.6 in Appendix 4.7 summarizes our meta-train, meta-dev and meta-test splits according to the complexity level of sentence pairs.

Scientific Press Release. *Biendata* [1] dataset consists of research papers from various scientific disciplines matched with press releases that describe them. Notably, rewriting scientific papers into press releases has mixed objectives that are not solely about text simplification. However, it presents a valuable use-case scenario of text simplification employed in the real world and it also provides a nice out-of-distribution test for our adaptative learning methods. The corpus consists of alignments at the title level, and for each scientific paper we extract domain meta-data via the Microsoft Academic Knowledge API [2]. We define subsets of the Biendata corpus according to scientific domains, so simplifying articles in each scientific domain is also considered as a different

[1] https://www.biendata.xyz/competition/hackathon
[2] https://www.microsoft.com/en-us/research/project/microsoft-academic-graph

task; in Table 4.8 in Appendix 4.7 we present statistics regarding the data splits used for the meta-train, meta-dev and meta-test subsets.

Auxiliary. *WikiLarge* [517] is a Wikipedia-based corpus created by combining existing simplification datasets. The training subset of WikiLarge is obtained by assembling datasets of parallel aligned Wikipedia - Simple Wikipedia sentence pairs available in the literature [202], [485], [535], while the development and test subsets contain complex sentences with simplifications provided by Amazon Mechanical Turk workers [497]. We split the WikiLarge dataset at random ensuring equal number of complex-simple sentence pairs in each subset; please see Table 4.7 in Appendix 4.7 for details on the meta-train, meta-dev and meta-test subsets.

WikiSmall [535] is a smaller text simplification benchmark containing 89,042 training sentence pairs, and 100 testing complex-simple sentence pairs. We only use this corpus sparingly to augment WikiLarge model training.

Each dataset contains a meta-train, meta-dev and meta-test set, and each set includes text simplification tasks that correspond to varying complexity levels (Newsela and WikiLarge) or different scientific domains (Biendata). As these tasks are all based on their own dataset, we can also describe them as different "domains" of text simplification.

4.4.2 Adaptation Paths

Our main goal is to investigate whether meta-learning or transfer learning is a suitable adaptation strategy when there is a distribution of low-resource text simplification tasks/domains available, how they compare to each other, and whether they can work as a team. In addition, we would like to determine if doing both task adaptation and domain adaptation can improve performance on new target tasks of interest. To answer these research questions, we design the following experiments. In Figure 4.4, we see multiple possible paths of adaptation process.

1. *Direct task adaptation:* we aim to determine if it is possible to adapt a pre-trained model (Source), either trained on general-domain knowledge or specifically designed for text simplification, to text simplification tasks (Target) via meta-learning and achieve good performance on unseen tasks (meta-test);

2. *Direct domain adaptation:* we would like to establish whether a pre-trained model (Source) can be adapted to text simplification domains (Target) via fine-tuning and achieve good performance on unseen domains;

3. *Two-stage adaptation via an intermediate:* our goal is to determine if there is any benefit combining task adaptation and domain adaptation through an intermediate task/domain

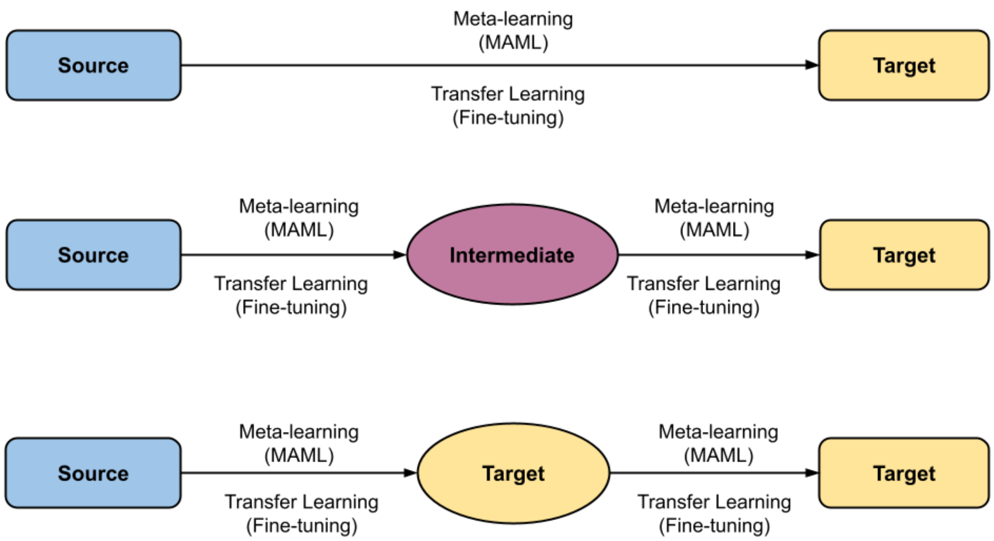


Figure 4.4: Adapting a pre-trained language model (source) to low-resource text simplification (target).

dataset (Intermediate). We would like to find out in which order the two-stage adaptation should be carried out;

4. *Two-stage adaptation via a pseudostop*: if no intermediate dataset is available, we would like to determine if it is possible to use the Target itself as the intermediate.

Note that in each leg of a two-stage process, we have two adaptation methods (meta-learning or transfer learning). Next, we present the specific details of the two methods in our context.

Meta-Learning (Task adaptation) For learning model parameters that facilitate adaptation to a new text simplification task in few steps and with minimal amount of text simplification examples, we use Model Agnostic Meta-Learning (MAML) [115]. In our experiments, we use the publicly available MAML implementation proposed in META-MT [412] [3], originally designed for fast adaptation in the context of neural machine translation with minimal amount of in-domain data. We adapt META-MT for the task of text simplification and use the first-order approximation of MAML (FOMAML) due to computational challenges associated with computing higher order gradients. The meta-learning loss function is optimized using Adam [215] optimizer with a learning rate $\alpha = 1e-5$ and a meta-batch size of 1. We initialize meta-parameters θ in two ways: *i)* by training a Transformer [463] model on the combination of WikiLarge and WikiSmall text simplification datasets, and *ii)* by leveraging the external knowledge encapsulated in the pre-trained Text-to-Text

[3] https://www.dropbox.com/s/jguxb75utg1dmxl/meta-mt.zip?dl=0

Transformer (T5) [371], a general purpose language model not particularly designed for the task of text simplification. We provide details on the hyper-parameter settings for these inner models in the transfer learning section below.

In addition, we also use Reptile [326], a first-order meta-learning algorithm designed for fast adaptation to new tasks. We intend to verify whether the same conclusions and insights can be drawn if different meta-learning algorithms are used for task adaptation. Our goal is not to compare the two meta-learning algorithms.

Transfer Learning/Fine-Tuning (Domain Adaptation) For determining the benefit of transfer learning from large-scale general-domain corpora to low-resource text simplification, we use the pre-trained Text-to-Text Transformer (T5) [371]. T5 is a sequence-to-sequence model with 60 million parameters pre-trained on a multi-task mixture of unsupervised and supervised tasks; each task is converted into a text-to-text format, thus allowing us to use T5 for the purpose of text simplification generation. We fine-tune T5 on the meta-train and meta-valid subsets of each text simplification dataset, then generate simplified outputs for the complex inputs from the meta-test test subset of each dataset by prompting the fine-tuned T5 with the keyphrase ”translate English to English”; training and validation batch size are set to 16, and the learning rate $\alpha = 1e - 4$.

In addition to T5, we also train a Transformer [463] model for text simplification on WikiLarge and WikiSmall datasets. In our implementation of the Transformer model, we use the Fairseq [337] library and following [412], we augment the Transformer architecture with adapter modules [172], [20] after each transformer block for more efficient model fine-tuning. The model we train relies on the transformer-base architecture with 6 encoder and 6 decoder layers and multi-head attention with 8 attention heads, the dimensionality of word embeddings is set to 512, feed-forward layers dimension is set to 2,048, and adapter modules have 32 hidden units; we use Adam [215] optimizer with a learning rate $\alpha = 7e - 4$.

4.4.3 Evaluation Metrics

There is no consensus on what is the single best evaluation metric for text simplification. We therefore employ a diverse portfolio of metrics to “meta”-assess the quality of the generated simplifications from different perspectives, including informativeness, relevance, fluency, readability, and adequacy. We use *SARI* [497] to evaluate the quality of the simplified output by comparing it against the source and reference simplifications, which is one of the most accepted metrics of text simplification in literature. We use *BLEU* [345] to measure the similarity between the generated text and gold standard references. We also use *FKGL* [214] to measure the readability of the output. In addition, we also use learnable evaluation metrics that train machine learning models on human annotated datasets to learn a scoring function that reproduces human judgements. *MoverScore* [523] measures the semantic distance between system outputs and reference texts using semantically aligned pretrained embeddings; the distance is computed using Word Mover's Distance [229] in the embedding space, yielding the amount of flow traveling between the contextualized representations. *MAUVE* [359] rewards model-generated text which resembles human-authored text by comparing the two distributions using Kullback-Leibler information divergence frontiers in a quantized low-dimensional embedding space. *BARTScore* [506] frames the evaluation of text generation models as a text generation problem, and uses BART [250] pre-trained sequence-to-sequence model to assess the probability of the system output (h) being generated from the source (s) and/or reference text (r). We apply BARTScore in different generation directions to determine *Faithfulness* ($s \rightarrow h$) as the likelihood the hypothesis could be generated based on the source text, *Precision* ($r \rightarrow h$) as the likelihood the hypothesis could be constructed based on the gold reference, *Recall* ($h \rightarrow r$) as the likelihood a gold reference could be generated by the hypothesis, and *F1* ($r \leftrightarrow h$) as the harmonic mean of precision and recall. Because the metric computes the average log-likelihood for the target tokens, the resulting BARTScore values are negative. Each of these evaluation metrics captures different aspects of text simplification generation, therefore in our analysis we account for their overall agreement with regards to the quality of the generated simplified output.

4.5 Experiment Results

We first intend to find out how well pre-trained language models can be directly adapted to the task of low-resource text simplification, if it is necessary at all (than training a model directly in the low-resource setting). If yes, we would like to understand whether the pre-trained model has to be purposed for text simplification or it can be trained on large-scale general purpose text corpora. If the latter, we would like to establish whether meta-learning or transfer learning works better (via one-stage adaptation) to transfer the general knowledge to text simplification, especially to new simplification domains with scarce data. Furthermore, we would like to determine if task and domain adaptation can complement each other as part of a two-stage adaptation approach; if yes, we would like to determine in which order the two-stage adaptation process should be carried out to optimize performance on out-of-distribution text simplification tasks and domains. Finally, when a natural intermediate dataset is not possible, we would like to determine whether using the target dataset itself for intermediate adaptation is a sensible alternative.

Baseline (No Adaptation). As a baseline, we do not adapt any pre-trained language models but train a Transformer model directly on Newsela or Biendata (based on meta-train and meta-dev) and evaluate its performance on the corresponding meta-test. Because the training data is limited, we anticipate that the performance would be suboptimal. Results are presented in Table 4.1. While the Transformer model generally scores high in BLEU (indicating similarity to reference) and low in FKGL (indicating high readability), the rest of evaluation metrics all indicate a sub-par performance to adaptation-based methods (see below), including SARI. Note that the FKGL score on Biendata is much lower than that of the ground-truth simplifications, indicating that the model has been oversimplifying the scientific content. For reference, we also include the results on WikiLarge dataset, which are much better than the other two datasets. As WikiLarge is the largest dataset of the three, this suggests that when training data is abundant, neural text simplification could yield good performance without adaptation from a pre-trained language model.

Table 4.1: Baseline meta-test test set results for the Transformer model trained on each text simplification dataset. Baseline FKGL reference scores - *Newsela: 3.733, Biendata: 9.692, WikiLarge: 5.973*; ⋆ denotes over-simplification.

Dataset	**Method**	**SARI (↑)**	**BLEU (↑)**	**FKGL (↓)**	**MOVER (↑)**	**MAUVE (↑)**	**BARTScore (↑)**			
							Faithfulness	*P*	*R*	*F1*
Newsela	Transformer	38.888	10.294	4.371	0.193	0.471	-3.806	-4.770	-4.686	-4.677
	ACCESS	27.062	13.501	8.097	0.250	0.326	-1.827	-4.276	-3.511	-3.793
	DMLMTL	38.601	8.181	⋆1.476	0.164	0.229	-4.004	-5.070	-4.767	-4.850
Biendata	Transformer	35.950	0.455	⋆6.715	0.143	0.019	-6.164	-5.924	-6.517	-6.153
	ACCESS	17.847	2.582	12.552	0.294	0.446	-1.747	-5.821	-5.654	-5.678
	DMLMTL	32.960	0.893	⋆9.180	0.154	0.053	-4.429	-6.655	-6.280	-6.393
WikiLarge	Transformer	47.361	41.557	⋆5.520	0.375	0.818	-2.380	-3.408	-3.687	-3.469
	ACCESS	36.075	32.577	8.536	0.356	0.325	-2.016	-3.864	-3.749	-3.734
	DMLMTL	31.233	3.425	⋆1.650	0.113	0.034	-3.731	-4.871	-5.487	-5.072

Additional Baselines (Pre-trained Text Simplification Models) In addition, we also select the current best text simplification models from the literature which have released pre-trained models. ACCESS [298] is a controllable sequence-to-sequence simplification model reported highest performance on WikiLarge, while Dynamic Multi-Level Multi-Task Learning for Sentence Simplification (DMLMTL) [153] reported the highest performance on Newsela. We evaluate these pre-trained text simplification models on our own data splits. The results on Newsela and WikiLarge are mostly consistent with literature. However, the performance of both pretrained models degrades significantly on Biendata, which demonstrates the critical need for task/domain adaptation. In fact, the Transformer model directly trained on each dataset outperforms both pretrained models on most metrics.

Direct task adaptation. In Table 4.2 we present results for adapting a pre-trained language model to the target task (Newsela or Biendata) through MAML. We test two source language models: one is the general-purpose T5 model released by Google and the other is the Transformer model we trained on WikiLarge (Wiki), which is purposed for text simplification. We observe that using T5 for MAML meta-parameter initialization yields better performance on new text simplification tasks according to the majority of evaluation metrics. The Transformer model trained on WikiLarge, although already purposed for text simplification, only achieves a higher BLEU and a better FKGL. Overall, adapting from a powerful pre-trained language model outperforms training a model directly from the limited resource (Table 4.1). To test the robustness of the results, we replace MAML with Reptile and the same pattern is observed (Table 4.2). These is not a consensus among the metrics whether MAML or Reptile is better in this task - note that our goal is not to compare the two meta learning models but rather the paths of task/domain adaptation.

Table 4.2: Direct task adaptation results on Newsela and Biendata meta-test test set; we use meta-learning (MAML, Reptile) for adapting an existing language model to new tasks. ⋆ denotes over-simplification according to FKGL.

Dataset	Method	SARI (↑)	BLEU (↑)	FKGL (↓)	MOVER (↑)	MAUVE (↑)	BARTScore (↑)			
							Faithfulness	*P*	*R*	*F1*
Newsela	MAML T5	25.343	**16.038**	8.096	**0.276**	**0.785**	**-1.195**	**-3.820**	**-3.208**	**-3.419**
	MAML Wiki	**36.025**	12.310	**5.184**	0.204	0.172	-3.257	-4.731	-5.014	-4.784
	Reptile T5	22.758	16.264	9.133	0.277	0.773	-1.068	-3.800	-3.121	-3.359
	Reptile Wiki	20.391	14.892	9.171	0.267	0.624	-1.114	-3.808	-3.081	-3.339
Biendata	MAML T5	25.804	**1.563**	16.348	**0.178**	**0.149**	**-2.558**	**-5.702**	**-5.755**	**-5.632**
	MAML Wiki	**35.548**	0.240	⋆**7.243**	0.122	0.004	-6.527	-6.584	-6.708	-6.586
	Reptile T5	18.867	1.587	16.916	0.166	0.081	-1.872	-5.429	-5.712	-5.434
	Reptile Wiki	10.004	1.384	18.876	0.137	0.027	-1.524	-4.980	-5.818	-5.262

Direct domain adaptation. In Table 4.3 we present results for adapting the pre-trained language models (T5 and Wiki) through fine-tuning to new text simplification domains in Newsela and Biendata meta-test test set. In line with our previous findings, using T5 as source yields superior performance to the Transformer trained on WikiLarge, according to most metrics. Comparing one-stage domain adaptation (Table 4.3) with one-stage task adaptation (Table 4.2), we observe that by and large domain adaptation (fine-tuning) outperforms task adaptation (either MAML or Reptile);

the benefit is more apparent on the out-of distribution scientific press release tasks and domains on Biendata. As using T5 as the source dominates the pre-trained Transformer on WikiLarge, we use T5 as the Source in there after.

Table 4.3: Direct domain adaptation results on Newsela and Biendata meta-test test set; we use fine-tuning for adapting an existing language model to new domains. ⋆ denotes over-simplification according to FKGL.

Dataset	Method	SARI (↑)	BLEU (↑)	FKGL (↓)	MOVER (↑)	MAUVE (↑)	BARTScore (↑)			
							Faithfulness	*P*	*R*	*F1*
Newsela	Fine-tune T5	32.310	**19.547**	7.638	**0.298**	**0.453**	**-1.624**	**-3.874**	**-3.162**	**-3.415**
	Fine-tune Wiki	**35.360**	18.009	**4.397**	0.272	0.158	-2.603	-4.847	-4.677	-4.676
Biendata	Fine-tune T5	35.989	**3.314**	11.279	**0.240**	**0.659**	**-3.324**	**-5.319**	**-5.519**	**-5.352**
	Fine-tune Wiki	**37.314**	1.066	⋆**6.827**	0.177	0.004	-5.652	-6.094	-6.193	-6.068

Two-stage adaptation. Next, we would like to investigate whether it is possible to combine the advantage of adapting to new domains with adapting to new tasks for more robust performance and better generalization on new text simplification tasks and domains. As part of a two-stage adaptation process, we aim to determine the ideal order in which to perform the adaptation, i.e. whether task adaptation should be performed ahead of domain adaptation, or vice versa. In addition, we explore if consecutive stages of task adaptation and domain adaptation could further improve upon one-stage adaptation results; nevertheless, our expectation is that combining task with domain adaptation adds complementary benefits and can outperform multiple stages of the same type of adaptation. In our analysis, we differentiate two cases: *i)* when there is an intermediate text simplification dataset available, and *ii)* when no other dataset is available, except for the source and target datasets.

Intermediate dataset available. We use WikiLarge as an intermediate dataset for task and domain adaptation. As part of the two-stage adaptation process, we first adapt pre-trained T5 to WikiLarge, then continue to adapt the resulting model to Newsela or Biendata; we explore possible combinations of task and domain adaptation at each stage of the pipeline, and present results for various combinations of the two-stage adaptation process on the Newsela and Biendata target tasks and domains in Table 4.4. Additionally, in Table 4.9 we include intermediate adaptation results on WikiLarge.

Table 4.4: Two-stage adaptation results on Newsela and Biendata meta-test test set when an intermediate dataset (WikiLarge) is used. **BOLD** and Underlined: best and second best within the block of either MAML or Reptile). ⋆ denotes over-simplification according to FKGL. Best single stage adaptation results under each metric included for reference but not highlighted in comparison. (*: results duplicated for comparison purposes.)

Dataset	Method	SARI (↑)	BLEU (↑)	FKGL (↓)	MOVER (↑)	MAUVE (↑)	BARTScore (↑)			
							Faithfulness	*P*	*R*	*F1*
Newsela	T5 domain + domain	**34.722**	19.376	8.319	0.284	0.079	-2.081	-4.182	-3.244	-3.596
	T5 domain + task (MAML)	24.681	16.971	8.415	0.283	0.834	**-1.102**	-3.771	-3.133	-3.354
	T5 task (MAML) + task (MAML)	28.711	15.378	7.462	0.267	0.619	-1.405	-3.821	-3.277	-3.457
	T5 task (MAML) + domain	33.978	**20.876**	**7.398**	**0.312**	**0.858**	-1.226	**-3.592**	**-3.096**	**-3.259**
	T5 domain + domain*	**34.722**	19.376	8.319	0.284	0.079	-2.081	-4.182	-3.244	-3.596
	T5 domain + task (Reptile)	23.467	17.042	9.442	0.285	0.787	**-1.037**	-3.772	**-3.095**	-3.332
	T5 task (Reptile) + task (Reptile)	26.337	17.098	8.727	0.278	0.818	-1.186	-3.829	-3.207	-3.432
	T5 task (Reptile) + domain	34.092	**20.861**	**8.054**	**0.311**	**0.876**	-1.237	**-3.613**	-3.098	**-3.270**
	best single stage	36.025	19.547	4.397	0.298	0.785	-1.195	-3.820	-3.162	-3.415
Biendata	T5 domain + domain	**37.850**	3.342	**10.932**	0.230	0.580	-3.700	**-5.244**	**-5.529**	**-5.315**
	T5 domain + task (MAML)	23.059	2.479	14.939	0.200	0.170	**-2.161**	-5.811	-5.671	-5.671
	T5 task (MAML) + task (MAML)	27.300	1.056	16.545	0.142	0.076	-3.252	-5.658	-5.907	-5.683
	T5 task (MAML) + domain	36.129	**3.419**	11.386	**0.236**	**0.596**	-3.270	-5.307	-5.530	-5.348
	T5 domain + domain*	**37.850**	3.342	**10.932**	0.230	0.580	-3.700	**-5.244**	**-5.529**	**-5.315**
	T5 domain + task (Reptile)	18.407	3.366	14.959	0.225	0.181	**-1.348**	-5.718	-5.532	-5.563
	T5 task (Reptile) + task (Reptile)	22.274	1.982	15.043	0.182	0.118	-2.168	-5.684	-5.771	-5.651
	T5 task (Reptile) + domain	37.175	**3.598**	11.063	**0.238**	**0.626**	-3.418	-5.263	-5.561	-5.341
	best single stage	37.314	3.314	⋆ 6.827	0.240	0.659	-2.558	-5.319	-5.519	-5.352

When an intermediate text simplification dataset is available as part of the two-stage adaptation process, our results indicate that the most promising strategy is to adapt to new tasks (through MAML or Reptile) in the first stage, and continue adapting to new domains (through fine-tuning) in the second stage. The benefit of doing task adaptation first is also supported by the intermediate results on WikiLarge, where adapting the pre-trained T5 model to the new task of text simplification yields better results than adapting to new domains. It is critical to do domain adaptation in the final stage (*domain* + *domain* is only second to *task* + *domain*), suggesting that the difference over data distributions is more critical than the difference over tasks in our scenario. This is particularly true on Biendata, where the content in different scientific domains may be very different. Repeating the same type of (task/domain) adaptation in both stages is less effective than *task* + *domain*, demonstrating the complementary benefit of learning to adapt to both tasks and domains for more robust generalization. Compared to one-stage task and one-stage domain adaptation, a two-stage task and domain adaptation consistently improves the quality of the generated simplifications according to the great majority of evaluation metrics. The findings are consistent when either MAML or Reptile is used for task adaptation.

No intermediate dataset available. While we have established the advantage of a two-stage adaptation procedure to address new text simplification tasks and domains, a potential limitation of this approach is the reliance on a third text simplification dataset as the intermediate. Given the scarcity of labels, we cannot assume the existence of such a related intermediate dataset for adaptation is always guaranteed. In such cases, we investigate whether it is possible to circumvent this additional requirement by using the source or target dataset itself for intermediate adaptation in the two-stage pipeline. In our experiments, we pick the Target dataset for intermediate adaptation, since the simplification model we aim to learn needs to be tailored specifically to target tasks and domains, and also we do not have access to the original dataset that T5 is trained upon. In Table 4.5 we present results when we adapt from *(Source → Target) → Target*, i.e. from *(pre-trained T5 → Newsela/Biendata) → Newsela/Biendata* tasks and domains via a two-stage adaptation process. In general, task adaptation followed by domain adaptation remains the best performing adaptation strategy, and the benefit is considerably more pronounced on the out-of-distribution tasks and domains from Biendata. We compare these results with Table 4.4 where an intermediate dataset is used, and observe that using the target dataset directly for intermediate adaptation yields slightly lower but comparable results to relying on Wikipedia for intermediate adaptation, and therefore successfully overcomes the need for extra data. The findings are mostly consistent when either MAML or Reptile is used for task adaptation. For interested readers, we have included sample outputs of both one-stage and two-stage adaptation paths in Table 4.10 in Appendix.

Table 4.5: Two-stage adaptation results on Newsela and Biendata when no intermediate dataset is available and the target dataset is used as a pseudo-intermediate. **BOLD**: best result within block of either MAML and Reptile

.

Dataset	Method	SARI (↑)	BLEU (↑)	FKGL (↓)	MOVER (↑)	MAUVE (↑)	BARTScore (↑)			
							Faithfulness	*P*	*R*	*F1*
Newsela	T5 task (MAML) + domain	**34.690**	**19.799**	**7.601**	0.294	0.251	-1.844	-4.012	-3.183	-3.487
	T5 domain + task (MAML)	30.878	19.141	7.732	**0.299**	**0.882**	**-1.323**	**-3.685**	**-3.136**	**-3.324**
	T5 task (Reptile) + domain	**35.865**	**21.788**	**7.768**	**0.317**	0.794	-1.286	**-3.589**	**-3.112**	**-3.268**
	T5 domain + task (Reptile)	30.501	19.478	8.362	0.301	**0.883**	**-1.172**	-3.689	-3.120	-3.316
Biendata	T5 task (MAML) + domain	**37.945**	**3.354**	**10.681**	**0.238**	**0.634**	-3.735	**-5.312**	-5.511	**-5.343**
	T5 domain + task (MAML)	32.821	2.973	13.582	0.233	0.467	**-3.149**	-5.763	**-5.508**	-5.564
	T5 task (Reptile) + domain	**37.175**	**3.599**	**11.063**	0.238	**0.626**	-3.417	**-5.263**	-5.561	**-5.341**
	T5 domain + task (Reptile)	31.885	3.441	13.853	**0.243**	0.464	**-2.646**	-5.703	**-5.431**	-5.502

In Figure 4.5 we demonstrate the effectiveness of the proposed one-stage and two-stage adaptation paths compared to the Transformer baseline. When gradually increasing the number of labeled training examples, we observe the faster convergence of the two adaptation paths compared to the baseline for the majority of evaluation metrics.

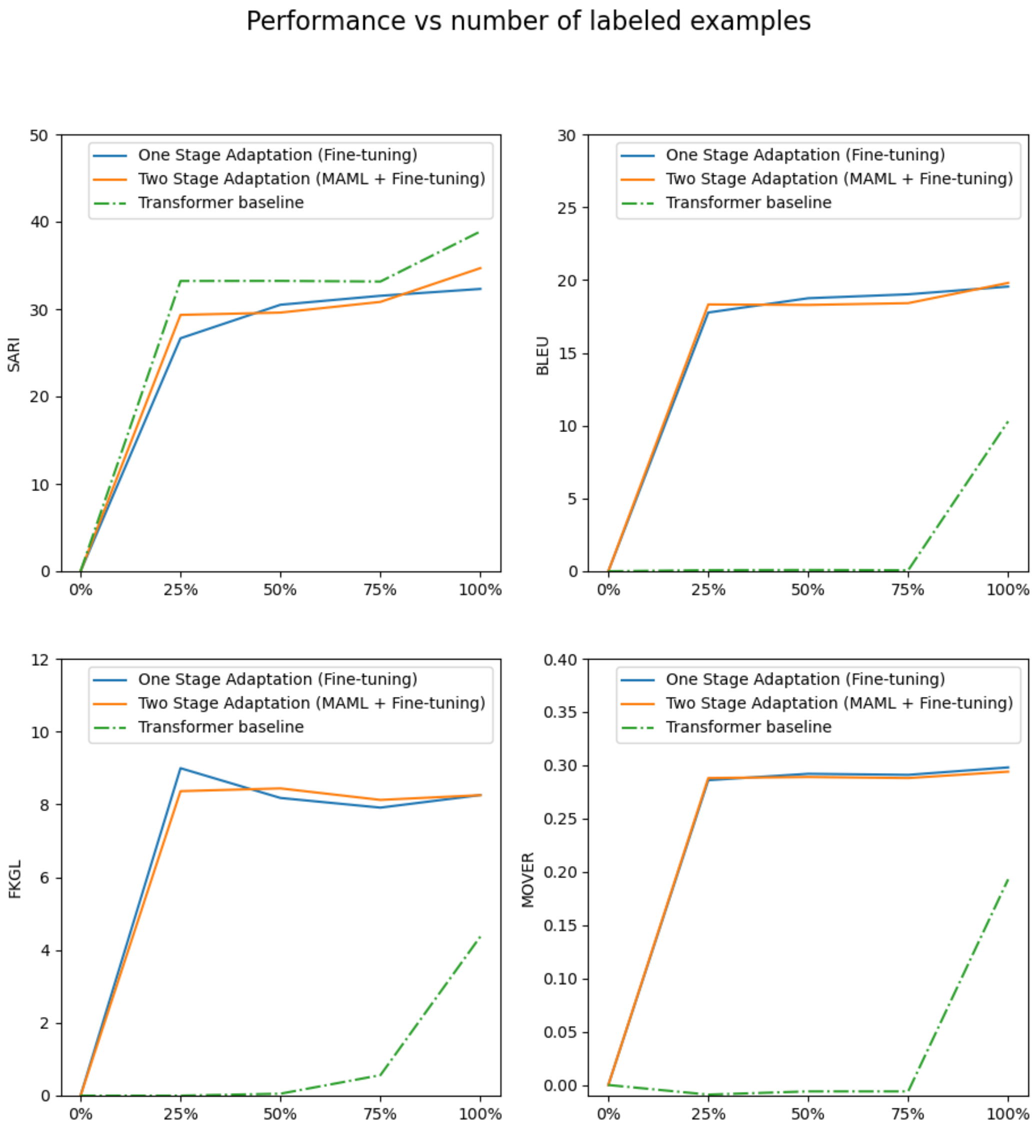


Figure 4.5: Performance of adaptation strategies with varying number of training examples.

4.6 Conclusion and Future Work

In this work, we frame the problem of low-resource text simplification from a task and domain adaptation perspective and learn how to quickly adapt pre-trained language models to new tasks and domains with few training examples. We examine the performance of state-of-the-art gradient-based meta-learning for task adaptation, and transfer learning from large-scale pre-trained language models for domain adaptation in a variety of tasks and domains. Our analysis reveals that when a direct adaptation approach is used, fine-tuning pre-trained language models outperforms meta-learning models for the task of low-resource text simplification; this trend is in line with previous findings in the literature [99], [36]. Nevertheless, decomposing the adaptation process into multiple steps can significantly increase target performance, provided that an auxiliary dataset is available for intermediate adaptation and careful attention is paid to performing adaptation in the correct order, i.e. task adaptation ahead of domain adaptation. When such an intermediate auxiliary dataset is not readily available, a "pseudostop" based on the target task/domain itself can be build between the source and the target.

Our findings represent preliminary foundations for proposing adaptation models that simultaneously perform task and domain adaptation in one goal. As we observe, the coupling of task adaptation (difference in Y) and domain adaptation (difference in X) is clearly beneficial comparing to either meta-learning or transfer-learning alone. Therefore creating a model that explicitly and jointly handles these two situations is a promising direction to explore. The utilization of stops (and even pseudostops) between the source and target tasks/domains also suggests that it may be valuable to further investigate a more structured solution of task/domain adaptation. We hope our insights will help inform future directions towards robust adaptation of neural language models to new tasks and domains for few-shot text simplification and other low-resource NLP tasks.

4.7 Appendix A1

Table 4.6: Newsela splits according to complexity level. 0 denotes the most complex level, and 4 represents the simplest.

Complexity level	Sentence pairs	TRAIN (70%)	DEV (15%)	TEST (15%)	
0 - 1	16,611	11,627	2,492	2,492	**META-TRAIN**
0 - 2	20,122	14,086	3,018	3,018	
0 - 3	19,891	13,923	2,984	2,984	
1 - 2	12,888	9,022	1,933	1,933	
1 - 3	13,296	9,308	1,994	1,994	
2 - 3	12,146	8,502	1,822	1,822	
2 - 4	9,780	6,846	1,467	1,467	**META-DEV**
3 - 4	10,185	7,129	1,528	1,528	
0 - 4	16,086	11,260	2,413	2,413	**META-TEST**
1 - 4	10,577	7,403	1,587	1,587	

Table 4.7: WikiLarge random splits.

Subset	Sentence pairs	TRAIN (50%)	DEV (25%)	TEST (25%)	
Wikipedia 0	20,000	10,000	5,000	5,000	**META-TRAIN**
Wikipedia 1	20,000	10,000	5,000	5,000	
Wikipedia 2	20,000	10,000	5,000	5,000	
Wikipedia 3	20,000	10,000	5,000	5,000	
Wikipedia 4	20,000	10,000	5,000	5,000	
Wikipedia 5	20,000	10,000	5,000	5,000	
Wikipedia 6	20,000	10,000	5,000	5,000	
Wikipedia 7	20,000	10,000	5,000	5,000	
Wikipedia 8	20,000	10,000	5,000	5,000	
Wikipedia 9	20,000	10,000	5,000	5,000	
Wikipedia 10	20,000	10,000	5,000	5,000	**META-DEV**
Wikipedia 11	20,000	10,000	5,000	5,000	
Wikipedia 12	20,000	10,000	5,000	5,000	
Wikipedia 13	20,000	10,000	5,000	5,000	
Wikipedia 14	20,000	10,000	5,000	5,000	
Wikipedia 15	20,000	10,000	5,000	5,000	**META-TEST**
Wikipedia 16	20,000	10,000	5,000	5,000	
Wikipedia 17	20,000	10,000	5,000	5,000	
Wikipedia 18	20,000	10,000	5,000	5,000	

Table 4.8: Biendata splits according to scientific domain.

Scientific Domain	Sentence pairs	TRAIN (50%)	DEV (25%)	TEST (25%)	
Medicine	7,993	3,997	1,998	1,998	
Biology	10,040	5,020	2,510	2,510	
Internal Medicine	1,095	547	274	274	
Psychology	3,367	1,683	842	842	
Chemistry	1,516	758	379	379	
Cancer Research	1,044	522	261	261	**META-TRAIN**
Neuroscience	1,411	705	353	353	
Virology	1,106	554	276	276	
Pediatrics	812	406	203	203	
Disease	582	292	145	145	
Immunology	2,281	1,141	570	570	
Genetics	2,151	1,075	538	538	
Social Psychology	1,090	546	272	272	
Surgery	1,261	631	315	315	
Psychiatry	1,045	523	261	261	**META-DEV**
Cognition	662	330	166	166	
Demography	992	496	248	248	
Climate Change	847	423	212	212	
Zoology	645	323	161	161	
Endocrinology	1,582	790	396	396	
Cell Biology	2,154	1,076	539	539	
Molecular Biology	904	452	226	226	
Biochemistry	640	320	160	160	
Physical Therapy	1,189	595	297	297	
Nanotechnology	378	188	95	95	
Gerontology	649	325	162	162	**META-TEST**
Computer Science	739	369	185	185	
Physics	1,108	554	277	277	
Materials Science	967	483	242	242	
Ecology	2,869	1,435	717	717	
Geography	658	330	164	164	
Economics	384	192	96	96	

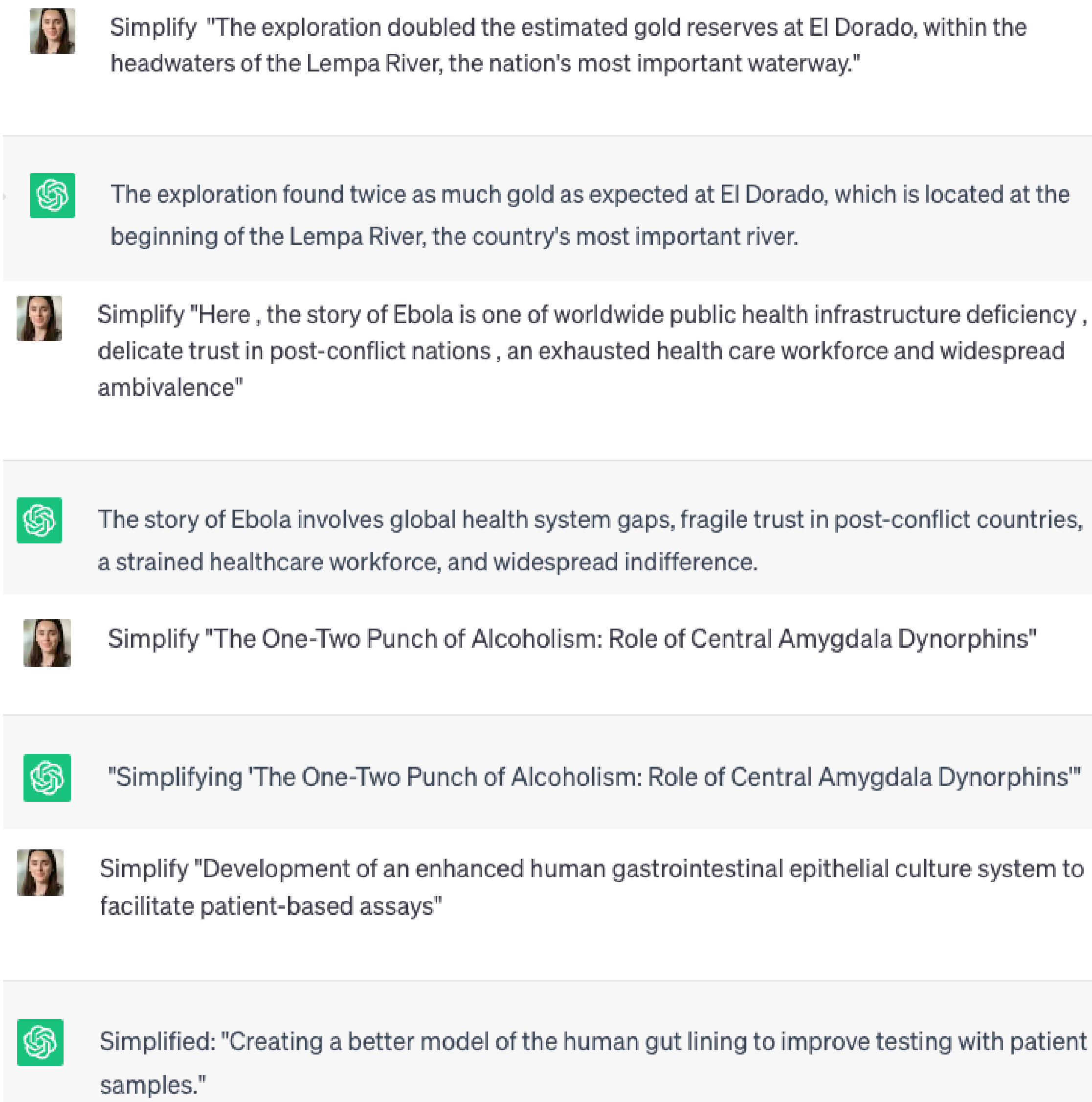
Table 4.9: Intermediate results on WikiLarge meta-test test set as part of the two-stage adaptation process. Baseline FKGL score for WikiLarge reference sentences: 5.973, $\star$ denotes over-simplification.

Dataset	Method	SARI (↑)	BLEU (↑)	FKGL (↓)	MOVER (↑)	MAUVE (↑)	BARTScore (↑)			
							Faithfulness	*P*	*R*	*F1*
WikiLarge	T5 Task Adaptation	**32.954**	34.722	6.584	**0.344**	**0.324**	**-1.286**	**-3.357**	**-3.411**	**-3.281**
	T5 Domain Adaptation	29.276	**42.608**	$\star$**5.175**	0.319	0.161	-1.816	-3.589	-3.932	-3.638

Table 4.10: Sample text simplification outputs through different adaptation paths.

Dataset	Model	Output
Newsela	**Complex sentence**	The exploration doubled the estimated gold reserves at El Dorado, within the headwaters of the Lempa River, the nation's most important waterway.
	Transformer	It found that the gold of the nation 's most important event , the nation 's most important event .
	Direct Task Adaptation	El Dorado is within the headwaters of the Lempa River, the nation's most important waterway.
	Direct Domain Adaptation	El Dorado, the nation's most important waterway, is within the headwaters of the Lempa River.
	Two-stage Task + Domain Adaptation	El Dorado is within the headwaters of the Lempa River, the nation's most important waterway.
	Simple sentence	The dig for gold would take place at a mine known as El Dorado.
	Complex sentence	Here , the story of Ebola is one of worldwide public health infrastructure deficiency , delicate trust in post-conflict nations , an exhausted health care workforce and widespread ambivalence to our duty as a global community .
	Transformer	Here , the story of Ebola is one of the public worldwide worldwide worldwide worldwide .
	Direct Task Adaptation	Here, the story of Ebola is one of worldwide public health infrastructure deficiency, delicate trust in post-conflict nations, an exhausted health care workforce and widespread ambivalence to our duty as a global community .
	Direct Domain Adaptation	The story of Ebola is one of the biggest problems in public health.
	Two-stage Task + Domain Adaptation	There is a lack of trust in post-conflict nations.
	Simple sentence	Ebola 's spread is about a worldwide lack of systems that support public health such as health care workers , hospitals and equipment .
Biendata	**Complex sentence**	The One-Two Punch of Alcoholism: Role of Central Amygdala Dynorphins
	Transformer	Researchers identify new way to reduce risk of childhood cancer
	Direct Task Adaptation	Alcoholism One-Two Punch of Alcoholism: Role of central Amygdala Dynorphins
	Direct Domain Adaptation	Alcoholism: How do we react to alcohol?
	Two-stage Task + Domain Adaptation	Alcoholism: How do we manage alcohol?
	Simple sentence	Alcoholism treatment: Kappa opioid receptors a new target
	Complex sentence	Development of an enhanced human gastrointestinal epithelial culture system to facilitate patient-based assays
	Transformer	Scientists discover new insights into how to fight against cancer
	One-stage Task Adaptation	Development of human gastrointestinal epithelial culture system to facilitate patient-based assayss
	Direct Domain Adaptation	Human gastrointestinal epithelial culture system could help treat patients
	Two-stage Task + Domain Adaptation	New way to test for gastrointestinal disorders
	Simple sentence	Growing human GI cells may lead to personalized treatments

In Figure 4.6 we present ChatGPT [333] generated simplifications for the same complex examples as in Table 4.10; in general, ChatGPT seems to follow the instruction to generate simpler and more accessible language for complex input texts. As our work was done before large language models as ChatGPT [333] or GPT-4 [334] were released, we leave the extensive quantitative and qualitative evaluation of these models for future work. In addition, as of the time of writing this thesis, the performance of these models is found to change substantially in a relatively short amount of time, and details related to when and how these models are updated are opaque to the general public [57]. For example, GPT-4's accuracy dropped from 97.6% in March 2023 to 2.4% in June 2023 on the task of identifying prime numbers, while GPT-3.5's accuracy has improved from 7.4% to 86.8% on the same task; variations in performance due to LLM drifts are observed as well on other tasks, including code generation, visual reasoning, and dangerous/sensitive question answering. This demonstrates the importance of continuously evaluating and monitoring the behaviour of privately-owned large language models when deployed in real-word applications, especially considering the lack of transparency and precise details for reproducing these models.



Simplify "The exploration doubled the estimated gold reserves at El Dorado, within the headwaters of the Lempa River, the nation's most important waterway."

The exploration found twice as much gold as expected at El Dorado, which is located at the beginning of the Lempa River, the country's most important river.

Simplify "Here , the story of Ebola is one of worldwide public health infrastructure deficiency , delicate trust in post-conflict nations , an exhausted health care workforce and widespread ambivalence"

The story of Ebola involves global health system gaps, fragile trust in post-conflict countries, a strained healthcare workforce, and widespread indifference.

Simplify "The One-Two Punch of Alcoholism: Role of Central Amygdala Dynorphins"

"Simplifying 'The One-Two Punch of Alcoholism: Role of Central Amygdala Dynorphins'"

Simplify "Development of an enhanced human gastrointestinal epithelial culture system to facilitate patient-based assays"

Simplified: "Creating a better model of the human gut lining to improve testing with patient samples."

Figure 4.6: ChatGPT generated simplifications for the complex sentences in Table 4.10.

CHAPTER 5

Judge the Judges: A Large-Scale Evaluation Study of Neural Language Models for Online Review Generation

In this chapter, we focus on the open challenge of evaluation in natural language generation. Our main contribution consists in a a large-scale, systematic experiment that evaluates the *evaluators* for NLG. We compare the preferences of different evaluators on a dozen representative deep-learning based NLG algorithms with human assessments of text quality in the context of online review generation. Our findings reveal significant differences among the evaluators and shed light on the potential factors that contribute to these differences. The analysis of a post experiment survey also provides important implications on how to guide the development of new NLG algorithms. This work is presented in:

- Gârbacea, Cristina, Samuel Carton, Shiyan Yan, and Qiaozhu Mei. *"Judge the Judges: A Large-Scale Evaluation Study of Neural Language Models for Online Review Generation."* In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. 2019.

We conduct a large-scale, systematic study to evaluate the existing evaluation methods for natural language generation in the context of generating online product reviews. We compare human-based evaluators with a variety of automated evaluation procedures, including discriminative evaluators that measure how well machine-generated text can be distinguished from human-written text, as well as word overlap metrics that assess how similar the generated text compares to human-written references. We determine to what extent these different evaluators agree on the ranking of a dozen of state-of-the-art generators for online product reviews. We find that human evaluators do not correlate well with discriminative evaluators, leaving a bigger question of whether adversarial accuracy is the correct objective for natural language generation. In general, distinguishing machine-generated text is challenging even for human evaluators, and human decisions correlate better with lexical overlaps. We find lexical diversity an intriguing metric that is indicative of the assessments of different evaluators. A post-experiment survey of participants provides insights into how to evaluate and improve the quality of natural language generation systems [1].

5.1 Introduction

Recent developments in neural language models [311], [379], [310], [309] have inspired the use of neural network based architectures for the task of natural language generation (NLG). Despite fast development of algorithms, there is an urgency to fill the huge gap in evaluating NLG systems. On one hand, a rigorous, efficient, and reproducible evaluation procedure is critical for the development of any machine learning technology and for correct interpretation of the state-of-the-art. On the other hand, evaluating the quality of language generation is inherently difficult due to the special properties of text, such as *subjectivity* and *non-compositionality*. Indeed, *"there is no agreed objective criterion for comparing the goodness of texts"* [78], and there lacks a clear model of text quality [160].

Conventionally, most NLG systems have been evaluated in a rather informal manner. [379] divide existing evaluation methods commonly employed in text generation into three categories: *i)* evaluations based on task performance, *ii)* human judgments and ratings, where human subjects are recruited to rate different dimensions of the generated texts, and *iii)* evaluations based on comparison to a reference corpus using automated metrics. *Task based evaluation* considers that the value of a piece of functional text lies in how well it serves the user to fulfill a specific application. It can be expensive, time-consuming, and often dependent on the good will of participants in the study. Besides that, it is hard to toss out the general quality of text generation from the special context (and confounds) of the task, or to generalize the evaluation conclusions across tasks. *Human annotation* is able to assess the quality of text more directly than task based evaluation. However, rigorously evaluating NLG systems with real users can be expensive and time consuming, and it does not scale well [380]. Human assessments also need to be consistent and repeatable for a meaningful evaluation [283]. Alternative strategies which are more effective in terms of cost and time are used more frequently.

Automated evaluation compares texts generated by the candidate algorithms to human-written texts. Word overlap metrics and more recent automated adversarial evaluators are widely employed in NLG as they are cheap, quick, repeatable, and do not require human subjects when a reference corpus is already available. In addition, they allow developers to make rapid changes to their systems and automatically tune parameters without human intervention. Despite the benefits,

[1] The experimental setup, data, and annotations are publicly available at: https://github.com/Crista23/JudgeTheJudges

however, the use of automated metrics in the field of NLG is controversial [379], and their results are often criticized as not meaningful for a number of reasons. First, these automatic evaluations rely on a high-quality corpus of references, which is not often available. Second, comparisons with a reference corpus do not assess the usefulness and the impact of the generated text on the readers as in human-based evaluations. Third, creating human written reference texts specifically for the purpose of evaluation could still be expensive, especially if these reference texts need to be created by skilled domain experts. Finally and most importantly, using automated evaluation metrics is sensible only if they correlate with results of human-based evaluations and if they are accurate predictors of text quality, which is never formally verified at scale.

We present a large-scale, systematic experiment that evaluates the *evaluators* for NLG. We compare three types of evaluators including human evaluators, automated adversarial evaluators trained to distinguish human-written from machine-generated product reviews, and word overlap metrics (such as BLEU and ROUGE) in a particular scenario, generating online product reviews. The preferences of different evaluators on a dozen representative deep-learning based NLG algorithms are compared with human assessments of the quality of the generated reviews. Our findings reveal significant differences among the evaluators and shed light on the potential factors that contribute to these differences. The analysis of a post experiment survey also provides important implications on how to guide the development of new NLG algorithms.

5.2 Related Work

5.2.1 Deep Learning Based NLG

Recently, a decent number of deep learning based models have been proposed for text generation. Recurrent Neural Networks (RNNs) and their variants, such as Long Short Term Memory (LSTM) [166] models, Google LM [193], and Scheduled Sampling (SS) [23] are widely used for generating textual data.

Generative Adversarial Networks [146], or GANs, train generative models through an adversarial process. Generating text with GANs is challenging due to the discrete nature of text data. SeqGAN [505] is one of the earliest GAN-based model for sequence generation, which treats the procedure as a sequential decision making process. RankGAN [266] proposes a framework that addresses the quality of a set of generated sequences collectively. Many GAN-based models [505], [266], [373], [55], [256], [519] are only capable of generating short texts. LeakGAN [154] is proposed for generating longer texts.

Deep learning architectures other than LSTM or GAN have also been proposed for text generation. [450] study NLG given particular contexts or situations and proposes two approaches based on the encoder-decoder framework. [96] address the same task and employ an additional soft attention mechanism. Pre-training enables better generalization in deep neural networks [104], especially when combined with supervised discriminative fine-tuning to learn universal robust representations [367], [89], [369]. [156] use a prototype-then-edit generative language model for sentences.

5.2.2 Automated Evaluation Metrics

The variety of NLG models are also evaluated with various approaches. Arguably, the most natural way to evaluate the quality of a generator is to involve humans as judges, either through some type of Turing test [293] to distinguish generated text from human input texts, or to directly compare the texts generated by different generators [306]. Such approaches are hard to scale and have to be redesigned whenever a new generator is included. Practically, it is critical to find automated metrics to evaluate the quality of a generator independent of human judges or an exhaustive set of competing generators.

Perplexity [189] is commonly used to evaluate the quality of a language model, which has also been employed to evaluate generators [502], [113], [133] even though it is commonly criticized for not being a direct measure of the quality of generated text [111]. Perplexity is a model dependent metric, and "how likely a sentence is generated by a given model" is not comparable across different models. Therefore we do not include perplexity in this study.

Discriminative Evaluation is an alternative way to evaluate a generator, which measures how likely its generated text can fool a classifier that aims to distinguish the generated text from human-written texts. In a way, this is an automated approximation of the Turing test, where machine judges are used to replace human judges. Discriminative machine judges can be trained either using a data set with explicit labels [336], or using a mixture of text written by real humans and those generated by the model being evaluated. The latter is usually referred to as *adversarial evaluation*. [33] proposes one of the earliest studies that uses adversarial error to assess the quality of generated sentences. Notably, maximizing the adversarial error is consistent to the objective of the generator in generative adversarial networks. [199] propose an adversarial loss to discriminate a dialogue model's output from human output. The discriminator prefers longer output and rarer language instead of the common responses generated. There however lacks evidence that a model that obtains a lower adversarial loss is better according to human evaluations.

Automatic dialogue evaluation is formulated as a learning problem in [284], who train an RNN to predict the scores a human would assign to dialogue responses. RNN predictions correlate with human judgments at the utterance and system level, however each response is evaluated in a very specific context and the system requires substantial human judgments for training. [256]

employ a discriminator (analogous to the human evaluator in the Turing test) both in training and testing and define adversarial success. Other work finds the performance of a discriminative agent (e.g., attention-based bidirectional LSTM binary classifier) is comparable with human judges at distinguishing between real and fake dialogue excerpts [38]. However, results show there is limited consensus among humans on what is considered as coherent dialogue passages.

Word Overlap Metrics, such as BLEU [345], ROUGE [265], and METEOR [16], are commonly used to measure the similarity between the generated text and human written references. [270] find that word overlap metrics present weak or no correlation with human judgments in non-task oriented dialogue systems and thus should be used with caution or in combination with user studies. In contrary, it is reported in [415] that text overlap metrics are indicative of human judgments in task-oriented dialogue settings, when used on datasets which contain multiple ground truth references. [77] find text overlap metrics too restrictive as they focus on fidelity of wording instead of fidelity of semantics. [43] consider an increase in BLEU insufficient for an actual improvement in the quality of a system and posit in favor of human evaluation.

BLEU and its variants (e.g., Self-BLEU) are used to evaluate GAN models [40, 534]. [422] compare frameworks for text generation including MLE, SeqGAN, LeakGAN and Inverse Reinforcement Learning using a simulated Turing test. A benchmarking experiment with GAN models is conducted in [285]; results show LeakGAN presents the highest BLEU scores on the test data. Similarly, BLEU and METEOR present highest correlations with human judgements [42], [147]. However, evaluation metrics are not robust across conditions, and no single metric consistently outperforms other metrics across all correlation levels [363].

Conventional neural language models trained with maximum likelihood can be on par or better than GANs [40], [405], [454]. However, log-likelihood is often computationally intractable [456]. Models with good likelihood can produce bad samples, and vice-versa [145]. Generative models should be evaluated with regards to the task they are intended for over the full quality-diversity spectrum [67], [162], [315].

While many generators are proposed and evaluated with various metrics, no existing work has systematically evaluated the different evaluators at scale, especially in the context of online review generation. Our work fills in this gap.

5.3 Experiment Design

We design a large-scale experiment to systematically analyze the procedures and metrics used for evaluating NLG models. To test the different *evaluators*, the experiment carefully chooses a particular application context and a variety of natural language generators in this context. Ideally, a sound automated evaluator should be able to distinguish good generators from suboptimal ones.

Its preferences (on ordering the generators) should be consistent to humans in the exact application context.

5.3.1 Experiment Context and Procedure

We design the experiment in the context of generating online product reviews. There are several reasons why review generation is a desirable task for the experiment: 1) online product reviews are widely available, and it is easy to collect a large number of examples for training/testing the generators; 2) Internet users are used to reading online reviews, and it is easy to recruit capable human judges to assess the quality of reviews; and 3) comparing to tasks like image caption generation or dialogue systems, review generation has minimal dependency on the conversation context or on non-textual data, which reduces possible confounds.

The general experiment procedure is presented in Figure 5.1. We start from the publicly available Amazon Product Reviews dataset [2] and select three most popular domains: *books*, *electronics*, and *movies*. After filtering rare products, inactive users, and overly long reviews, the dataset is randomly split into three parts, to train, to validate, and to test the candidate review generators (denoted as *G-train*, *G-valid*, and *G-test*). Every generative model is trained and validated using the same datasets, and then charged to generate a number of product reviews (details are included in the next section). These generated reviews, mixed with the real reviews in *G-test*, are randomly split into three new subsets for training, validating, and testing candidate (discriminative) evaluators, denoted as *D-train*, *D-valid*, and *D-test*. Finally, a random sample of reviews from *D-test* are sent for human evaluation.

5.3.2 Review Generators

Although our goal is to evaluate the evaluators, it is critical to include a wide range of text generators with various degrees of quality. A good evaluator should be able to distinguish the high-quality generators from the low-quality ones. We select a diverse set of generative models from recent literature. The goal of this study is *not* to name the best generative model, and it is unfeasible to include all existing models. Our criteria are: (1) the models are published before 2018, when our experiment is conducted; (2) the models represent different learning strategies and quality levels; (3) the models have publicly available implementations, for reproducibility purposes. In Table 5.1 we list the candidate generators. It is not an exhaustive list of what are currently available. For implementation details of these models please see Appendix 5.6.1.

Every generator (except Google LM) is trained and validated on *G-train* and *G-valid* datasets, and used to generate the same number of machine-generated (a.k.a., fake) reviews (see Table 5.2).

[2] http://jmcauley.ucsd.edu/data/amazon/

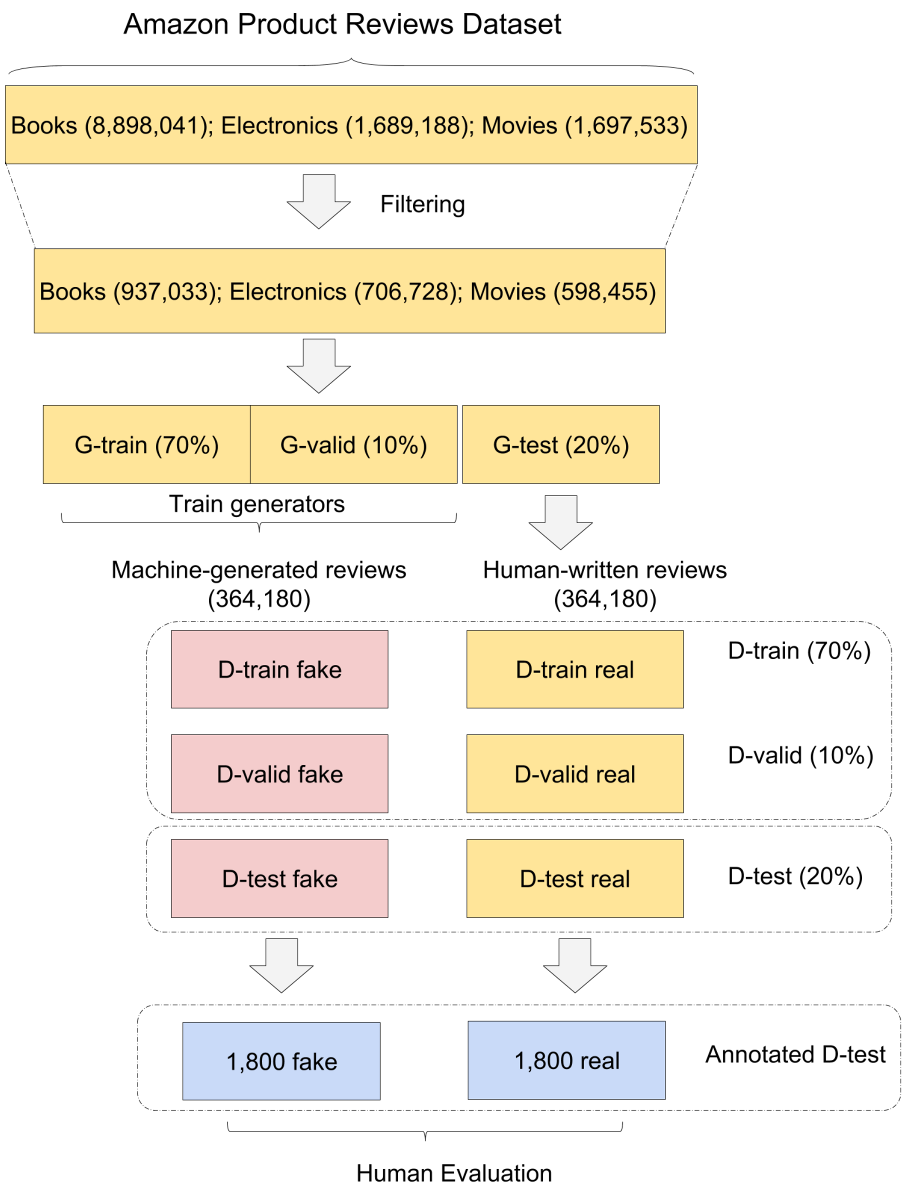


Figure 5.1: Overview of the Experiment Procedure.

We follow the best practice in literature to train these models, although it is possible that the performance of models might not be optimal due to various constraints. This will not affect the validity of the experiment as our goal is to evaluate the **evaluators** instead of the individual generators. Google LM was not trained on reviews, but it provides a sanity check for the experiment - a reasonable evaluator should not rank it higher than those trained for generating reviews.

Table 5.1: Candidate models for review generation. * indicates that review generation using these models are conditional on context information such as product ids; other models are context independent.

Generative Model	Adversarial Framework
Word LSTM temp 1.0 [166]	No
Word LSTM temp 0.7 [166]	No
Word LSTM temp 0.5 [166]	No
Scheduled Sampling [23]	No
Google LM [193]	No
Attention Attribute to Sequence* [96]	No
Contexts to Sequences* [450]	No
Gated Contexts to Sequences* [450]	No
MLE SeqGAN [505]	Yes
SeqGAN [505]	Yes
RankGAN [266]	Yes
LeakGAN [154]	Yes

Table 5.2: Number of generated reviews by each model.

Generative Model	**Total**	**D-Train**	**D-Valid**	**D-Test**
$\forall$ model in Table 5.1 except Google LM	32,500	22,750	3,250	6,500
Google LM	6,680	4,676	668	1,336

5.3.3 Evaluators

We include a comprehensive set of evaluators for the quality of the aforementioned generators: *i)* human evaluators, *ii)* discriminative evaluators, and *iii)* text overlap evaluators. The evaluators are the main subjects of the experiment.

5.3.3.1 Human evaluators

We conduct a careful power analysis [66], which suggests that at least 111 examples per generative model should be human annotated to infer that the machine-generated reviews are comparable in quality to human-written reviews, at a minimal statistically significance level of 0.05. Per this calculation, we sample 150 examples for each of the 12 generators for human evaluation. This totals 1,800 machine-generated reviews, to which we add 1,800 human-written reviews, or a total of 3,600 product reviews sent for human annotation. We markup out-of-vocabulary words in *both* human-written and machine-generated reviews to control for confounds of using certain rare words. There is no significant difference in proportion of the markup token between the two classes (2.5%-real vs. 2.2%-fake). We recruit 900 human annotators through the Amazon Mechanical Turk (AMT) platform. Each annotator is presented 20 reviews, a mixture of 10 real (i.e., human written) and 10 fake (i.e., machine generated), and they are charged to label each review as real or fake based on their own judgment. Clear instructions are presented to the workers that markup tokens are present in both classes and cannot be used to decide whether a review is real or fake. Each page is annotated by 5 distinct human evaluators. The 5 judgments on every review are used to assemble two distinct **human evaluators**: *H1* - **individual votes**, treating all human annotations independently, and *H2* - **majority votes** of the 5 human judgments. For every *annotated* review, the human evaluator ($H1$ or $H2$) makes a call which can be either right or wrong with regard to the ground truth. A generator is high quality if the human evaluator achieves low accuracy identifying the reviews as fake.

5.3.3.2 Discriminative evaluators

The inclusion of multiple generators provides the opportunity of creating **meta-adversarial evaluators**, trained using a *pool* of generated reviews by *many* generators, mixed with a larger number of "real" reviews (*D-train* and *D-valid* datasets). Such a "pooling" strategy is similar to the standard practice used by the TREC conferences to evaluate different information retrieval systems [161]. Comparing to individual adversarial evaluators, a meta-evaluator is supposed to be more robust and fair, and it can be applied to evaluate new generators without being retrained. In our experiment, we find that the meta-adversarial evaluators rank the generators in similar orders to the best individual adversarial evaluators.

We employ a total of 7 meta-adversarial evaluators: 3 deep, among which one using LSTM [166], one using Convolutional Neural Network (CNN) [240], and one using a combination of LSTM and CNN architectures; 4 shallow, based on Naive Bayes (NB) [385], Random Forest (RF) [262], Support Vector Machines (SVM) [74], and XGBoost [60], with unigrams, bigrams, and trigrams as features and on balanced training sets. We find the best hyper-parameters using random search and prevent the models from overfitting by using early stopping. For every review in *D-test* (either annotated or not), a meta-adversarial evaluator makes a judgment call. A generator is considered high quality if the meta-adversarial evaluator makes more mistakes on reviews it generated.

5.3.3.3 Word-overlap evaluators

We include a set of 4 text-overlap metrics used for NLG evaluation: BLEU and METEOR (specific to machine translation), ROUGE (used in text summarization), and CIDEr [464] (used in image description evaluation). These metrics rely on matching n-grams in the target text (i.e., generated reviews) to the "references" (i.e., human-written reviews). The higher the overlap (similarity), the higher the quality of generated text. For every generated review in *D-test Fake*, we assemble the set of references by retrieving the top-10 most similar human-written reviews in *D-test Real* using a simple vector space model. We compute 600-dimensional vector representation of reviews using Sent2Vec [341], pretrained on English Wikipedia, and retrieve the top-k nearest neighbors for each review based on cosine similarity of the embedding vectors. The rationale of using nearest neighbors of each generated review as references is that to appear "real", a generated review just need to be similar to *some* real reviews instead of *all*. A generator is considered high quality if its generated reviews obtain a high average score by a text overlap evaluator. In total, we analyze and compare 13 candidate evaluators (2 human evaluators, 7 discriminative evaluators, and 4 text-overlap metrics), based on the *D-test* dataset.

5.4 Results

First, we are interested in the accuracy of individual evaluators - how well they can distinguish "fake" (machine-generated) reviews from "real" (human-written) reviews. Second, we are interested in how an evaluator assesses the quality of the 12 generators instead of individual reviews. The absolute scores an evaluator gives to the generators are not as informative as how it ranks them: a good evaluator should be able to rank good generators above bad generators. Last but not least, we are interested in how the rankings by different evaluators correlate with each other. Intuitively, an automated evaluator that ranks the generators in similar orders as the human evaluators is more reasonable and can potentially be used as the surrogate of humans.

5.4.1 Results of Individual Evaluators

5.4.1.1 Human evaluators

Every review is annotated by 5 human judges as either "fake" or "real." The inter-annotator agreement (Fleiss-Kappa score [117]) is $k = 0.2748$. This suggests that *distinguishing machine-generated reviews from real in general is a hard task even for humans*; there is limited consensus on what counts as a realistic review. The low agreement also implies that any automated evaluator that mimics human judges is not necessarily the most "accurate."

In Figure 5.2 we present the accuracy of two human evaluators on individual annotated reviews, based on either all 5 annotations or their majority votes for each review. Comparing to the ground-truth (of whether a review is machine-generated or collected from Amazon), individual human decisions are 66.61% accurate, while their majority votes can reach 72.63%. Neither of them is close to perfect. *We observe that human evaluators generally do better at correctly labelling*

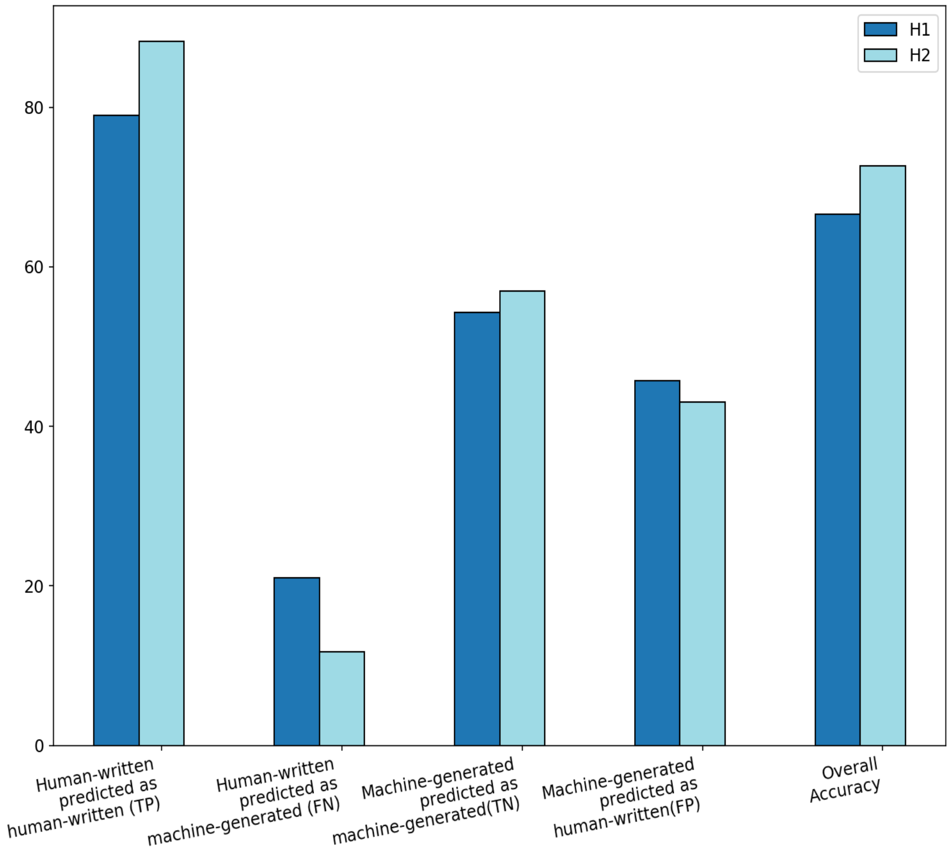


Figure 5.2: Accuracy of human evaluators on individual reviews: *H1* - individual votes; *H2* - majority votes.

human-written reviews as real (true positive rate of 78.96% for $H1$ *and 88.31% for* $H2$*), and they are confused by machine-generated reviews in close to half of the cases (true negative rate of 54.26% for* $H1$ *and 56.95% for* $H2$*).* This trend reassures previous observations [450].

We then look at how the human evaluators rank the 12 generators, according to the accuracy of human evaluators on all (fake) reviews generated by each of the generators. The lower the accuracy, the more likely the human evaluator is confused by the generated reviews, and thus the better the generator. We observe a substantial variance in the accuracy of both human evaluators on different generators, which suggests that human evaluators are able to distinguish between generators. The generator ranked the highest by both human evaluators is *Gated Contexts to Sequences*. Google LM is ranked on the lower side, which makes sense as the model is not trained to generate reviews. Interestingly, humans tend not to be fooled by reviews generated by the GAN-based models (MLE SeqGAN, SeqGAN, RankGAN and LeakGAN), even though their objective is to confuse fake from real. GAN-generated reviews tend to be easily distinguishable from the real reviews by human judges.

5.4.1.2 Discriminative evaluators

We then analyze the 7 meta-adversarial evaluators. Different from human evaluators that are applied to the 3,600 annotated reviews, the discriminative evaluators are applied to *all* reviews in *D-test*.

Meta-adversarial Evaluators. On individual reviews, the three deep learning based and the one SVM based evaluators achieve higher accuracy than the two human evaluators, indicating that adversarial evaluators can distinguish a single machine-generated review from human-written better than humans (Figure 5.3 and Table 5.4 in Appendix 5.6.3.2). Their true positive rates and true negative rates are more balanced than human evaluators. Meta-discriminators commonly rank GAN-based generators the highest. This makes sense as the objective of GAN is consistent to the (reversed) evaluator accuracy. Interestingly, by simply setting the temperature parameter of Word LSTM to 1.0, it achieves comparable performance to the GANs.

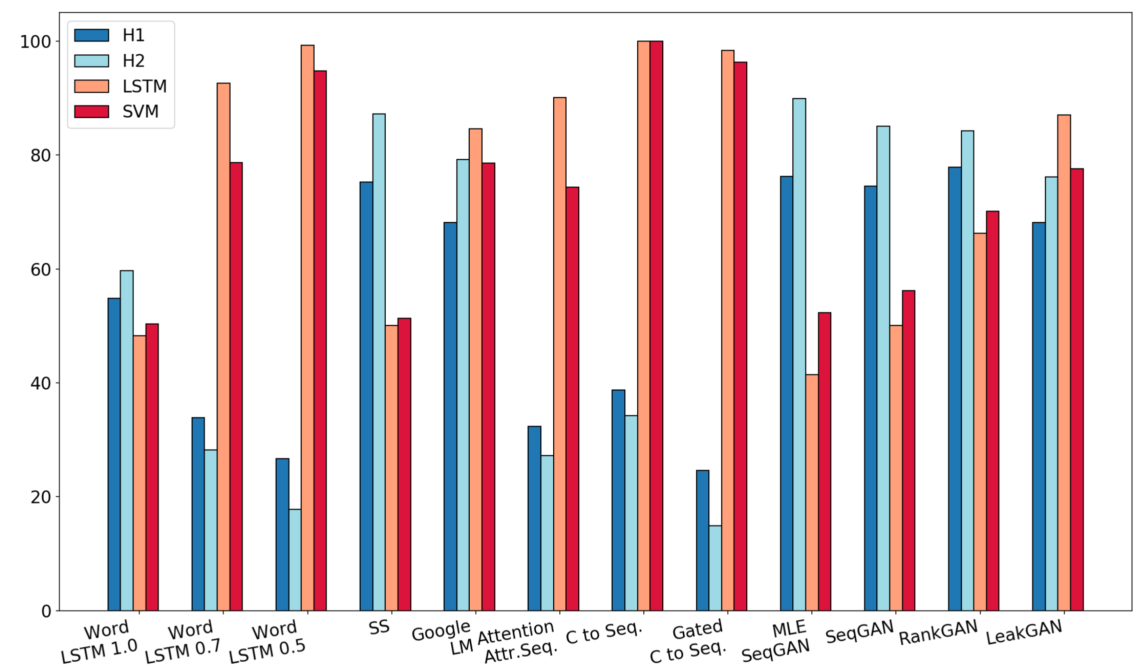


Figure 5.3: Accuracy of human (H1, H2) and meta-adversarial evaluators (LSTM, SVM) on reviews generated by individual generators. **The lower the accuracy, the better the generator.**

5.4.1.3 Word-Overlap Evaluators

The generators are ranked based on the average scores of their generated reviews. In Figure 5.4 we present the average scores of the 12 generators by each evaluator. Different word-overlap evaluators also tend to rank the generators in similar orders. Interestingly, the top-ranked generator according to three evaluators is *Contexts to Sequences*, while CIDEr scores highest the *Gated Contexts to Sequences* model. GAN-based generators are generally ranked low; please also see Appendix 5.6.3.3.

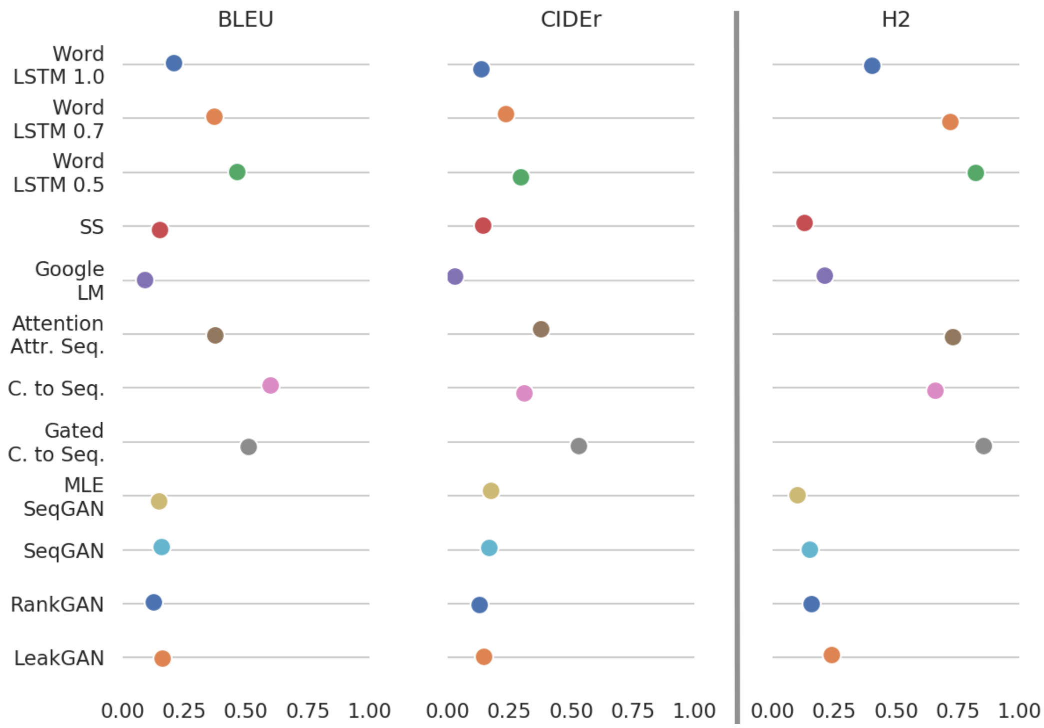


Figure 5.4: Text-Overlap Evaluators (BLEU and CIDEr) scores for individual generators. **The higher the better.** The rankings are overall similar, as GAN-based generators are ranked low.

5.4.2 Comparing Evaluators

To what degree do the evaluators agree on the ranking of generators? We are more interested in how the automated evaluators compare to the human evaluators, and whether there is any suitable automated surrogate for human judges at all. To do this, we compute the correlations between $H1$, $H2$ and each discriminative evaluator and correlations between $H1$, $H2$ and the text-overlap evaluators, based on either their decisions on individual reviews, their scores of the generators (by Pearson's coefficient [114]), and their rankings of the generators (by Spearman's ρ [433] and Kendall's τ [81]). Patterns of the three correlation metrics are similar; please see Figure 5.5 and

Table 5.5 in Appendix 5.6.3.4.

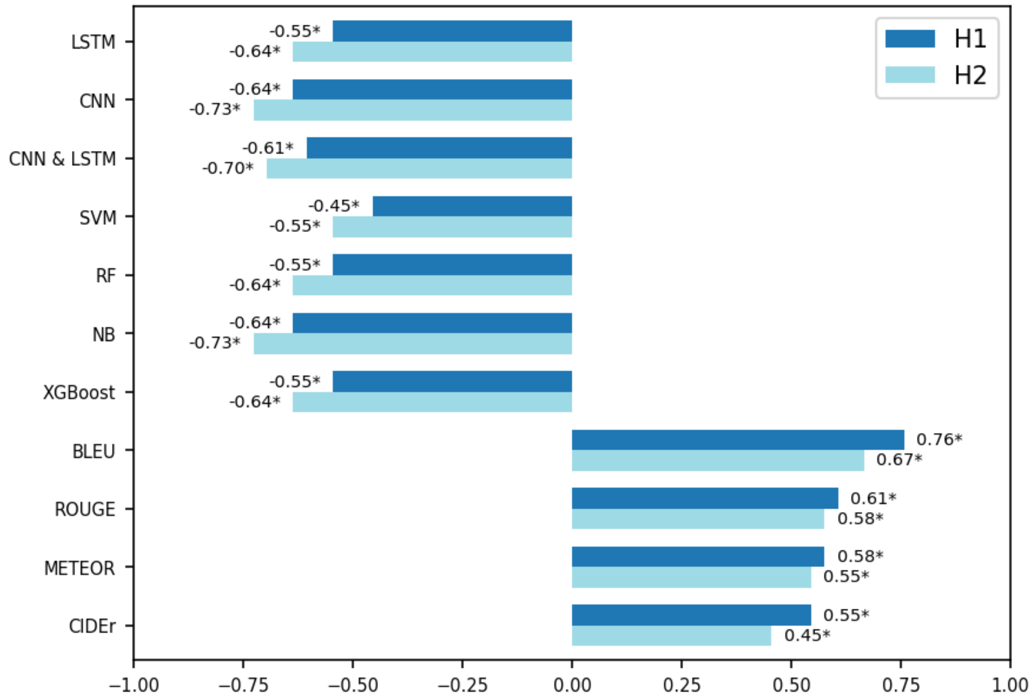


Figure 5.5: Kendall τ-b between human and automated evaluators. Human's ranking is positively correlated to text-overlap evaluators and negatively correlated to adversarial evaluators (* is $p \leq 0.05$).

Surprisingly, none of the discriminative evaluators have a positive correlation with the human evaluators. That says, *generators that fool machine judges easily are less likely to confuse human judges, and vice versa*. *Word-overlap evaluators tend to have a positive correlation with the human evaluators in ranking the generators*. Among them, BLEU appears to be closer to humans. This pattern is consistent in all three types of correlations. These two observations are intriguing, which indicates that when identifying fake reviews, humans might focus more on word usage rather than trying to construct a "decision boundary" mentally.

In summary, we find that 1) human evaluators cannot distinguish machine-generated reviews from real reviews perfectly, with significant bias between the two classes; 2) meta-adversarial evaluators can better distinguish individual fake reviews, but their rankings at the generator level tend to be negatively correlated with human evaluators; and 3) text-overlap evaluators are highly correlated with human evaluators in ranking generators.

5.5 Discussion

We carried a systematic experiment that evaluates the evaluators for NLG. Results have intriguing implications to both the evaluation and the construction of natural language generators. We conduct in-depth analysis to discover possible explanations.

5.5.1 Granularity of Judgments

We charged the Turkers to label individual reviews as either fake or real instead of evaluating each generator as a whole. Comparing to an adversarial discriminator, a human judge has not seen many "training" examples of *fake* reviews or generators. That explains why the meta-adversarial evaluators are better at identifying fake reviews. In this context, humans are likely to judge whether a review is real based on how "similar" it appears to the true reviews they are used to seeing online.

This finding provides interesting implications to the selection of evaluation methods for different tasks. In tasks that are set up to judge individual pieces of generated text (e.g., reviews, translations, summaries, captions, fake news) where there exists human-written ground-truth, it is better to use word-overlap metrics instead of adversarial evaluators. When judgments are made on the agent/system level (e.g., whether a Twitter account is a bot), signals like how similar the agent outputs are or how much the agent memorizes the training examples may become more useful than word usage, and a discriminative evaluator may be more effective than word-overlap metrics. Our finding also implies that adversarial accuracy might not be the optimal objective for NLG if the goal is to generate documents that humans consider as real. Indeed, a fake review that fools humans does not necessarily need to fool a machine that has seen everything. In Appendix 5.7.2 we provide more details.

5.5.2 Imperfect Ground Truth

One important thing to note is that all discriminative evaluators are trained using natural labels (i.e., treating all examples from the Amazon review dataset as positive and examples generated by the candidate models as negative) instead of human-annotated labels. Some reviews posted on Amazon may have been generated by bots, and if that is the case, treating them as human-written examples may bias the discriminators. To verify this, we apply the already trained meta-discriminators to the human-annotated subset (3,600 reviews) instead of the full *D-test* set, and we use the majority vote of human judges (whether a review is fake or real) to surrogate the natural "ground-truth" labels (whether a review is generated or sampled from Amazon).

When the meta-adversarial evaluators are tested using human majority-votes as ground-truth, the scores and rankings of these discriminative evaluators are more inline with the human evalua-

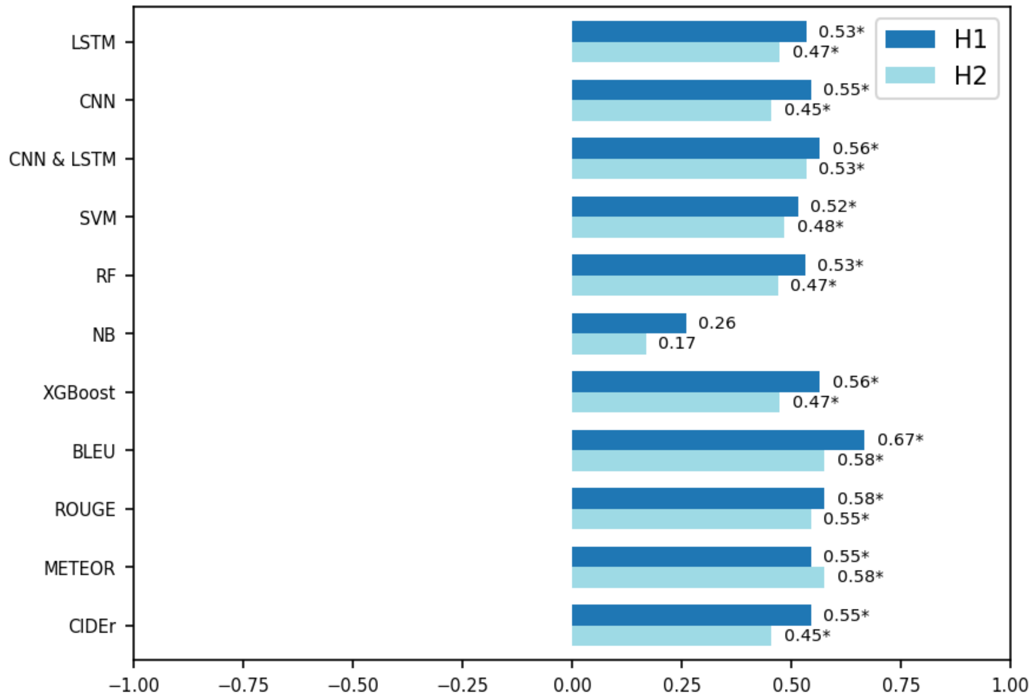


Figure 5.6: Kendall τ-b correlation coefficients between human evaluators and automated evaluators, tested on the **annotated subset of *D-test*** with *majority votes* as ground-truth (* denotes $p \leq 0.05$).

tors, although still not as highly correlated as BLEU; please see Figure 5.6. Indeed, discriminative evaluators suffer the most from low-quality labels, as they were directly trained using the imperfect ground-truth. Word-overlap evaluators are more robust, as they only take the most relevant parts of the test set as references (more likely to be high quality). Our results also suggest that when adversarial training is used, selection of training examples must be done with caution. If the "ground-truth" is hijacked by low quality or "fake" examples, models trained by GAN may be significantly biased. This finding is related to the recent literature of the robustness and security of machine learning models [344]. Appendix 5.7.3 contains further details.

5.5.3 Role of Diversity

We assess the role diversity plays in rankings the generators. Diversity of a generator is measured by either the lexical diversity [9] or Self-BLEU [534] of the samples produced by the generator. Results obtained (see Figure 5.7) indicate generators that produce the least diverse samples are easily distinguished by the meta-discriminators, while they confuse humans the most. This confirms that adversarial discriminators capture limitations of generative models in lack of diversity [199].

Similarly, we measure to what extent the generators are memorizing the training set *G-train*

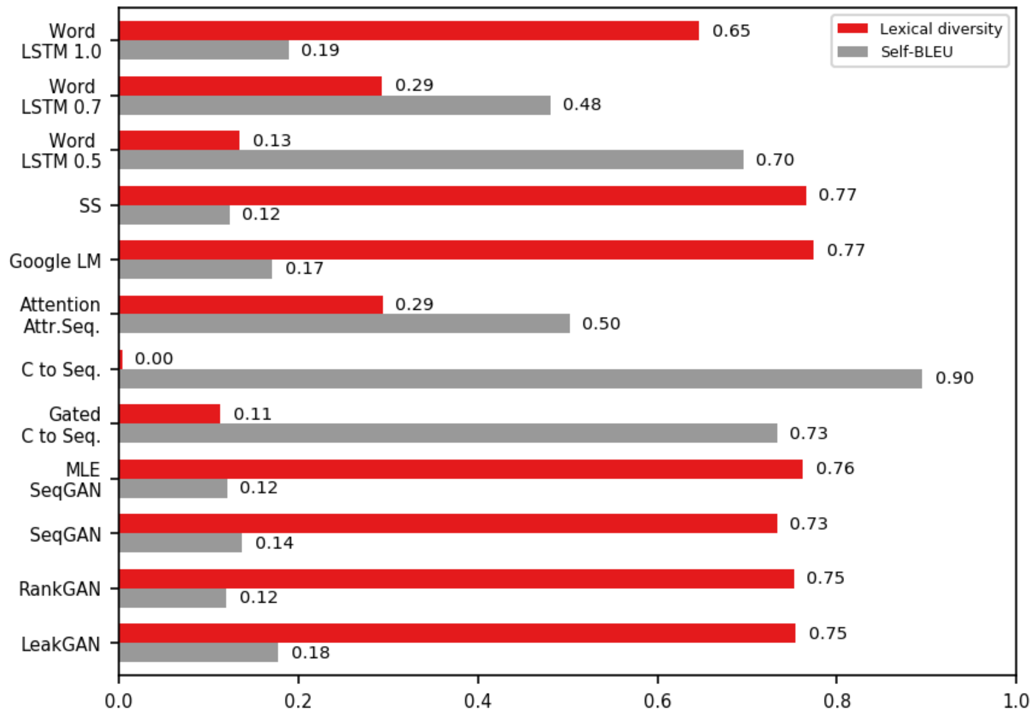


Figure 5.7: Self-BLEU scores (the lower the more diverse) and lexical diversity scores (the higher the more diverse) are highly correlated in ranking the generators.

as the average BLEU scores of generated reviews using their nearest neighbors in *G-train* as references. We observe the generators do not memorize the training set, and GAN models generate reviews that have fewer overlap with *G-train*; this finding is in line with recent theoretical studies on memorization in GANs [320].

The effects of diversity indicate that when humans are distinguishing individual reviews as real or fake, whether or not a fake review is sufficiently different from the other generated reviews is not a major factor for their decision. Instead, they tend to focus on whether the review looks similar to the reviews they have seen in reality. A discriminative evaluator is more powerful in making decisions at a system level (e.g., a dialog system or a bot account), where diversity becomes a major factor. In Appendix 5.7.4 we provide more details.

5.5.4 User Study

Finally, we are interested in the reasons why human annotators label certain reviews as fake (machine-written). After annotating a batch of reviews, workers are asked to explain their decisions by filling in an optional free-text comment. This enables us to have a better understanding of what differentiates machine-generated from human-written reviews from human's perspective.

Analyzing their comments, we identify the main reasons why human evaluators annotate a review as machine-written. These are mainly related to the presence of grammatical errors in the review text, wrong wording or inappropriate choice of expressions, redundant use of specific phrases or contradictory arguments in the review. Interestingly, human evaluators' innate biases are also reflected in their decisions: they are likely to categorize a review as fake if it is too formal, lacks emotion and personal pronouns, or is too vague and generic. Please see Appendix 5.7.1.

5.5.5 Summary

In summary, our findings represent a preliminary foundation for proposing more solid and robust evaluation metrics and objectives of natural language generation. The low inter-rater agreement we observe suggests that judging *individual* pieces of text as machine- or human-generated is a difficult task even for humans. In this context, discriminative evaluators are not as correlated with human judges as word-overlap evaluators. That implies that adversarial accuracy might not be the optimal objective for generating individual documents when realism is the main concern. In contrast, GAN based models may more easily pass a Turing test on a *system* level, or in a conversational context. When the judges have seen enough examples from the same generator, the next example had better be somewhat different.

Our results also suggest that when adversarial evaluation is used, the training examples must be carefully selected to avoid false-positives. We also find that when humans are distinguishing fake reviews from real ones, they tend to focus more on the usage of words, expressions, emotions, and other details. This may affect the design of objectives for the next generation of NLG models. In future work, we would like to determine to what extent it is possible to predict human judgements of text quality based on scores of automated evaluation metrics.

5.5.6 Future Outlook

Given the latest advancements in large language models (LLMs), it is becoming increasingly difficult for humans to distinguish machine-generated from human-written texts. The emergence of models such as ChatGPT [333] or GPT-4 [334] further complicates the problem of chatbot detection, as these models can generate high-quality texts that mimick human behaviour to a certain extent [472]. In line with our findings, other works in the literature report humans perform only slightly better than chance when classifying real from fake texts [131], [508]. Unfortunately, this may lead to the misuse of text generation models in many real-world application contexts with negative consequences for our society (for eg., convincing but inaccurate news articles, election manipulation campaigns, fake news, dishonesty in educational assignments, impersonating human users, eroding trust in online interactions, etc.). To this end, giving readers confidence in the origin of the texts they consume and minimizing the threats posed by adversaries becomes extremely important. For establishing the authenticity of LLM-generated texts, engaging in real-time conversations via sets of questions designed to explore differences between how bots and humans process and generate language are proposed [472]. Leaving from the observation that LLMs have difficulty with skills such as symbolic manipulation, noise filtering or graphical understanding, questions are divided into two categories: *i)* easy for humans but difficult for bots (counting the number of occurrences of a character in a string, substitution of characters in a string given a specific rule, identification of a specific character in a string, noise filtering when a string is modified with extra characters), and *ii)* easy for bots but difficult for humans (memorization, computation, domain-specific questions not encountered in everyday life). If the overlap between the given answer and the ground-truth is higher than a pre-set threshold, the likelihood of the answer being machine-generated increases. DetectGPT [314] proposes zero-shot machine-generated text detection by assessing the average per-token log probability under the original model of a piece of text before and after perturbing it. Unlike human-written text, model-generated text tends to lie in areas of negative curvature of the log probability function. Machine-generated text detection can also be framed as a binary classification problem [188], [13], [431], [106], however the detection model may overfit to the training topics and needs retraining whenever a new text generation models is released. Watermarking language model output consists in adding human imperceptible signals to the generated texts according to specific hashing rules, but which are statistically detectable to machine algorithms as synthetic data [217]. Interestingly, the best deep pre-trained language models for generating misinformation are found to also be the best at detecting disinformation when used in a discriminative setting [508]. As large language models are constantly improving and becoming more versatile, robust and accurate methods that can differentiate machine-generated from human-written texts are critically needed.

5.6 Appendix A2

5.6.1 Implementation Details for Review Generators

Recurrent Neural Networks (RNNs) directly model the generation process of text sequences, and provide an end-to-end solution to learning the generating function from large quantities of data. These networks maintain a hidden layer of neurons with recurrent connections to their own previous values, which in theory gives them the potential to model long span dependencies. For an input sequence $x = x_1, x_2, \dots, x_t$, the hidden state h_t which summarizes the information of the entire sequence up to timestep t is recursively updated as $h_t = f(h_{t-1}, x_t)$, where $f(.,.)$ denotes a non-linear transformation function. The overall probability of the sequence is calculated as:

$$p(x) = \prod_{t=1}^{T} p(x_t|h_{t-1}), \tag{5.1}$$

and the probability of generating the next word x_{t+1} given its low dimensional continuous representation $O_{x_{t+1}}$ and input sequence x_t is defined as:

$$p(x_{t+1}|x \leq t) = p(x_{t+1}|h_t) \propto \exp(O_{x_{t+1}}^T h_t) \tag{5.2}$$

However, in practice the gradient computation is difficult to propagate back in time due to exploding or vanishing gradients [165], [26], making the learning of arbitrarily long phenomena challenging in RNNs. Long Short Term Memory networks (LSTMs) [166] effectively address these limitations by relying on a memory state and gating functions to control the flow of the information throughout the network – and in particular what information is written to the memory state, what information is read from the memory state, and what information is removed (or forgotten) from the memory state. The mathematical formulation of LSTM units can be expressed as follows:

$$\begin{aligned}
i^{(t)} &= \sigma(W^{(i)}x^{(t)} + U^{(i)}h^{(t-1)}) && \text{(Input gate)} \\
f^{(t)} &= \sigma(W^{(f)}x^{(t)} + U^{(f)}h^{(t-1)}) && \text{(Forget gate)} \\
o^{(t)} &= \sigma(W^{(o)}x^{(t)} + U^{(o)}h^{(t-1)}) && \text{(Output gate)} \\
\widetilde{c}^{(t)} &= \tanh(W^{(c)}x^{(t)} + U^{(c)}h^{(t-1)} && \text{(New memory cell)} \\
c^{(t)} &= f^{(t)} \circ \widetilde{c}^{(t-1)} + i^{(t)} \circ \widetilde{c}^{(t)} && \text{(Final memory cell)} \\
h^{(t)} &= o^{(t)} \circ \tanh(c^{(t)})
\end{aligned} \tag{5.3}$$

In the above set of equations, the input word $x^{(t)}$ and the past hidden state $h^{(t-1)}$ are used to generate new memory $\widetilde{c}^{(t)}$ which includes features of the new word $x^{(t)}$ without prior determination of whether $x^{(t)}$ is important and worth keeping. The role of the input gate is to check whether it is sensible to store the new input word given the word $x^{(t)}$ itself and the past hidden state $h^{(t-1)}$; the input gate produces $i^{(t)}$ as output, which encapsulates the worthiness decision of preserving the input information. Similarly to the input gate, the forget gate also determines the usefulness of a word by inferring whether the past memory cell is used to compute the current memory cell by looking at the input word word $x^{(t)}$ itself and the past hidden state $h^{(t-1)}$; it produces $f^{(t)}$ as output, which encapsulates the worthiness decision of preserving the past memory cell. In the final memory generation stage, the advice of the input gate $i^{(t)}$ to gate the new memory $\widetilde{c}^{(t)}$ and the advice of the forget gate $f^{(t)}$ to forget the past memory $\widetilde{c}^{(t-1)}$ are both considered, and the two results are summed up to produce the final memory $c^{(t)}$. The output gate is used to separate the hidden state h^t from the final memory of the network $c^{(t)}$. Given that every state of the LSTM is relying on hidden states and that the final memory $c^{(t)}$ contains a lot of information not necessarily required to be saved in the hidden state, the output gate discriminatively assesses which parts of the memory $c^{(t)}$ should be kept inside the hidden state h^t. In our experiments we employ an LSTM generative model trained at word level. Sampling from a trained word language model can be done in two ways: beam search [11] and random sampling [149]. Following [450], we use random sampling with different values for the temperature parameter. Sampling from the LSTM model with a high temperature results in the model generating diverse samples at the cost of introducing some mistakes, while small temperatures generate conservative samples without a lot of content diversity. In our experiments, we empirically set the temperatures to the following values: 1.0, 0.7 and 0.5.

RNNs, and LSTMs in particular, have become the standard for modeling machine learning problems that involve temporal and sequential data including text. The data is modeled via a fully-observed directed graphical model, where the distribution over a discrete time sequence $y_1, y_2, \ldots, y_T$ is decomposed into an ordered product of conditional distributions over tokens:

$$P(y_1, y_2, \ldots, y_T) = P(y_1) \prod_{t=1}^{T} P(y_t | y_1, \ldots, y_{t-1}) \tag{5.4}$$

For models with recurrent connections from their outputs leading back into the model, *teacher forcing* [482] is the most popular training strategy. This procedure emerges from the maximum likelihood criterion, in which at training time $t+1$ the model receives as input the ground truth output y^t:

$$\begin{aligned} \log p(y^{(1)}, y^{(2)} | x^{(1)}, x^{(2)}) = \log p(y^{(2)} | y^{(1)}, x^{(1)}, x^{(2)}) \\ + \log p(y^{(1)} | x^{(1)}, x^{(2)}) \end{aligned} \tag{5.5}$$

The model in Equation 5.5 above illustrates the conditional maximum likelihood criterion at timestep $t = 2$. The model is trained to maximize the conditional probability of $y^{(2)}$ given the sequence x generated so far and the previous $y^{(1)}$ value. Therefore, maximum likelihood specifies that at training time the previous token generated by the model is replaced with ground-truth examples y_t that are fed back into the model for predicting outputs at later time steps. Feeding back ground truth samples at training time forces the RNN to stay close to the ground-truth sequence. However, at inference time, the ground truth sequence is no longer available conditioning, and each y_t is generated by the model itself (i.e. sampled from its conditional distribution over the sequence given the previously generated samples). This discrepancy between training time and inference time causes errors in the model predictions that accumulate and amplify quickly over the generated sequence as the model is in a part of the state space it has never seen during training time. Small prediction errors compound in the RNN's conditioning context, and as the generated sample starts to diverge from sequences it has seen during training, the prediction performance of the RNN worsens [233].

To alleviate this problem, Bengio et al [23] propose *Scheduled Sampling (SS)*, a learning strategy for training RNNs which mixes inputs from the ground-truth sequence with inputs generated by the model itself at training time. SS relies on curriculum learning [25] to change the training process from a fully guided scheme using the true previous token to a less guided scheme mostly using the generated token. The choice of replacing the ground truth with the model's prediction is determined by a coin flip with some probability, independently for each token. The probability of using the ground truth is set to a high value initially. As the model gradually keeps improving, samples from the model become more frequent and the model is partially fed with its own synthetic data as prefix in a similar way to inference mode. Therefore, the training objective is slowly changed from an easy task where the previous token is known, to a realistic task where the previous token is provided by the model itself. The scheduled sampling training scheme is meant to make the model more robust and forces it to deal with its own mistakes at training time, in a similar way to inference time. However, as the model generates several consecutive tokens y_t-s, it is not clear whether the correct target distribution remains the same as in the ground truth sequence. The authors propose two solutions: *i)* make the self-generated sequences short, and *ii)* anneal the probability of using self-generated vs. ground-truth samples to 0, according to some schedule.

Despite its impressive empirical performance, Huszar et al [180] show that SS is an inconsistent training strategy which pushes models towards memorising the distribution of symbols conditioned on their position in the sequence instead of on the prefix of preceding symbols. According to the authors, SS pays no attention to the content of the sequence prefix, and uses the hidden states to implement a simple counter which makes the model likely to recover from its own mistakes. Moreover, it is possible that the good performance of the model on image captioning datasets is either due to the algorithm not running until convergence, or to a lucky combination of factors including the model structure, early stopping, random restarts, and the annealing schedule. The authors recommend adversarial training strategies as a much better choice for generative models.

Tang et al [450] study the the problem of NLG at particular contexts or situations. The authors focus on user review data due to its richness of context, sentiments and opinions expressed. They propose two approaches built on top of the encoder-decoder framework to generate user reviews as text sequences from user product contexts. In the first approach, *Contexts to Sequences*, the authors encode the product context information $\overrightarrow{C} = \{\overrightarrow{c}_i\}_{i=1,\ldots,K}$, where $\overrightarrow{c}_i$ denotes a type of context and K the number of context types, into a continuous semantic representation, which is fed into an LSTM decoder to generate text sequences. Despite promising results shown by the method, the authors consider that for long generated sequences the information from contexts is not propagated to distant words. In their second approach, *Gated Contexts to Sequences*, the authors add skip-connections to directly build the dependency between contexts h_C and each word when predicting the next word x_{t+1} in a sequence. When a new word in a sequence is generated, it does not only depend on the current hidden state h_t, but it also depends on the context representation h_C. Similar to the first model, the decoder is a vanilla recurrent neural network with LSTM unit.

Focusing on the same problem as Tang et al [450], Dong et al [96] propose *Attention Enhanced Attribute to Sequence Model*. The model learns to encode product attributes into vectors by means of an encoder network, and then generate reviews by conditioning on the encoded vectors inside a sequence decoder, and an attention mechanism [11], [495] which learns soft alignments between the input attributes and the generated words. The product review generation problem is formally defined as follows. Given input attributes $a = (a_1, \ldots, a_{|a|})$, generate a product review $r = (y_1, \ldots, y_{|r|})$ which maximizes the conditional probability $p(r|a)$:

$$p(r|a) = \prod_{t=1}^{|r|} p(y_t|(y_1, \ldots, y_{t-1}), a) \tag{5.6}$$

While the number of attributes $|a|$ is fixed for each product, the review text r is a sequence of variable length. In our experiments we use the two models proposed by Tang et al [450] and Dong et al [96] to generate use product reviews given the context information and the review text of each product in the Amazon dataset.

In addition to the already mentioned models, we also employ a pre-trained model released by Google, commonly referred to as Google LM [193]. The model is an important contribution to the field of neural language modeling which emphasizes large scale recurrent neural network training. The model was trained on the One Billion Word Benchmark [56], a publicly available dataset containing mainly news data and used as a reference standard for measuring the progress of statistical language modeling. The dataset includes 1 billion words in total with a vocabulary of 800,000 unique words. While for count based language models it is considered a medium-sized dataset, for neural network based language models the benchmark is regarded as a very large dataset. In terms of the model architecture, the GoogleLM model is a 2-layer LSTM neural network with 8,192 and respectively 1,024 hidden units in each layer, the largest Google was able to fit into GPU memory. The model uses Convolutional Neural Networks (CNNs) character embeddings as input, and makes predictions one character at a time, which presents the advantage that the model does not need to learn long-term dependencies in the data. We employ GoogleLM to generate sentences with a topic which identifies with the existing three categories (books, electronics and movies) present in the Amazon dataset we used.

Generative Adversarial Networks (GANs) [146] represent a training methodology for generative models via an adversarial process, and are aimed at generating synthetic data which resembles the real data. The GAN framework works through the interplay between two feedforward neural network models, a generative model G and a discriminative model D, trained simultaneously by competing against each other. The generative model G aims to capture the data distribution and generate high quality synthetic data, while the discriminative model D estimates the probability a sample comes from the real training data and not from the synthetic data generated by G. Concretely, the generator G takes as input a vector of random numbers z, and transforms it into the form of the data we are interested in imitating; the discriminator D takes as input either the real data x or generated data $G(z)$, and outputs probability $P(x)$ of the respective data being real. The GAN framework is equivalent to a minimax two-player game between the two models G and D:

$$\begin{aligned}\min_G \max_D V(D, G) = {} & \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] \\ & + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]\end{aligned} \tag{5.7}$$

Adversarial learning algorithms iteratively sample batches from the data and noise distributions, and use noisy gradient information to simulatenously ascend in the parameters θ_d of D, while descending in the parameters θ_g of G. The discriminator D is optimized to increase the likelihood of assigning a high probability to the real data x and a low probability to the fake generated data $G(z)$. The gradient for the discriminator can be expressed as follows:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \big[\log D(x^{(i)}) + \log(1 - D(G(z^{(i)}))) \big] \tag{5.8}$$

Alternatively, the generator G is optimized to increase the probability the generated data $G(z)$ is rated highly:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \big[\log(1 - D(G(z^{(i)}))) \big] \tag{5.9}$$

The goal of the generator G is to maximize the probability of discriminator D making a mistake by generating highly realistic data, while the discriminator D is learnt to distinguish whether a given data instance is real or not. The gradient of the training loss from the discriminator D is used as guidance for updating the parameters of the generator G. Gradient optimization is alternated between the two networks D and G as illustrated in Equations 5.8 and 5.9 on batches of real and generated data until GAN converges, at which point the data produced by GAN is the most realistic the network is capable of modeling.

However, GAN's applicability to discrete data is limited, despite the great success at generating realistic real valued synthetic samples in many computer vision tasks for eg., image generation [35], [532], [447], image style transfer [288], [533] and semantic segmentation [289], [432]. Training generative models of text using GANs is challenging due to the discrete nature of text data, which makes it difficult to backpropagate the gradient from the discriminator D to the generator G. GANs are designed for generating real-valued, continuous data, and the gradient of the loss from discriminator D w.r.t. the output of generator G is used to guide G to slightly change the generated value to make it more realistic (i.e. the gradient of the output of the discriminator network with respect to the synthetic data indicates how to slightly change the synthetic data to make it more plausible). Changes can be made to the synthetic data if it is based on real numbers, however for discrete tokens the slight change guidance is not a useful signal, as it is very likely that there is no corresponding token to the slight change given the limited vocabulary space[3]. In addition, a further reason why GANs cannot be applied to text data is because the discriminator D can only asses a complete sequence. When having to provide feedback for partially generated sequences, it is non-trivial to balance the current score of the partially generated sequence with the future score after the entire sequence has been generated [505]. In the literature there are two approaches on how to deal with the problem of non-differentiable output and finding the optimal weights in a neural network: the REINFORCE algorithm, and Gumbel-Softmax reparameterization. We present each method below.

REINFORCE [481] algorithms, also known as *REward Increments, score-function estimators*, or *likelihood-ratio methods* adjust the weights of a neural network based on the log derivative trick in a direction that lies along the gradient of expected reinforcement without explicitly computing gradient estimates. It is a policy gradient method which uses the likelihood ratio trick $\left(\frac{\nabla_\theta p(X,\theta)}{P(X,\theta)} = \nabla_\theta \log p(X,\theta); \frac{\partial}{\partial_x} \log f(x) = \frac{f'(x)}{f(x)}\right)$ to update the parameters of an agent and increase the probability that the agent's policy will select a rewarding action given a state. Given the trajectory $\tau_t = (u_1, \ldots, u_{t-1}, x_0, \ldots, x_t)$ made up of a sequence of states x_k and control actions u_k, the goal of policy gradient is to find policy π_ϑ which takes as input trajectory τ_t and outputs a new control action that maximizes the total reward after L time steps. π_ϑ is a parametric randomized policy which assumes a probability distribution over actions:

$$p(\tau;\vartheta) = \prod_{t=0}^{L-1} p(x_{t+1}|x_t, u_t)\pi_v(u_t|\tau_t) \tag{5.10}$$

If we define the reward of a trajectory as:

[3] https://www.reddit.com/r/MachineLearning/comments/40ldq6/generative_adversarial_networks_for_text/

$$R(\tau) = \sum_{t=0}^{N} R_t(x_t, u_t), \tag{5.11}$$

the reinforcement learning optimization problem becomes:

$$\max_{\vartheta} J(\vartheta) = \max_{\vartheta} \mathbb{E}_{p(\tau|\vartheta)}[R(\tau)] \tag{5.12}$$

Then policy gradient can be derived as follows:

$$\begin{aligned} \nabla_{\vartheta} J(\vartheta) &= \int R(\tau) \nabla_{\vartheta} p(\tau; \vartheta) d\tau \\ &= \int R(\tau) \frac{\nabla_{\vartheta} p(\tau; \vartheta)}{p(\tau; \vartheta)} p(\tau; \vartheta) d\tau \\ &= \int (R(\tau) \nabla_{\vartheta} \log p(\tau; \vartheta)) p(\tau; \vartheta) d\tau \\ &= \mathbb{E}_{p(\tau; \vartheta)}[R(\tau) \nabla_{\vartheta} \log p(\tau; \vartheta)] \end{aligned} \tag{5.13}$$

From Equation 5.13 we have that the gradient of J w.r.t. ϑ is equal to the expected value of the function $G(\tau, \vartheta) = R(\tau)\nabla_{\vartheta} \log p(\tau; \vartheta)$. This function provides an unbiased estimate of the gradient of J and can be computed by running policy π_{ϑ} and sampling a trajectory τ without knowing the dynamics of the system since $p(x_{t+1}|x_t, u_t)$ does not depend on parameter ϑ. Following this direction is equivalent to running stochastic gradient descent on J.

$$\nabla_{\vartheta} \log p(\tau; \vartheta) = \sum_{t=0}^{L-1} \nabla_{\vartheta} \log \pi_{\vartheta}(u_t|\tau_t) \tag{5.14}$$

The policy gradient algorithm can be summarized:

1. Choose ϑ_0, stepsize sequence α_k, and set $k = 0$;
2. Run the simulator with policy π_{ϑ_k} and sample τ_k;
3. $\vartheta_{k+1} = \vartheta_k + \alpha_k R(\tau_k) \sum_{t=0}^{L-1} \nabla_{\vartheta} \log \pi_{\vartheta}(u_{tk}|\tau_t)$;
4. $k = k + 1$, go to step 2.

The policy gradient algorithm can be run on any problem if sampling from π_{ϑ} can be done efficiently. Policy gradient is simple as it optimizes over a parametric family $p(u; \vartheta)$ instead of optimizing over the space of all probability distributions. However, there are constraints regarding the probability distribution, which should be easy to sample from, easy to search by gradient methods, and rich enough to approximate delta functions. In addition, the complexity of the method depends on the dimensionality of the search space and can be slow to converge. Finally, the policy gradient update is noisy, and its variance increases proportionally with the simulation length L.

The other solution to the problem of dealing with non-differentiable output is to use the the *Gumbel-Softmax* [185] approach, and replace the non-differentiable sample from the categorical distribution with a differentiable sample from a Gumbel-Softmax distribution. The Gumbel-Softmax distribution is a continuous distribution on the simplex that can approximate categorical samples. Parameter gradients can be easily computed by applying the reparameterization trick [216], a popular technique used in variational inference and adversarial learning of generative models in which the expectation of a measurable function g of a random variable ϵ is calculated by integrating $g(\epsilon)$ with respect to the distribution of ϵ:

$$\mathbb{E}(g(\epsilon)) = \int g(\epsilon) dF_\epsilon \tag{5.15}$$

Therefore, in order to compute the expectation of $z = g(\epsilon)$ we do not need to know explicitly the distribution of z, but only know g and the distribution of ϵ. This can alternatively be expressed as:

$$\mathbb{E}_{\epsilon \sim p(\epsilon)}(g(\epsilon)) = \mathbb{E}_{z \sim p(z)}(z) \tag{5.16}$$

If the distribution of variable z depends on parameter ϕ, i.e. $z \sim p_\phi(z)$, and if we can assume $z = g(\epsilon, \phi)$ for a known function g of parameters ϕ and noise distribution $\epsilon \sim \mathcal{N}(0, 1)$, then for any measurable function f:

$$\begin{aligned} \mathbb{E}_{\epsilon \sim p(\epsilon)}(f(g(\epsilon, \phi))) &= \mathbb{E}_{z \sim p_\phi(z)}(f(z)) \\ \mathbb{E}_{\epsilon \sim p(\epsilon)}(\nabla f(g(\epsilon, \phi))) &= \nabla_\phi \mathbb{E}_{\epsilon \sim p(\epsilon)}(f(g(\epsilon, \phi))) \\ &= \nabla_\phi \mathbb{E}_{z \sim p_\phi(z)}(f(z)) \end{aligned} \tag{5.17}$$

In equation 5.17, z has been conveniently expressed such that functions of z can be defined as integrals w.r.t. to a density that does not depend on the parameter ϕ. Constructing unbiased estimates of the gradient is done using Monte Carlo methods:

$$\nabla_\phi \mathbb{E}_{z \sim p_\phi(z)}(f(z)) \sim \frac{1}{M} \sum_{i=1}^{M} \nabla f(g(\epsilon^i, \phi)) \tag{5.18}$$

The reparameterization trick aims to make the randomness of a model an input to that model instead of letting it happen inside the model. Given this, the network model is deterministic and we can differentiate with respect to sampling from the model. An example of applying the reparameterization trick is to rewrite samples drawn from the normal distribution $z \sim \mathcal{N}(\mu, \sigma)$ as $z = \mu + \sigma\epsilon$, with $\epsilon \sim \mathcal{N}(0, 1)$. In this way stochastic nodes are avoided during backpropagation. However, the re-parameterization trick cannot be directly applied to discrete valued random variables, for eg. text data, as gradients cannot backpropagate through discrete nodes in the computational graph.

The Gumbel-Softmax trick attempts to overcome the inability to apply the reparameterization trick to discrete data. It parameterizes a discrete distribution in terms of a Gumbel distribution, i.e. even if the corresponding function is not continuous, it will be made continuous by applying a continuous approximation to it. A random variable G has a standard Gumbel distribution if $G = -\log(-\log(U)), U \sim$ Unif$[0, 1]$. Any discrete distribution can be parameterized in terms of Gumbel random variables as follows. If X is a discrete random variable with $P(X = k) \propto \alpha_k$ random variable and $\{G_k\}_{k \leq K}$ an i.i.d. sequence of standard Gumbel random variables, then:

$$X = \arg\max_k(\log \alpha_k + G_k) \tag{5.19}$$

Equation 5.19 illustrates sampling from a categorical distribution: draw Gumbel noise by transforming uniform samples, add it to $\log \alpha_k$, then take the value of k that yields the maximum. The $\arg\max$ operation that relates the Gumbel samples is not continuous, however discrete random variables can be expressed as one-hot vectors and take values in the probability simplex:

$$\Delta^{K-1} = \{x \in R_+^K, \sum_{k=1}^{K} x_k = 1\} \tag{5.20}$$

A one_hot vector corresponds to a discrete category, and since the $\arg\max$ function is not differentiable, a softmax function can be used instead as a continuous approximation of $\arg\max$:

$$f_\tau(x)_k = \frac{\exp(x_k/\tau)}{\sum_{k=1}^{K} \exp(x_k/\tau)} \tag{5.21}$$

Therefore, the sequence of simplex-valued random variables X^τ is:

$$\begin{aligned} X^\tau = (X_k^\tau)_k &= f_\tau(\log \alpha + G) \\ &= \frac{\exp((\log \alpha_k + G_k)/\tau)}{\sum_{i=1}^{K} \exp((\log \alpha_i + G_i)/\tau)} \end{aligned} \tag{5.22}$$

Equation 5.22 is known as the Gumbel-Softmax distribution and can be evaluated exactly for different values of x, α and τ, where τ is a temperature parameter that controls how closely the samples from the Gumbel-Softmax distribution approximate those from the categorical distribution. When $\tau \to 0$, the softmax function becomes an $\arg\max$ function and the Gumbel-Softmax distribution becomes the categorical distribution. At training time τ is a set to a value greater than 0 which allows gradients to backpropagate past the sample, and then is gradually annealed to a value close to 0. The Gumbel Softmax trick is important as it allows for the inference and generation of discrete objects. A direct application of this technique is generating text via GANs.

In summary, GANs have shown impressive performance at generating natural images nearly indistinguishable from real images, however applying GANs to text generation is a non-trivial task due to the special nature of the linguistic representation. According to Dai et al [77], the two main challenges to overcome when using GANs with textual input are:

i) first, text generation is a sequential non-differentiable sampling procedure which samples a discrete token at each time step (vs. image generation where the transformation from the input random vector to the produced output image is a deterministic continuous mapping); the non-differentiability of text makes it difficult to apply back-propagation directly, and to this end, classical reinforcement learning methods such as Policy Gradient [446] have been used. In policy gradient the production of each word is considered as an action for which the reward comes from the evaluator, and gradients can be back-propagated by approximating the stochastic policy with a parametric function.

ii) second, in the GAN setting the generator receives feedback from the evaluator when the entire sample is produced, however for sequence generation this causes difficulties during training, such as vanishing gradients and error propagation. To allow the generator to get early feedback when a text sequence is partly generated, Monte Carlo rollouts are used to calculate the approximated expected future reward. This has been found empirically to improve the efficiency and stability of the training process.

Unlike in conventional GAN settings that deal with image generation, the production of sentences is a discrete sampling process, which is also non-differentiable. A natural question that arises is how can the feedback be back-propagated from the discriminator to the generator under such a formulation. Policy gradient considers a sentence as a sequence of actions, where each word w_t is an action and the choices of such actions are governed by a policy π_θ. The generative procedure begins with an initial state $S_{1:0}$ which is the empty sentence, and at each time step t the policy π_θ takes as input the previously generated words $S_{1:t-1}$ up until time $t-1$, as well as the noise vector z, and yields a conditional distribution $\pi_\theta(w_t|z, S_{1:t-1})$ over the vocabulary words. The computation is done one step at a time moving along the LSTM network and sampling an action w_t from the conditional distribution up until w_t will be equal to the end of sentence indicator, in which case the sentence is terminated. The reward for the generated sequence of actions S will be a score r calculated by the discriminator. However, this score can be computed only after the sentence has been completely generated, and in practice this leads to difficulties such as vanishing gradients and very slow training convergence. Early feedback is used to evaluate the expected future reward when the sentence is partially generated, and the expectation can be approximated using Monte Carlo rollouts. The Monte Carlo rollout method is suitable to use when a part of the sentence $S_{1:t}$ has been already generated, and we continue to sample the remaining words of the sentence from the LSTM network until the end of sentence token is encountered. The conditional simulation is conducted n times, which results in n sentences. For each sentence we compute an evaluation score, and the rewards obtained by the simulated sentences are averaged to approximate the expected future reward of the current sentence. In this way updating the generator is possible with feedback coming from the discriminator. The utility of the policy gradient method is that by using the expected future reward the generator is provided with early feedback and becomes trainable with gradient descent.

Yu et al propose SeqGAN [505], a GAN-based sequence generation framework with policy gradient, which is the first work to employ GANs for generating sequences of discrete tokens to overcome the limitations of GANs on textual data. SeqGAN treats the sequence generation procedure as a sequential decision making process [10]. A discriminator is used to evaluate the generated sequence and provide feedback to the generative model to guide its learning. It is a well known problem of GANs that for text data (discrete ouputs) the gradient cannot be passed back from the discriminator to the generator. SeqGAN addresses this problem by treating the generator as a stochastic parameterized policy trained via policy gradient [446] and optimized by directly performing gradient policy update, therefore avoiding the differentiation difficulty for discrete data. The reinforcement learning reward comes from the discriminator based on the likelihood that it would be fooled judged on a complete sequence of tokens, and is passed back to the intermediate state-action steps using Monte Carlo search [37].

The sequence generation problem is defined as follows. Given a dataset of human written sequences, train a generative model G_θ parameterized by θ to output sequence $Y_{1:T} = (y_1, \ldots, y_t, \ldots, y_T), y_t \in Y$, where Y is the word vocabulary. The current state is the sequence of tokens $(y_1, \ldots, y_{t-1})$ generated until timestep t, and the action a taken from this state is the selection of next token y_t. The policy model $G_\theta(y_t|Y_{1:t-1})$ is stochastic and will select an action according to the leant probability distribution of the input tokens. The state transition from the current state $s = Y_{1:t-1}$ to the next state $s^{'} = Y_{1:t}$ after choosing action $a = y$ is deterministic, i. e. $\delta^a_{s,s^{'}} = 1$ for next state $s^{'}$, and $\delta^a_{s,s^{''}} = 0$ for other next states $s^{''}$. The discriminative model $D_\phi(Y_{1:T})$ is used to guide the generator G_θ, and outputs a probability indicating how likely a sequence $Y_{1:T}$ produced by G_θ comes from real sequence data. D_ϕ is trained with both real and fake examples from the real sequence data and the synthetic data generated by G_θ. The objective of the generator model (policy) $G_\theta(y_y|Y_{1:t-1})$ is to maximize its expected end reward R_T which comes from the discriminator D_ϕ for a sequence which is generated starting from initial state s_0:

$$J(\theta) = \mathbb{E}[R_T|s_0, \theta] = \sum_{y_1 \in Y} G_\theta(y_1|s_0) Q_{D_\phi}^{G_\theta}(s_0, y_1) \tag{5.23}$$

The action-value function $Q_{D_\phi}^{G_\theta}(s, a)$ for a sequence represents the expected cumulative reward starting from state s, taking action a and then following policy G_θ. The action value function $Q_{D_\phi}^{G_\theta}(s, a)$ is calculated as the estimated probability (reward) the discriminator $D_\phi(Y_{1:T}^n)$ assigns to the generated sample being real:

$$Q_{D_\phi}^{G_\theta}(a = y_T, s = Y_{1:T-1}) = D_\phi(Y_{1:T}^n) \tag{5.24}$$

In the GAN setup, the discriminator D_ϕ can only provide a reward at the end of a finished sequence. In order to evaluate the action-value function $Q_{D_\phi}^{G_\theta}(s, a)$ for an intermediate state s, Monte Carlo search with roll-out policy G_β (identical to the generator G_θ policy) is used to sample the unknown remaining $T - t$ tokens that result in a complete sentence. The roll-out policy G_β starts from the current state s and is run for N times to get an accurate assessment of the action-value function $Q_{D_\phi}^{G_\theta}(s, a)$ through a batch of N output samples, thus reducing the variance of the estimation:

$$\begin{aligned} \{Y_{1:T}^1, \ldots, Y_{1:T}^N\} &= MC^{G_\beta}(Y_{1:t}; N) \\ Q_{D_\phi}^{G_\theta}(a = y_t, s = Y_{1:t-1}) &= \begin{cases} \frac{1}{N}\sum_{n=1}^{N} D_\phi(Y_{1:T}^n), \\ \text{if } Y_{1:T}^n \in MC^{G_\beta}(Y_{1:t;N}), t < T \\ D_\phi(Y_{1:t}), \text{if } t = T \end{cases} \end{aligned} \tag{5.25}$$

The generator starts with random sampling at first, but once more realistic samples have been generated, the discriminator D_ϕ is updated (which will in turn improve the generator model iteratively):

$$\min_\phi -\mathbb{E}_{Y \sim p_{\text{data}}}[\log D_\phi(Y)] - \mathbb{E}_{Y \sim G_\theta}[\log(1 - D_\phi(Y))] \tag{5.26}$$

The generator G_θ is updated every time a new discriminator D_ϕ has been obtained. The gradient of the generator's objective function $J(\theta)$ w.r.t the generator's parameters θ is expressed as follows:

$$\begin{aligned} \nabla_\theta J(\theta) = \sum_{t=1}^{T} \mathbb{E}_{Y_{1:t-1} \sim G_\theta} \Bigg[\sum_{y_t \in Y} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot \\ \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t) \Bigg] \end{aligned} \tag{5.27}$$

Expectation $\mathbb{E}$ can be approximated by sampling methods, and generator's parameters are updated:

$$\theta \leftarrow \theta + \alpha_h \nabla_\theta J(\theta), \text{where } \alpha_h - \text{learning rate} \tag{5.28}$$

In the initial stages of training, the generator G_θ is pre-trained via maximum likelihood estimation, and the discriminator D_ϕ is pre-trained via minimizing the cross-entropy between the ground truth label and the predicted probability; after the pre-training stage is over, the generator and the discriminator are trained alternatively. The SeqGAN authors chose an LSTM [398] architecture for the generator in order to avoid the vanishing and the exploding gradient problem of back-propagation through time, and a CNN [240], [212] architecture with highway networks [435] as discriminator. The evaluation metric is set to minimize the average negative log-likelihood between the generated data and an oracle considered as the human observer:

$$\text{NLL}_{\text{oracle}} = -\mathbb{E}_{Y_{1:T \sim G_\theta}} \left[\sum_{t=1}^{T} \log G_{\text{oracle}}(y_t | Y_{1:t-1}) \right] \tag{5.29}$$

Lin et al [266] consider that GANs restrict the discriminator too much by forcing it to be a binary classifier. Because of this setup, the discriminator is limited in its learning capacity especially for tasks with a rich structure, such as when generating natural language expressions. The authors propose a generative adversarial framework called RankGAN, which is able to capture the richness and diversity of language by learning a relative ranking model between the machine written and human written sentences in an adversarial framework. The adversarial network consists of two neural network models, a generator G_θ and a ranker R_ϕ, where θ and ϕ are parameters. The RankGAN discriminator R_ϕ, instead of performing a binary classification task as in conventional GANs, is trained to rank the machine-written sentences lower than human-written sentences w.r.t. a human-written reference set. Alternatively, the generator G_θ is trained to confuse the ranker R in such a way that machine written sentences are ranked higher than human written sentences with regard to the reference set. The authors consider that by viewing a set of samples collectively (instead of just one sample) and evaluating their quality through relative ranking, the discriminator can make better judgements regarding the quality of the samples, which helps in turn the generator better learn to generate realistic sequences. The problem can be expressed mathematically as G_θ and R_ϕ playing a minimax game with the objective function $\mathcal{L}$:

$$\begin{aligned} \min_\theta \max_\phi \mathcal{L}(G_\theta, R_\phi) = \mathbb{E}_{s \sim P_h}[\log R_\phi(s|U, C^-)] + \\ \mathbb{E}_{s \sim G_\theta}[\log(1 - R_\phi(s|U, C^+))] \end{aligned} \tag{5.30}$$

The ranker R_ϕ is optimized to increase the likelihood of assigning a high probability to the real sentence s and a low probability to the fake generated data G_θ. $s \sim P_h$ denotes that sentence s is sampled from human written sentences, while $s \sim G_\theta$ denotes that sentence s is sampled from machine written sentences. U is a reference set which is used for estimating relative ranks. C^+ and C^- are comparison sets with regards to input sentences. When the input sentence s is sampled from the real data, C^- is sampled from the generated data, and alternatively when the sentence s is sampled from the synthetic data generated by G_θ, C^+ is sampled from human written data.

Similar to SeqGAN, the authors use policy gradient to overcome the non-differentiability problem of text data. However, unlike SeqGAN, the regression based discriminator is replaced with a ranker and a new learning objective function. The generative model G_θ is an LSTM network, while the ranker R_ϕ is a CNN network. The rewards for training the model are encoded with relative ranking information. When a sequence is incomplete, an intermediate reward is computed using Monte Carlo rollout methods. The expected future reward V for partial sequences is defined as:

$$V_{\theta,\phi}(s_{1:t-1,U}) = \mathbb{E}_{s_r \sim G_\theta}[\log R_\phi(s_r|U, C^+, s_{1:t-1})] \tag{5.31}$$

In Equation 5.31 above, s_r denotes a complete sequence sampled by using rollout methods starting from sequence $s_{1:t-1}$. A total of n different paths are sampled, and their corresponding ranking scores are computed. The average ranking score is used to approximate the expected future reward for the current partially generated sequence $s_{1:t-1}$; the ranking score of an input sentence s given reference sentence u and comparison set C (where $C = C^+$ if sentence s is machine generated, $C = C^-$ otherwise) is computed using a softmax-like formula:

$$\begin{gathered} P(s|u, C) = \frac{\exp(\gamma\alpha(s|u))}{\sum_{s' \in C'} \exp(\gamma\alpha(s'|u))}, \text{where} \\ \alpha(s|u) = \cos(y_s, y_u) = \frac{y_s y_u}{||y_s||||y_u||} \end{gathered} \tag{5.32}$$

In Equation 5.32, y_s is the embedded feature vector of the input sentence, and y_u is the embedded feature vector of the reference sentence. The gradient of the objective function for generator G_θ for start state s_0, vocabulary V, and generator policy π_θ is computed as:

$$\begin{aligned} \nabla_\theta \mathcal{L}_\theta(s_0) = \mathbb{E}_{s_{1:T} \sim G_\theta}\Bigg[\sum_{t=1}^{T} \sum_{w_t \in V} \nabla_\theta \pi_\theta(w_t|s_{1:t-1}) \cdot \\ \cdot V_{\theta,\phi}(s_{1:t}, U)\Bigg] \end{aligned} \tag{5.33}$$

Therefore, RankGAN deals with the gradient vanishing problem of GAN by replacing the original binary classifier discriminator with a ranking model in a learning-to-rank framework. The ranking score is computed by taking a softmax over the expected cosine distances from the generated sequences to the real data.

Guo et al [154] find that a limitation of current GAN frameworks for text generation [505], [266], [373], [55], [256], [519] is that they are only capable of generating short texts, within a limited length of around 20 tokens. Generating longer sequences is a less studied but more challenging research problem with a lot of useful applications, such as the auto-generation of news articles or product descriptions. Nevertheless, long text generation faces the issue that the binary guiding signal from generator D is sparse and non-informative; it does not provide useful information regarding the intermediate syntactic structure and semantics of the generated text so that the generator G could learn from that signal. Besides that, it is only available after the entire sequence has been generated, and the final reward value does not provide much guidance on how to alter the parameters of G at training time. Moreover, the approach of relying on binary feedback from the discriminator requires a very large number of real and generated samples to improve G. Aiming to make the guiding signal coming from the discriminator D more informative, the authors propose LeakGAN [154], a GAN approach for adversarial text generation in which the discriminative model D is allowed to leak its own high-level extracted features (in addition to providing the final reward value) to better guide the training of the generative model G. The authors pick a hierarchical generator for G, which is made up of two distinct modules: a *high-level manager* module, and a *low-level worker* module. The high level manager module (or mediator) receives the feature map representation of the discriminator D; this is not normally allowed in the conventional GAN setup as this feature map is internally maintained by the discriminator. The manager embeds this feature map representation coming from the discriminator and passes it over to the worker module. The worker first encodes the current generated sequence, and combines this resulting encoding with the embedding produced by the manager to decide what action to take at the current state. Therefore, LeakGAN "leaks" guiding signals from the discriminator D to the generator G more frequently and more informatively throughout the sequence generation process and not at the end only, helping G improve better and faster.

The discriminator D_ϕ is made up of a feature extractor $\mathcal{F}(.;\phi_f)$ and a final sigmoid classification layer. For input sequence s, D_ϕ is defined as:

$$D_\phi(s) = \text{sigmoid}(\phi_l^T \mathcal{F}(s;\phi_f)) = \text{sigmoid}(\phi_l^T f) \tag{5.34}$$

The feature vector in the last layer of D_ϕ is denoted as $f = \mathcal{F}(s;\phi_f)$, and it will be leaked to the generator G_θ. A natural implication of this approach is that the reward the generator G_θ receives for a partially generated sequence is directly related to the quality of the extracted features by the discriminator D_ϕ. Therefore, for the discriminator D_ϕ to yield a high reward, it is necessary to find a highly rewarding region in the extracted feature space. The authors consider that compared to a scalar signal, the feature vector f is more informative as it captures the position of the generated words in the extracted feature space. D_ϕ is implemented as a CNN network. The manager module $\mathcal{M}(f_t, h^M_{t-1}; \theta_m)$ of the hierarchical generator G_θ receives as input the extracted feature vector f_t, which it combines with its internal hidden state to produce the goal vector g_t:

$$\begin{aligned} g_t^{'} &= \mathcal{M}(f_t, h^M_{t-1}; \theta_m) \\ g_t &= \frac{g_t^{'}}{||g_t^{'}||} \end{aligned} \tag{5.35}$$

The goal vector embedding w_t of goal g_t is computed by applying a linear transformation ψ with weight matrix W_ψ to the sum of recent c goals:

$$w_t = \psi(\sum_{i=1}^{c} g_{t-i}) = W_\psi(\sum_{i=1}^{c} g_{t-i}) \tag{5.36}$$

w_t is fed to the worker module $\mathcal{W}(.; \theta_w)$, which is in charge with the generation of the next token. The worker module takes the current word x_t as input and outputs matrix O_t; this matrix is then combined through a softmax with the goal vector embedding w_t:

$$\begin{aligned} O_t, h^W_t &= \mathcal{W}(x_t, h^W_{t-1}; \theta_w) \\ G_\theta(.|s_t) &= \text{softmax}(O_t w_t / \alpha) \end{aligned} \tag{5.37}$$

At training time, the manager and the worker modules are trained separately – the manager is trained to predict which are the most rewarding positions in the discriminative feature space, while the worker is rewarded to follow these directions. The gradient for the manager module is defined as:

$$\nabla^{\text{adv}}_{\theta_m} g_t = -Q_{\mathcal{F}}(s_t, g_t) \nabla_{\theta_m} d_{\cos}(f_{t+c} - f_t, g_t(\theta_m)) \tag{5.38}$$

$Q_{\mathcal{F}}(s_t, g_t)$ defines the expected reward under the current policy and can be approximated using Monte Carlo search. $d_{\cos}$ computes cosine similarity between the goal vector $g_t(\theta_m)$ produced by the manager, and the change in feature representation $f_{t+c} - f_t$ after c transitions. In order to achieve a high reward, the loss function is trying to force the goal vector to match the transition in feature space. Before the adversarial training takes place, the manager undergoes a pre-training stage with a separate training scheme which mimics the transition of real text samples in the feature space:

$$\nabla_{\theta_m}^{\text{pre}} = -\nabla_{\theta_m} d_{\cos}(f'_{t+c} - f'_t, g_t(\theta_m)) \tag{5.39}$$

The worker uses the REINFORCE algorithm during training to maximize the reward when taking action x_t given the previous state is s_{t-1}:

$$\begin{aligned} \nabla_{\theta_w} \mathbb{E}_{s_{t-1} \sim G} \left[\sum_{x_t} r_t^I \mathcal{W}(x_t | s_{t-1}; \theta_w) \right] = \\ \mathbb{E}_{s_{t-1} \sim G, x_t \sim \mathcal{W}(x_t | s_{t-1})} \left[r_t^I \nabla_{\theta_w} \log \mathcal{W}(x_t | s_{t-1}; \theta_w) \right] \\ r_t^I = \frac{1}{c} \sum_{i=1}^{c} d_{\cos}(f_t - f_{t-i}, g_{t-i}) \end{aligned} \tag{5.40}$$

During the adversarial training process, the generator G_θ and the discriminator D_ϕ are trained in alternative stages. When the generator G_θ is trained, the worker $\mathcal{W}(.;\theta_w)$ and the manager $\mathcal{M}(.;\theta_m)$ modules are trained alternatively fixing each other.

Mode collapse [145] is a common problem when training GAN models, when the generator learns to produce samples with extremely low variety, limiting the usefulness of the leant GAN model. In mode collapse the generator network learns to output samples from a few modes of the data distribution only, missing out on many other modes even though samples from these missing modes can be found throughout the training data. Mode collapse can range from complete collapse, when the generated samples are entirely identical, to partial collapse when the generated samples present some common properties [434], [390]. Several attempts have been made to address the problem, which include: *i)* directly encouraging the generator cost function to account for the diversity of the generated batches by comparing these samples across a batch in order to determine whether the entire batch is real or fake, *ii)* anticipate counterplay, in which the generator learns to fool the discriminator before the discriminator has a chance to respond (and therefore taking counterplay into account), *iii)* experience replay, which minimizes the switching between modes by showing old fake generated samples to the discriminator every now and then, and *iv)* using multiple GANs, in which a GAN is trained for each different mode so that when combined, the GANs altogether cover all modes.

In LeakGAN, in order to address mode collapse, the authors propose an interleaved training scheme, which combines supervised training using maximum likelihood estimation with GAN adversarial training (instead of carrying only GAN adversarial training after the pretraining stage). Blending two training schemes is considered useful by the authors as it helps LeakGAN overcome local minimums, alleviates mode collapse and acts as an implicit regularizer on the generative model.

5.6.2 Samples produced by the review generators

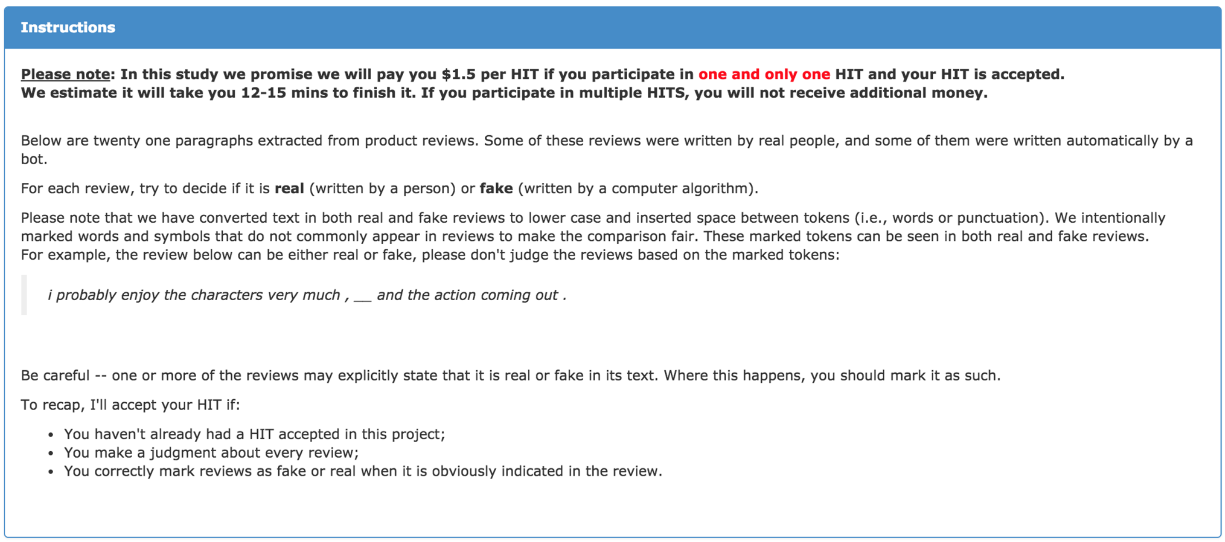


Figure 5.8: Screenshot with instructions presented to Amazon Mechanical Turk workers.

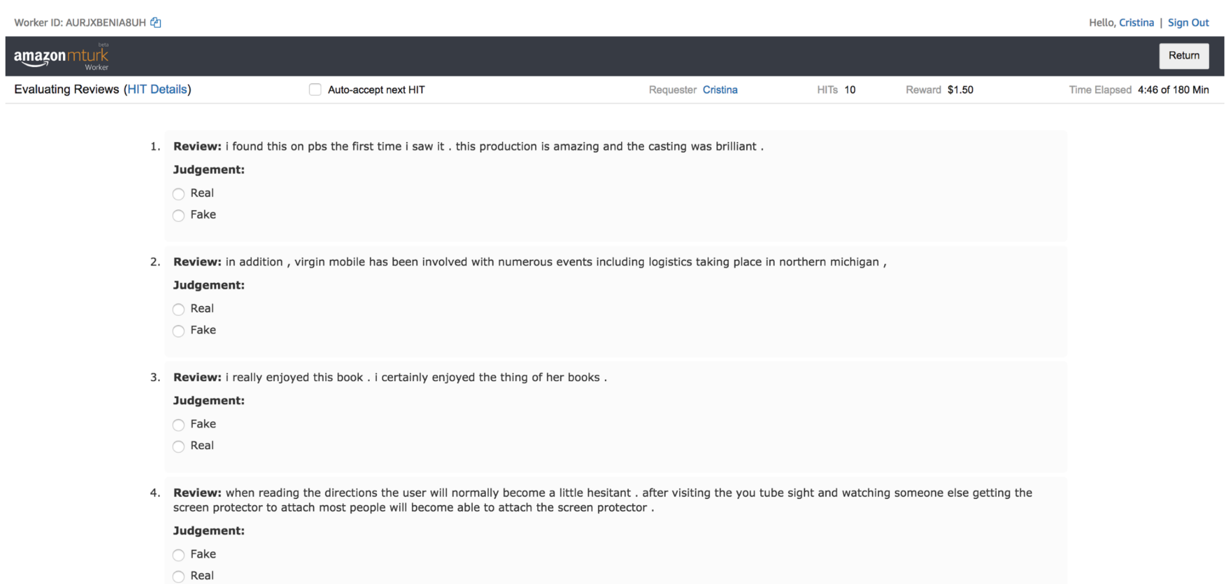


Figure 5.9: Screenshot of the Amazon Mechanical Turk user study interface.

Figure 5.8 shows the instructions given to the AMT workers who participated in this study. In

Figure 5.9 we include a screen-shot of the user interface when annotating reviews.

In what follows we present samples generated by the review generators on which human annotators disagree most on whether these are human-written or machine-generated.

- Word LSTM temp 1.0
    a) i so enjoyed this book . i felt __ though . i especially like loving horses in the __ . and the story is well written .
    b) one of a different type on locked paranormal / vacation book . i enjoyed the characters and the plot . great mixture of historical fiction .
    c) this first edition of the complete series 8 years over six episodes just makes you laugh . the original tv is by far my cup of tea !
    d) works out of the box ! wouldn ' t spend the money for a better keyboard . use this with the matching kindle screen as well .
- Word LSTM temp 0.7
    a) i am looking forward to the next book . i am a __ __ and i enjoyed the story . i like books where the characters are real .
    b) this is an exciting book i could n ' t put down . i will probably read more books by this author . this is a must read .
    c) okay , that ' s how i expected this movie . it was okay but it was so boring . i was bored and was disappointed .
    d) this cable is not bad . it is so cheap and it works great . i ' ve used this for a couple of months now and __ on the ipad
- Word LSTM temp 0.5
    a) this book was a great read ! the story was exciting and a bit __ . i really enjoyed the characters and the story line .
    b) this is a great cable for the price . i would recommend this product to anyone needing a cable for a great price .
    c) this is a great series . it is a must see for anyone who loves period dramas . i love the __ .
    d) these batteries seem to be working as expected . i have had no problems with this product . i would recommend this to anyone .

- Scheduled Sampling

  a) like most of the ones i have ! the tablet that came starts working properly .

  b) i have had any almost using keyboards with an iphone case and kept it nicely and time . and it works well .

  c) have got to watch it many times again and the seasons of __ each episode we can all watch it .

  d) very interesting characters and likable characters that grow when you gave me __ of the __ because of the dog . what can i say is i absolutely loved it .

- Google LM

  a) __ systems generally require less bandwidth and __ with operating systems , __ users to write and edit data nearly anywhere .

  b) seems all but impossible to access . __ is all a __ and gets a bad __ on every __ .

  c) __ is based in __ __ , __ , with a commercial office in __

  d) oved this clip and the __ and __ apps were about so much fun that __ paid a big price . __ 2 and 3 like crazy .

- Attention Attribute to Sequence

  a) i am always waiting for the next book to come out . i am a big fan of sean black and will .

  b) purchased this to use with my macbook pro . it worked out perfectly , as described . no complaints .

  c) great book all of the great mystery books . i enjoyed all of them and was sad when the book ended .

  d) this is a great product . i ’ ve had it for over a year now and it ’ s still going strong . i ’ m very happy with this purchase .

- Contexts to Sequences

  a) i love this series . i love the characters and the story . i love the characters and the story line .

  b) a great book and a great read . i love the characters and the story . i would recommend this book to anyone .

c) i enjoyed the story . it was a good read . i would recommend it to anyone who likes a good read .

d) i love this book and i love the characters . i love this book and i was not disappointed .

- Gated Contexts to Sequences

  a) this is the first book i have read by this author . would recommend to anyone who likes a good romance book .

  b) one of the best books i have ever read . the chemistry between the two main characters was a good read .

  c) this book is awesome . lots of action and intrigue . i ’ m glad i bought this book . thank you for sharing

  d) great story and plot . sometimes a little slow at times but overall a good read .

- MLE SeqGAN

  a) you will like this movie - get this set . . . better than expected award for the characters . bad ending .

  b) this switch converter works fine with all games and works perfect , sturdy program to zero manual products . nice feel .

  c) i could not put it down . it was an interesting clean book , but i was expecting many more individuals in this story so i read in a long time .

  d) great story . in college kids has been lost the __ mysteries , chris __ son is not better .

- SeqGAN

  a) it was slow he kept me interested , and i think i thoroughly enjoyed the story .

  b) i enjoyed this book and look forward to getting to __ larson .

  c) received in excellent condition . i thought it was great but didn ’ t know that movies were more than high ratings which i am my cup of tea .

  d) awesome cute story . kudos to mr much __ of the sookie ’ s story .

- RankGAN

  a) robin williams is ok . just a great movie with __ now . __ is a great film with three stars ! wonderful video for a very good movie .

b) i have loved this movie so i could like the dvd sort of info . hot slow . love the old ford shows to though . __ a great actor .

c) this was a very amazing . __ laws and oh fact she became __ and __ is very unlikely together on the case .

d) i say so i would that originally arrived so i love the circular inch screen . i am sad how it works .

- LeakGAN

a) i really enjoyed reading this book . the author did an excellent job in delivering for all his writing books into us as business . a great summer read .

b) just loved it , so much could read more of this series , i like it but it was not written in a book that is well written , but very interesting .

c) i love hockey - baseball movie coming meets hockey ’ s et addicted fear the birds feature so popular films have developed far worse reviews .

d) a very good book with a lot of twists in this book . i will be checking out more of this author next book .

5.6.3 Results

5.6.3.1 Human Evaluators

We chose the task of distinguishing machine-generated from real reviews because it is a straightforward surrogate of a Turing test. Moreover, how much their generated content can fool humans has been a key claim of many artificial intelligence models recently. The low inter-rater agreement suggests that this is a difficult task even for humans, which we hope would trigger the community to rethink about these claims. There are indeed finer-grained, perhaps more agreeable aspects of text quality (including semantic coherence, syntactic correctness, fluency, adequacy, diversity and readability). We decided not to include them in this experiment for two reasons: 1) as the first study, we are not sure which aspects human raters would consider when they judge for the realism of a review; 2) we wanted to keep the experiment design simple, and many of these aspects are harder to define. In the post-experiment survey, the raters commented on the reasons why they considered reviews as fake.

The low inter-rater agreement (0.27) reflects the difficulty/ subjectivity of the task: identifying individual reviews as human-written or machine-generated. Low human agreement is commonly reported in subjective evaluation tasks. Since our goal is to evaluate the **evaluators** instead of the competing algorithms, it is important to use a task neither too easy or too hard, so that there are distinguishable differences among the performances of competitors (including humans). When using the majority vote of human judgements, the accuracy of humans improved to a reasonable 72.63 %.

5.6.3.2 Discriminative Evaluators

In Table 5.3 and Table 5.4 we present comprehensive results for the meta-adversarial evaluators.

5.6.3.3 Text-Overlap Evaluators

In Figure 5.10 we present detailed results for all word overlap evaluators we used in this study.

5.6.3.4 Comparing Evaluators

In Table 5.5 we present correlation results between the evaluators included in this work.

5.6.3.5 Diversity Analysis

In Table 5.6 we present results for the Self-BLEU metric, while in Table 5.7 we present the correlation of Self-BLEU with the other evaluators. In addition, in Table 5.8 we present correlation

Table 5.3: Accuracy of deep (LSTM) and shallow (SVM) meta-adversarial evaluators. **The lower the better.** Meta-adversarial evaluators do better than humans on individual reviews, with less bias between the two classes. GAN-based generators are considered to be the best by meta-adversarial evaluators.

Generators	LSTM	SVM
Word LSTM temp 1.0	48.29 %	**50.31** %
Word LSTM temp 0.7	92.58 %	78.69 %
Word LSTM temp 0.5	99.31 %	94.74 %
Scheduled Sampling	50.09 %	51.31 %
Google LM	84.58 %	78.59 %
Attention Attribute to Sequence	90.08 %	74.37 %
Contexts to Sequences	100.00 %	100.00 %
Gated Contexts to Sequences	98.37 %	96.26 %
MLE SeqGAN	**41.45** %	52.35 %
SeqGAN	50.05 %	56.20 %
RankGAN	66.28 %	70.17 %
LeakGAN	87.03 %	77.55 %
D-test (all)	77.58 %	74.50 %
D-test (human-written)	80.12 %	75.98 %
D-test (machine-generated)	75.04 %	73.01 %

Table 5.4: Accuracy of deep (LSTM, CNN, CNN & LSTM) and shallow (SVM, RF, NB, XGBoost) meta-adversarial evaluators. **The lower the better.** Meta-adversarial evaluators do better than humans on individual reviews, with less bias between the two classes. GAN-based generators are considered best by meta-adversarial evaluators.

Generators	LSTM	CNN	CNN & LSTM	SVM	RF	NB	XGBoost
Word LSTM temp 1.0	48.29 %	55.22 %	45.68 %	**50.31** %	53.63 %	32.77 %	48.97 %
Word LSTM temp 0.7	92.58 %	93.14 %	91.02 %	78.69 %	81.05 %	79.92 %	80.49 %
Word LSTM temp 0.5	99.31 %	99.35 %	99.08 %	94.74 %	94.29 %	96.86 %	94.71 %
Scheduled Sampling	50.09 %	48.77 %	43.37 %	51.31 %	52.88 %	**20.97** %	44.12 %
Google LM	84.58 %	74.03 %	74.85 %	78.59 %	82.71 %	48.28 %	82.41 %
Attention Attribute to Sequence	90.08 %	91.78 %	89.94 %	74.37 %	77.29 %	80.02 %	71.68 %
Contexts to Sequences	100.00 %	100.00 %	99.97 %	100.00 %	99.98 %	100.00 %	99.98 %
Gated Contexts to Sequences	98.37 %	99.06 %	98.38 %	96.26 %	95.35 %	98.63 %	93.62 %
MLE SeqGAN	**41.45** %	**47.54** %	**41.91** %	52.35 %	**51.14** %	21.83 %	**43.71** %
SeqGAN	50.05 %	52.91 %	47.35 %	56.20 %	54.91 %	25.60 %	48.11 %
RankGAN	66.28 %	67.23 %	59.37 %	70.17 %	61.94 %	35.98 %	61.23 %
LeakGAN	87.03 %	80.28 %	79.57 %	77.55 %	67.74 %	46.80 %	63.80 %
D-test (all)	77.58 %	74.72 %	75.18 %	74.50 %	70.31 %	70.74 %	73.79 %
D-test (human-written)	80.12 %	73.54 %	77.99 %	75.98 %	68.59 %	83.53 %	79.10 %
D-test (machine-generated)	75.04 %	75.90 %	72.38 %	73.01 %	72.04 %	57.95 %	68.48 %

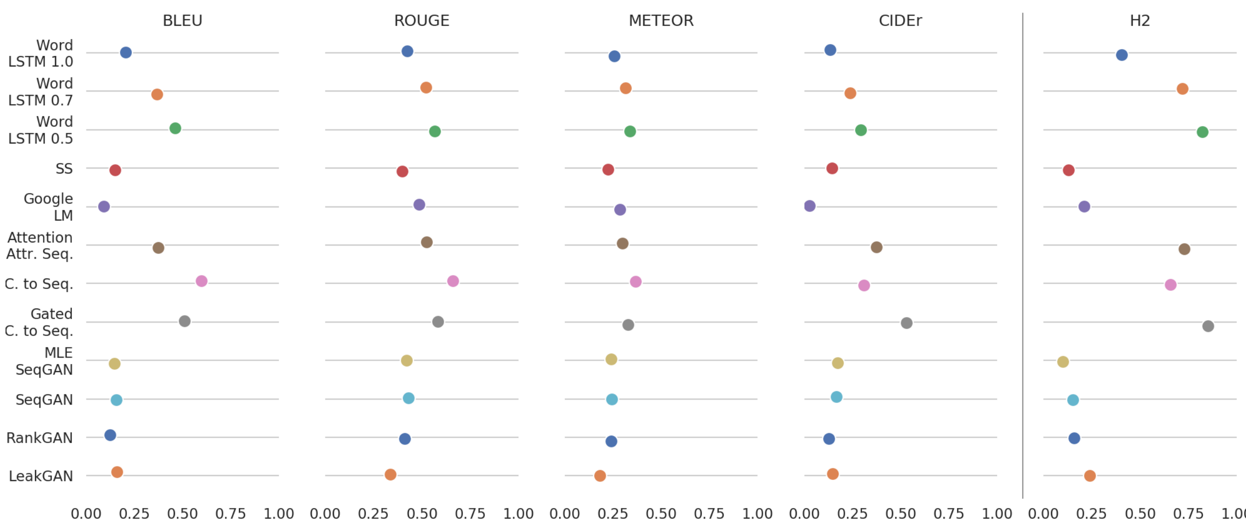


Figure 5.10: Text-Overlap Evaluators (BLEU, ROUGE, METEOR and CIDEr) scores for individual generators. **The higher the better.** The rankings are overall similar, as GAN-based generators are ranked low.

results for BLEU G-Train and the rest of the evaluators.

Evaluation Method	Kendall tau-b (H1)	Spearman (H1)	Pearson (H1)	Kendall tau-b (H2)	Spearman (H2)	Pearson (H2)
SVM Individual-discriminators	-0.4545*	-0.6294*	-0.6716*	-0.5455*	-0.6783*	-0.6823*
LSTM meta-discriminator	-0.5455*	-0.7552*	-0.7699*	-0.6364*	-0.8042*	-0.7829*
CNN meta-discriminator	-0.6363*	-0.8112*	-0.8616*	-0.7273*	-0.8741*	-0.8766*
CNN & LSTM meta-discriminator	-0.6060*	-0.7902*	-0.8392*	-0.6970*	-0.8462*	-0.8507*
SVM meta-discriminator	-0.4545*	-0.6573*	-0.7207*	-0.5455*	-0.6993*	-0.7405
RF meta-discriminator	-0.5455*	-0.7273*	-0.7994*	-0.6364*	-0.7832*	-0.8075*
NB meta-discriminator	-0.6364*	-0.8112*	-0.9290*	-0.7273*	-0.8741*	-0.9388*
XGBoost meta-discriminator	-0.5455*	-0.7413*	-0.7764*	-0.6364*	-0.8042*	-0.7878*
BLEU evaluator	0.7576*	0.8601*	0.8974*	0.6666*	0.8182*	0.9060*
ROUGE evaluator	0.6060*	0.7692*	0.8054*	0.5758*	0.7483*	0.8073*
METEOR evaluator	0.5758*	0.7762*	0.8225*	0.5455*	0.7622*	0.8231*
CIDEr evaluator	0.5455*	0.7413*	0.8117*	0.4545*	0.6643*	0.8203*

Table 5.5: Kendall tau-b, Spearman and Pearson correlation coefficients between human evaluators $H1$, $H2$, and discriminative evaluators and word-overlap evaluators (* denotes statistical significant result with $p \leq 0.05$).

Generative Text Model	Self-BLEU	Lexical diversity
Word LSTM temp 1.0	0.1886	0.6467
Word LSTM temp 0.7	0.4804	0.2932
Word LSTM temp 0.5	0.6960	0.1347
Scheduled Sampling	0.1233	0.7652
Google LM	0.1706	**0.7745**
Attention Attribute to Sequence	0.5021	0.2939
Contexts to Sequences	0.8950	0.0032
Gated Contexts to Sequences	0.7330	0.1129
MLE SeqGAN	0.1206	0.7622
SeqGAN	0.1370	0.7330
RankGAN	**0.1195**	0.7519
LeakGAN	0.1775	0.7541

Table 5.6: Self-BLEU diversity scores per generator (the lower the more diverse), and lexical diversity scores (the higher the more diverse). There is high correlation between the two metrics with respect to the rankings of the generative text models.

Self-BLEU	**Kendall tau-b**	**Spearman**	**Pearson**
H1 evaluator	-0.8788*	-0.9301*	-0.8920*
H2 evaluator	-0.7879*	-0.8881*	-0.9001*
LSTM meta-discriminator	0.6667*	0.8252*	0.7953*
CNN meta-discriminator	0.7576*	0.8811*	0.8740*
CNN & LSTM meta-discriminator	0.7273*	0.8601*	0.8622*
SVM meta-discriminator	0.5758*	0.7413*	0.8518*
RF meta-discriminator	0.6667*	0.8112*	0.8944*
NB meta-discriminator	0.7576*	0.8811*	0.9569*
XGBoost meta-discriminator	0.6667*	0.8252*	0.8693*
BLEU evaluator	-0.8788	-0.9301*	-0.9880*
ROUGE evaluator	-0.7273*	-0.8392*	-0.9299*
METEOR evaluator	-0.6967*	-0.8462*	-0.8955*
CIDEr evaluator	-0.5455*	-0.7413*	-0.7987*

Table 5.7: Kendall tau-b, Spearman and Pearson correlation coefficients between Self-BLEU diversity rankings and the three evaluation methods - human evaluators $H1$, $H2$, discriminative evaluators and word-overlap based evaluators (* denotes statistical significant result with $p \leq 0.05$). Meta-discriminators have been trained on D-train, D-valid sets and tested on the **annotated D-test set with ground-truth test labels**.

BLEU G-train	**Kendall tau-b**	**Spearman**	**Pearson**
H1 evaluator	0.7176*	0.8511*	0.9111*
H2 evaluator	0.6260*	0.8091*	0.9209*
LSTM meta-discriminator	-0.5649*	-0.7461*	-0.7091*
CNN meta-discriminator	-0.6565	-0.7951*	-0.8213*
CNN & LSTM meta-discriminator	-0.6260*	-0.7811*	-0.7951*
SVM meta-discriminator	-0.4428*	-0.6130*	-0.7442*
RF meta-discriminator	-0.5038*	-0.6340*	-0.7864*
NB meta-discriminator	-0.6260*	-0.7601*	-0.9164*
XGBoost meta-discriminator	-0.5649*	-0.6550*	-0.7586*
BLEU evaluator	0.9619*	0.9912*	0.9936*
ROUGE evaluator	0.5954*	0.7496*	0.8717*
METEOR evaluator	0.6260*	0.7636*	0.8477*
CIDEr evaluator	0.6565*	0.8371*	0.8318*

Table 5.8: Kendall tau-b, Spearman and Pearson correlation coefficients between BLEU G-train rankings and the three evaluation methods - human evaluators $H1$, $H2$, discriminative evaluators and word-overlap based evaluators (* denotes statistical significant result with $p \leq 0.05$). Meta-discriminators have been trained on D-train, D-valid sets and tested on the **annotated D-test set with ground-truth test labels**.

5.7 Discussion

5.7.1 User Study

A more detailed list of major clusters of reasons is as follows:

1. Grammar/ typo/ mis-spelling: the language does not flow well.

2. Too general/ too generic/ vagueness: generated reviews are vague, in lack of details.

3. Word choice (wording): in lack of slang, use the wrong words.

4. Flow (not fluent)/ structured/ logical: the sentences level language errors.

5. Contradictory arguments: some arguments support opposite opinions.

6. Emotion: lack of emotion, personality in the comments.

7. Repeated text: using words/ phrases repetitively.

8. Overly same as human: too advertisement, too formal, too likely to be real.

5.7.2 Granularity of Judgements

We charged the Turkers to label individual reviews as either fake or real. Each human judge only annotates 20 reviews, and they do not know which reviews are generated by the same generator. Comparing to an adversarial discriminator, a human judge has not seen many “training” examples of fake reviews or generators. That explains why the meta-adversarial evaluators are better at identifying fake reviews. In this context, humans are likely to judge whether a review is real based on how “similar” it appears to the true reviews they are used to see online. Indeed, other works in the literature find that annotator’s categorization decisions are guided by a small set of examples retrieved from memory at decision time [348], [137]. That is probably why their decisions are better correlated to text-overlap metrics that measures the similarity between a review and a set of references. This hypothesis is supported by a post-experiment survey of the human judges; please see Appendix 5.6.2 for user study samples.

This finding provides interesting implications to the selection of evaluation methods for different tasks. In tasks that are set up to judge individual pieces of generated text (e.g., reviews, translations, summaries, captions, fake news) where there exists human-written ground-truth, it is better to use word-overlap metrics instead of adversarial evaluators. Indeed, when the audience are not trained by reading lots of bot-generated texts, it is more reasonable to use an evaluator that mimics their decision-making process.

In some scenarios, the task is to make judgments in the context of a longer conversation or a set of documents (e.g., conversation agents, dialogue systems, social bots). The difference is that human subjects are exposed to machine-generated text, so that they may be better trained to distinguish fake from real. Moreover, when judgments are made on the agent/ system level (e.g., whether a Twitter account is a bot), signals like how similar the agent outputs are or how much the agent memorizes the training examples may become more useful than word usage, and a discriminative evaluator may be more effective than text-overlap metrics.

Our experiment also provide implications to improving NLG models, which implies that adversarial accuracy might not be the optimal objective for NLG if the goal is to generate documents that humans consider as real. Indeed, a fake review that fools humans does not necessarily need to fool a machine that has seen everything.

In contrast, GAN based models may perform better when judged as a whole system instead of individual items, or in a conversational context. When the human judges have seen enough examples from the same generator, the next example had better be somewhat different.

5.7.3 Imperfect Ground-truth

One important thing to note is that all discriminative evaluators are trained using natural labels (i.e., treating all examples from the Amazon review dataset as positive and examples generated by the candidate models as negative) instead of human-annotated labels. It is possible that if they were trained with human labels, the discriminative evaluators would have been more consistent to the human evaluators. Indeed, some reviews posted on Amazon may have been generated by bots, and if that is the case, treating them as human-written examples may bias the discriminators.

One way to verify this is to consider an alternative "ground-truth". We apply the already trained meta-discriminators to the human-annotated subset (3,600 reviews) instead of the full *D-test* set, and we use the majority vote of human judges (whether a review is fake or real) to surrogate the "ground-truth" labels (whether a review is generated or sampled from Amazon).

Surprisingly, when the meta-adversarial evaluators are tested using human majority-votes as ground-truth, both the accuracy numbers and the rankings of the generators are significantly different from Table 5.3 and Table 5.4 (which used natural labels as ground-truth). We note that the scores and rankings are more inline with the human evaluators. To confirm the intuition, we calculate the correlations between the meta-discriminators and the human evaluators using the annotated subset only. Replacing the natural ground-truth with human annotated labels, the meta-discriminators become positively correlated with human evaluators (Figure 5.6), although BLEU still appears to be the best evaluator.

These results indicate that when the "ground-truth" used by an automated Turing test is ques-

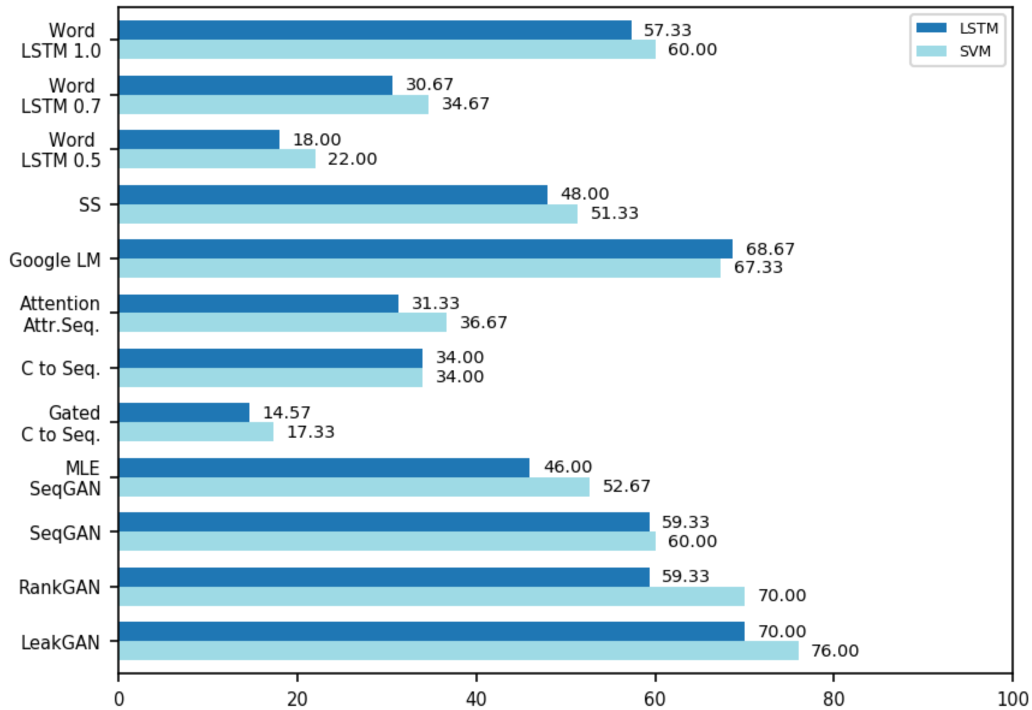


Figure 5.11: Accuracy of deep (LSTM) and shallow (SVM) meta-discriminators when tested on the **annotated subset of *D-test***, with *majority votes* as ground-truth. The lower the better.

tionable, the decisions of the evaluators may be biased. Discriminative evaluators suffer the most from the bias, as they were directly trained using the imperfect ground-truth. Text-overlap evaluators are more robust, as they only take the most relevant parts of the test set as references (more likely to be high quality).

Our results also suggest that when adversarial training is used, the selection of training examples must be done with caution. If the "ground-truth" is hijacked by low quality or "fake" examples, models trained by GAN may be significantly biased. This finding is related to the recent literature of the robustness and security of machine learning models.

5.7.4 Role of Diversity

We also assess the role diversity plays in the rankings of the generators. To this end, we measure lexical diversity [9] of the samples produced by each generator as the ratio of unique tokens to the total number of tokens. We compute in turn lexical diversity for unigrams, bigrams and trigrams, and observe that the generators that produce the least diverse samples are easily distinguished by the meta-discriminators, while they confuse human evaluators the most. Alternatively, samples produced by the most diverse generators are hardest to distinguish by the meta-discriminators, while human evaluators present higher accuracy at classifying them. As reported in [199], the lack of lexical richness can be a weakness of the generators, making them easily detected by a machine learning classifier. Meanwhile, a discriminator's preference for rarer language does not necessarily mean it is favouring higher quality reviews.

In addition to lexical diversity, Self-BLEU [534] is an interesting measurement of the diversity of a set of text (average BLEU score of each document using the same collection as reference, therefore the lower the more diverse). In Figure 5.7 we present Self-BLEU scores for each generator, applied to their generated text in *D-test fake*. We also compute the correlation coefficients between the rankings of generators by Self-BLEU and the rankings by the evaluators (please see Figure 5.12). Results obtained indicate that Self-BLEU presents negative correlation with human evaluators and word-overlap evaluators, and positive correlation with discriminative evaluators. This result confirms the findings in literature [199] that discriminators in adversarial evaluation are capturing known limitations of the generative models such as lack of diversity.

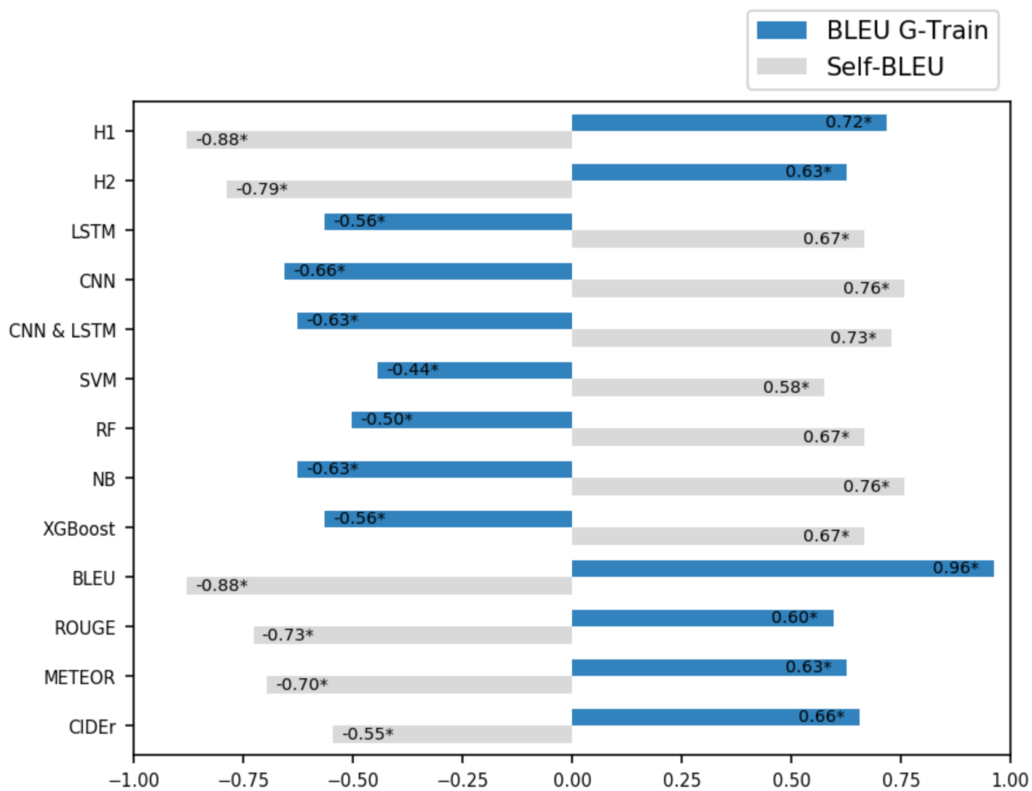


Figure 5.12: Kendall τ-b correlation coefficients between BLEU G-train and Self-BLEU rankings, and the three evaluation methods - human evaluators $H1$, $H2$, discriminative evaluators and word-overlap based evaluators (* denotes $p \leq 0.05$). Meta-discriminators have been trained on D-train, D-valid sets and tested on the **annotated D-test set with ground-truth test labels**.

Following this insight, an important question to answer is to what extent the generators are

Generative Text Model	**BLEU G-Train**
Word LSTM temp 1.0	0.2701
Word LSTM temp 0.7	0.4998
Word LSTM temp 0.5	0.6294
Scheduled Sampling	0.1707
Google LM	0.0475
Attention Attribute to Sequence	0.5122
Contexts to Sequences	0.7542
Gated Contexts to Sequences	0.6240
MLE SeqGAN	0.1707
SeqGAN	0.1751
RankGAN	0.1525
LeakGAN	0.1871

Table 5.9: BLEU results when evaluating the generated reviews using G-train as the reference corpus (a lower score indicates less n-grams in common between the training set G-train and the generated text). GAN models present low similarity with the training set.

simply memorizing the training set *G-train*. To this end, we assess the degree of n-gram overlap between the generated reviews and the training reviews using the BLEU evaluator. In Table 5.9 we present the average BLEU scores of generated reviews using their nearest neighbors in *G-train* as references. We observe that generally the generators do not memorize the training set, and GAN models generate reviews that have fewer overlap with *G-train*. In Figure 5.12 we include the correlation between the divergence from training and the ratings by evaluators in the study. BLEU w.r.t. *G-train* presents highly positive correlation with BLEU w.r.t. *D-test real*, and it is also positively correlated with the human evaluators $H1$ and $H2$.

The effects of diversity is perhaps not hard to explain. At the particular task of distinguishing fake reviews from real, all decisions are made on individual reviews. And because a human judge was not exposed to many fake reviews generated by the same generator, whether or not a fake review is sufficiently different from the other generated reviews is not a major factor for their decision. Instead, the major factor is whether the generated review looks similar to the reviews they have seen in reality. Instead, a discriminative evaluator makes the decision after seeing many positive and negative examples, and a fake review that can fool an adversarial classifier has to be sufficiently different from all other fake reviews it has encountered (therefore diversity of a generator is a major indicator of its ability to pass an adversarial judge).

CHAPTER 6

Conclusions and Future Directions

In this dissertation, we have covered aspects related to natural language generation and evaluation, including the accessibility, steerability, explainability, adaptability and evaluation of current text generation models. We put forward a series of analyses and solutions to existing challenges in the literature, hoping the present work contributes to enhancing our collective understanding of this rapidly evolving research field. As we conclude, we first provide a summary of the work presented, and then discuss future research directions.

6.1 Thesis Summary

- **Chapter 2 – Literature Review: Why is constrained natural language generation particularly challenging?** In this chapter, we presented an extensive survey focused on the emerging problem of neural natural language generation with constraints, differentiating between the ambiguous use of *conditions* and *constraints* in the literature. We outlined approaches, learning methodologies, model architectures, evaluation metrics, as well as limitations of current models and evaluation approaches. We hope this serves as an informative guide towards advancing the state-of-the-art in constrained NLG and evaluation research.

- **Chapter 3 – Explainable Prediction of Text Complexity: The Missing Preliminaries for Text Simplification** We decomposed the ambiguous notion of text simplification into a compact, transparent, and logically dependent pipeline of modular sub-task that increase the transparency and explainability of text simplification systems. We focused on the analysis of the first two steps in this pipeline: 1) predicting whether a given piece of text needs to be simplified at all, and 2) identifying which part of the text needs to be simplified. We demonstrated the importance of these steps: by simply applying explainable complexity prediction as a preliminary step, the out-of-sample text simplification performance of the state-of-the-art, black-box models can be improved by a large margin.

- **Chapter 4 – Adapting Pre-trained Language Models to Low-Resource Text Simplification: The Path Matters** We framed the problem of low-resource text simplification from a task and domain adaptation perspective, and explored ways in which a pre-trained large language model can be fast adapted to new text simplification tasks and domains with few training examples. We find that when directly adapting a Web-scale pre-trained language model to low-resource text simplification tasks, fine-tuning based methods present a competitive advantage over meta-learning approaches. Surprisingly, adding an intermediate stop in the adaptation path between the source and target, an auxiliary dataset and task that allow for the decomposition of the adaptation process into multiple steps, significantly increases the performance of the target task. The performance is however sensitive to the selection and ordering of the adaptation strategy (task adaptation vs. domain adaptation) in the two steps. When such an intermediate dataset is not available, one can build a "pseudostop" using the target domain/task itself. Our extensive analysis serves as a preliminary step towards bridging these two popular paradigms of few-shot adaptive learning and towards developing more structured solutions to task/domain adaptation in a novel setting.

- **Chapter 5 – Judge the Judges: A Large-Scale Evaluation Study of Neural Language Models for Online Review Generation** We conducted a large-scale, systematic experiment to analyze the procedures and metrics used for evaluating NLG models, including human evaluators, automated adversarial evaluators trained to distinguish human-written from machine-generated texts, and word overlap metrics. We find that none of the evaluators is close to perfect. Human evaluators generally do better at correctly labelling human-written reviews as real, and they are confused by machine-generated reviews in close to half of the cases. Adversarial evaluators have more balanced true positive rates and true negative rates compared to human evaluators, and rank GAN-based generators highest. Surprisingly, none of the discriminative evaluators have a positive correlation with the human evaluators. That says, generators that fool machine judges easily are less likely to confuse human judges, and vice versa. Word-overlap evaluators tend to have a positive correlation with the human evaluators in ranking the generators. Our findings represent a preliminary foundation for proposing more solid and robust evaluation metrics and objectives for NLG.

Taken together, our contributions serve to advance our common understanding of specific NLG challenges, inform on concrete solutions to address these challenges, and provide insights for guiding the development of better NLG algorithms and evaluation metrics. Looking beyond the problem of low-resource text simplification we are focusing on in this thesis, we believe the methodology proposed in this dissertation (explainable decomposition, chain of adaptations to new tasks and domains, and meta-evaluation) may benefit other areas related to generative AI. These decomposition practices are well aligned with the philosophy of ”chain-of-thoughts” [477] that is a corner stone of large language models, although the latter appears after our work. We are excited with the progress of the field over the past years, and glad to be able to contribute with the present research. Nevertheless, we acknowledge that these contributions are far from sufficient to solve the long-standing outstanding research challenges related to safe and robust natural language generation and evaluation. We aim to continue working on addressing these problems, and expect to witness the emergence of brand-new models that are trustworthy, efficient, explainable and controllable; this requires innovative ideas and creative approaches that go far beyond the current trend of scaling up existing models. In what follows we outline important directions for future research.

6.2 Future Directions

With an outlook towards the future of natural language generation and evaluation, we present below research directions that are important for further advancing the robustness, efficiency and transparency of current systems.

- **Better Model Architectures:** Autoregressive architectures based on Transformer [463] are the backbone of powerful large language models from the GPT family [367], [368], [369], including ChatGPT [333] or GPT-4 [334]. Nevertheless, these architectures are inherently limited in nature due to the underlying next-word prediction paradigm. Given the output is generated in a forward left-to-right manner, the model cannot backtrack to correct its own mistakes or make edits before reaching the final form. This also limits the ability of the model to do far-ahead planning, which requires complex reasoning abilities and content iterations [39]. Planning is particularly important for constrained text generation, since it requires an organized thinking scheme at both local and global level (i.e. fast thinking vs. slow thinking to “oversee the thought process”) [80].

  Another limitation is the restricted maximum sequence length of current models which does not allow to take longer context into account. Performance substantially decreases as the input context gets longer, and the degradation in performance is significantly worse when relevant information is placed in the middle of the input context, as opposed to beginning or end [273]. This introduces spurious correlations and is detrimental to models’ ability to learn in-context and generalize. Besides, self-attention has quadratic complexity with the input sequence length for vanilla Transformer models. Scaling up sequence length requires finding the right balance between computational complexity and model expressivity [93]. Moreover, it is important to consider improving LLMs training and inference efficiency given their size is making them challenging to use in most real-world applications [126], [175], [172].

- **Robust Natural Language Understanding and Grounding:** With the increase in popularity and widespread deployment of current models, it is important to assess their performance critically. As a proxy to their ability to generalize in real-world situations, their performance is measured on held-out data on various leaderboards. Nevertheless, these held-out datasets are not comprehensive nor representative enough of the diversity of real-world test cases, and may contain harmful, intended or undesirable biases, which may lead to overestimating the actual model performance [237]. Moreover, when model performance is summarized as a single aggregate statistic, it is difficult to understand the failure modes of the model and what can be done to fix them [487]. Comprehensive behavioral testing of NLP models is important not only for quantifying their linguistic abilities [194], but also for performing sanity checks such as prediction invariance in the presence of perturbations or identifying spurious correlations that affect out-of-distribution generalization [384], [192]. While LLMs demonstrate a certain level of understanding of the physical world, the fact they are trained on written text only leaves them unaware of essential embodied knowledge and skills; they are not robust enough to perform many reasoning and planning tasks in physical environments, such as navigation, interaction with objects, sensing and tracking the world state [492]. Grounding language models to world models permits the acquisition of embodied knowledge and skills necessary for solving tasks in physical world.

- **Continual and Lifelong Learning:** Current large language models are trained on static datasets and encode world knowledge in their parameters, however this knowledge can become quickly outdated as the world is non-stationary and fast-changing. When asked to predict future utterances from beyond their training period, LLMs performance degrades and becomes increasingly worse with time [237]; this illustrates their struggle with temporal generalization and emphasizes the contrast between the inherent dynamic nature of language and the current static language modelling paradigm. The challenge to update internal world knowledge without forgetting previously learnt knowledge is nontrivial though. Pre-training language models from scratch with a newly updated text corpus of a similar scale to the one used during the initial pre-training is undesirable due to computational and environmental aspects [349]. In the literature, the acquisition of new world knowledge by LLMs has been approached from either the perspective of *retrieval-augmented generation* [251], [272], [30], [208], [419] where at inference time models search for updated information from external sources, or from a *continual learning* point of view which allows models to continually update their knowledge with new information by continuing the pre-training process on a smaller corpus that contains new knowledge [186], [238], [237]. However, both approaches come with specific challenges: memory-augmented models may suffer from hallucination [511], i.e. making up false information despite being presented with up-to-date knowledge at inference time, while continual learning approaches are known to suffer from catastrophic forgetting [218], i.e. forgetting previously learnt knowledge as they learn new information. Developing adaptive language models that can continually acquire incrementally available information from non-stationary data distributions in our constantly evolving world and with no performance degradation over time represents an important research frontier [346], [116].

- **Better Evaluation Metrics:** Proper evaluation of natural language generation remains particularly challenging. Human evaluations are considered the gold standard for the assessment of natural language generations, nevertheless the lack of standardized setups, consistency, transparency and reproducibility over time and over different annotator populations are important challenges in the way of effective human evaluation of text generation models [210]. In addition, conducting human evaluations is often times a time-consuming and expensive process. To this end, automated evaluation metrics are frequently used as proxy to human judgements; their usefulness however is debatable. Word overlap metrics are found to present low correlation with human judgements for state-of-the art NLG systems [355], [300], [325], [92], [387] or when used as targets for optimization [43], [443]. As NLG models become increasingly powerful, evaluation metrics based on pre-trained large language models are proposed, however their use is not without shortcomings. First, these evaluation metrics are biased towards generations from their own underlying model [163]. Second, they are not robust to various adversarial fooling attacks and simple perturbations, for example lexical overlap with low semantic similarity (indicating the metric can be fooled by using the same words but in different order) and factuality errors [387], [201], [63], [384]. Third, evaluation metrics based on black-box language model representations lack explainability and interpretability of their output scores, which are important aspects for spotting quality issues and establishing trust in model decisions [52], [244]. Given a single overall score assigned is not informative in understanding which aspects of the text generation model need improvement [387], text generation models could be leveraged as explainability tools for evaluation metrics; ideally, a metric does not only output one or several scores, but also provides a textual explanation for the significance of the metric score [244], [201]. For both human and automated evaluations, it is important to report scores with uncertainty estimates to better inform comparisons across different models [210]. Moreover, NLG outputs need not only be correct, but also diverse, however in the literature there are no principled methods for quantifying different aspects of diversity, such as form and content diversity [453].

- **Cautious Deployment of NLP Models in the Real World:** As NLP models are being increasingly deployed with significant impact in the real-world, it is imperative to consider aspects such as factuality, explainability/trustworthiness, fairness/bias mitigation, privacy preservation and alignment with human values. We discuss these challenges in turn below.

Factuality While current models can generate fluent and realistic-looking texts in many downstream tasks and application scenarios, they are also known to generate content that lacks in global and factual consistency, is not entailed by the original input, or is cogent-sounding but simply wrong [303], [449], [528], [267]. Detecting model hallucinations and factual inconsistencies is crucial for reducing the spread of misinformation and for improving trust in model outputs, particularly in real-world, mission-critical applications [231], [353]. The main challenge in evaluating factual precision is that generated content is typically a mixture of both true and false information, therefore factuality cannot be treated as a binary concept [342]. Moreover, not all factual errors are equally important: their number does impact the perceived factuality of a piece of text, however validating every piece of information may not always be possible due to time and cost constraints. In general, the ability of large language models to answer factual questions is associated with the number of documents related to the question the model has seen during pre-training [196]. Evaluations of factuality focus on identifying series of atomic facts and determining what percentage are supported by a reliable knowledge source [312].

Explainability/Trustworthiness Explanations play a central role in human learning, allowing humans to generalize and adapt to new situations [282]. For black-box AI algorithms, elaborating on the rationale used to solve a problem can help increase trust in the inner workings of the model and allow to assess additional safety or non-discrimination criteria [97]. Nevertheless, our understanding of how current black-box models work is still very limited. The field is missing a clear definition of what constitutes explainability, how it connects to the target audience, as well as standard terminology and robust evaluation metrics for measuring the fidelity, causality and faithfulness of the claimed explanations in supporting the model's prediction [82]. In addition to enhancing human's understanding of the model behaviour, explanations may also help the model itself better "understand" the underlying task. For example, explanations of answers in a few-shot prompt are found to improve the performance of in-context learning task inference for LLMs [234].

Fairness/Bias Mitigation The goal of machine learning fairness is to ensure that model inaccuracies and data biases are not going to result in the unfavourable treatment of certain demographics, with negative consequences for individuals and society [417], [45]. Examples of such sensitive attributes in the data include age, race, gender, disabilities, sexual or political orientation, marital status, income, geographical location, etc. Definitions of fairness in the literature account for properties of model outputs with respect to sensitive attributes, as well as relationships among other relevant variables in the data. In general, unfairness in machine learning arises from two sources: biases in the data and biases in algorithms [304]. Approaches for ensuring fairness in the predictions of machine learning models include removing biases from the extracted data representations during pre-processing before using them as model inputs, imposing fairness constraints at training time, or post-processing model outputs to make them fair [332]. Measuring bias in NLP models is done via fairness metrics which quantify differences in the model behaviour across different social categories based on the notions of group and counterfactual fairness [76]. While bias mitigation is a pressing problem, in the literature the notion of bias is conceptualized differently, making it hard to compare across methods and evaluations. Moreover, motivations for analyzing bias are often times vague and inconsistent, terminology is imprecise, and assumptions are being made about what kinds of system behaviors are harmful, to whom, and why [28].

Privacy Large language models can memorize sensitive parts of their training data, which then becomes susceptible to adversarial attacks [49], [47], [468]. Many attacks find overfitting a sufficient condition for privacy leakage, as it often indicates the memorization of training set examples [503]. Using only black-box query access, unique and secret sequences can be extracted with serious negative consequences, such as credit card/ssn numbers, or names of individuals and their contact information. Protecting NLP models from leaking private data is becoming increasingly important given their widespread adoption in real-world products [420]. In the literature, training with differential privacy, fine-tuning an additional small set of parameters with private data and curating datasets are used to prevent leakage of sensitive information from the model training set [48], [504], [21].

Alignment with Human Values For artificial intelligence technologies to be beneficial to our society, it is important that their goals are in line with human values. Nevertheless, the behaviour alignment problem is challenging due to difficulty in precisely defining and measuring human preferences, undesired secondary behaviour that may arise as a result of the primary alignment goal, and a narrowing window of opportunity to correct problems as artificial agents become increasingly more capable [204]. For NLP in particular, current models often violate human preferences and display unintended behaviours, for example they can easily degenerate into biased, offensive and toxic language [130], make up facts and output factually incorrect information [439], or simply not follow user instructions [29]. One of the reasons behind this misaligned behaviour in current LLMs is that the language modeling objective of predicting the next token does not reflect the implicit goal of staying helpful, honest and harmless while following user instructions [36], [369]. In the literature, aligning language models with user intent has been approached from the perspective of fine-tuning with human feedback using reinforcement learning algorithms that leverage human preferences as a reward signal [338], [395], [370], [333], [334]. However, designing suitable reward functions requires more than just large datasets of human labeled preferences; encoding the right inductive biases is equally important [243]. Other challenges include training agents in the lack of well-specified rewards, robustness to distribution shifts and preventing unacceptable outcomes before they even occur. Finally, as language technologies are being increasingly deployed in the real world, the social and ethical impact of NLP research requires careful thought and consideration [173].

BIBLIOGRAPHY

[1] Emil Abrahamsson, Timothy Forni, Maria Skeppstedt, and Maria Kvist. Medical text simplification using synonym replacement: Adapting assessment of word difficulty to a compounding language. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)*, pages 57–65, 2014.

[2] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

[3] Sandra Aluisio, Lucia Specia, Caroline Gasperin, and Carolina Scarton. Readability assessment for text simplification. In *Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9, 2010.

[4] Fernando Alva-Manchego, Joachim Bingel, Gustavo Paetzold, Carolina Scarton, and Lucia Specia. Learning how to simplify from explicit labeling of complex-simplified text pairs. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 295–305, 2017.

[5] Jacopo Amidei, Paul Piwek, and Alistair Willis. Identifying annotator bias: A new irt-based method for bias identification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4787–4797, 2020.

[6] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, 2017.

[7] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. " what is relevant in a text document?": An interpretable machine learning approach. *PloS one*, 12(8):e0181142, 2017.

[8] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, 2021.

[9] Kevin Bache, David Newman, and Padhraic Smyth. Text-based measures of document diversity. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 23–31. ACM, 2013.

[10] Philip Bachman and Doina Precup. Data generation as sequential decision making. In *Advances in Neural Information Processing Systems*, pages 3249–3257, 2015.

[11] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[12] Ashutosh Baheti, Alan Ritter, Jiwei Li, and William B Dolan. Generating more interesting responses in neural conversation models with distributional constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3970–3980, 2018.

[13] Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. Real or fake? learning to discriminate machine from human generated text. *arXiv preprint arXiv:1906.03351*, 2019.

[14] Anusha Balakrishnan, Jinfeng Rao, Kartikeya Upasani, Michael White, and Rajen Subba. Constrained decoding for neural nlg from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 831–844, 2019.

[15] Sriram Balasubramanian, Naman Jain, Gaurav Jindal, Abhijeet Awasthi, and Sunita Sarawagi. What's in a name? are bert named entity representations just as good for any other name? *arXiv preprint arXiv:2007.06897*, 2020.

[16] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[17] Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, 2020.

[18] Yu Bao, Hao Zhou, Shujian Huang, Lei Li, Lili Mou, Olga Vechtomova, Xinyu Dai, and Jiajun Chen. Generating sentences from disentangled syntactic and semantic spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6008–6019, 2019.

[19] Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. Few-shot text classification with distributional signatures. In *International Conference on Learning Representations*, 2020.

[20] Ankur Bapna and Orhan Firat. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, 2019.

[21] Rouzbeh Behnia, Mohammadreza Reza Ebrahimi, Jason Pacheco, and Balaji Padmanabhan. Ew-tune: A framework for privately fine-tuning large language models with differential privacy. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 560–566. IEEE, 2022.

[22] Emily M Bender and Alexander Koller. Climbing towards nlu: On meaning, form, and understanding in the age of data.

[23] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[24] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[25] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[26] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[27] Sumithra Bhakthavatsalam, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, and Peter Clark. Think you have solved direct-answer question answering? try arc-da, the direct-answer ai2 reasoning challenge. *arXiv preprint arXiv:2102.03315*, 2021.

[28] Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. Language (technology) is power: A critical survey of" bias" in nlp. *arXiv preprint arXiv:2005.14050*, 2020.

[29] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[30] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[31] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

[32] Antoine Bosselut, Asli Çelikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. Discourse-aware neural rewards for coherent text generation. In *NAACL-HLT*, 2018.

[33] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, 2016.

[34] Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, 34:15787–15800, 2021.

[35] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*, 2016.

[36] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[37] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[38] Elia Bruni and Raquel Fernández. Adversarial evaluation for open-domain dialogue generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 284–288, 2017.

[39] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[40] Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. Language gans falling short. *arXiv preprint arXiv:1811.02549*, 2018.

[41] Zefan Cai, Baobao Chang, and Wenjuan Han. Human-in-the-loop through chain-of-thought. *arXiv preprint arXiv:2306.07932*, 2023.

[42] Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further meta-evaluation of machine translation. In *Proceedings of the third workshop on statistical machine translation*, pages 70–106. Association for Computational Linguistics, 2008.

[43] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluation the role of bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

[44] Tianshi Cao, Marc T Law, and Sanja Fidler. A theoretical analysis of the number of shots in few-shot learning. In *International Conference on Learning Representations*.

[45] Yang Trista Cao, Yada Pruksachatkun, Kai-Wei Chang, Rahul Gupta, Varun Kumar, Jwala Dhamala, and Aram Galstyan. On the intrinsic and extrinsic fairness evaluation metrics for contextualized language representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 561–570, 2022.

[46] Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. Faithful to the original: Fact aware neural abstractive summarization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[47] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

[48] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[49] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[50] Samuel Carton, Qiaozhu Mei, and Paul Resnick. Extractive adversarial networks: High-recall explanations for identifying personal attacks in social media posts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3497–3507, 2018.

[51] Rich Caruana. Learning many related tasks at the same time with backpropagation. *Advances in neural information processing systems*, 7, 1994.

[52] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*, 2020.

[53] Arun Tejasvi Chaganty, Stephen Mussman, and Percy Liang. The price of debiasing automatic metrics in natural language evaluation. *arXiv preprint arXiv:1807.02202*, 2018.

[54] William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. Kermit: Generative insertion-based modeling for sequences. *arXiv preprint arXiv:1906.01604*, 2019.

[55] Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983*, 2017.

[56] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

[57] Lingjiao Chen, Matei Zaharia, and James Zou. How is chatgpt's behavior changing over time? *arXiv preprint arXiv:2307.09009*, 2023.

[58] Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. Controllable paraphrase generation with a syntactic exemplar. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5972–5984, 2019.

[59] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, 2017.

[60] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[61] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

[62] Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[63] Yanran Chen and Steffen Eger. Menli: Robust evaluation metrics from natural language inference. *arXiv preprint arXiv:2208.07316*, 2022.

[64] Leshem Choshen, Lior Fox, Zohar Aizenbud, and Omri Abend. On the weaknesses of reinforcement learning for neural machine translation. *arXiv preprint arXiv:1907.01752*, 2019.

[65] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[66] Erik Christensen. Methodology of superiority vs. equivalence trials and non-inferiority trials. *Journal of hepatology*, 46(5):947–954, 2007.

[67] Ondřej Cífka, Aliaksei Severyn, Enrique Alfonseca, and Katja Filippova. Eval all, trust a few, do wrong to none: Comparing sentence generation models. *arXiv preprint arXiv:1804.07972*, 2018.

[68] Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A Smith. All that's 'human'is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, 2021.

[69] Elizabeth Clark, Asli Celikyilmaz, and Noah A Smith. Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760, 2019.

[70] Elizabeth Clark, Yangfeng Ji, and Noah A Smith. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2250–2260, 2018.

[71] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[72] Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Daniel S Weld. Pretrained language models for sequential sentence classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3684–3690, 2019.

[73] Pierre Jean A Colombo, Chloé Clavel, and Pablo Piantanida. Infolm: A new metric to evaluate summarization & data2text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10554–10562, 2022.

[74] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[75] Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, et al. Systran's pure neural machine translation systems. *arXiv preprint arXiv:1610.05540*, 2016.

[76] Paula Czarnowska, Yogarshi Vyas, and Kashif Shah. Quantifying social biases in nlp: A generalization and empirical comparison of extrinsic fairness metrics. *Transactions of the Association for Computational Linguistics*, 9:1249–1267, 2021.

[77] Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979, 2017.

[78] Robert Dale and Chris Mellish. Towards evaluation in natural language generation. In *In Proceedings of First International Conference on Language Resources and Evaluation*, 1998.

[79] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*, 2020.

[80] Kahneman Daniel. *Thinking, fast and slow*. 2017.

[81] Wayne W Daniel et al. *Applied nonparametric statistics*. Houghton Mifflin, 1978.

[82] Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. A survey of the state of explainable ai for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational

Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 447–459, 2020.

[83] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: a simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

[84] Oscar Day and Taghi M Khoshgoftaar. A survey on heterogeneous transfer learning. *Journal of Big Data*, 4(1):1–42, 2017.

[85] Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack Rae. Training language gans from scratch. In *Advances in Neural Information Processing Systems*, pages 4302–4313, 2019.

[86] Mostafa Dehghani, Yi Tay, Alexey A Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery. *arXiv preprint arXiv:2107.07002*, 2021.

[87] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[88] Louise Deléger and Pierre Zweigenbaum. Extracting lay paraphrases of specialized expressions from monolingual comparable medical corpora. In *Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, pages 2–10. Association for Computational Linguistics, 2009.

[89] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[90] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[91] Siobhan Lucy Devlin. *Simplifying natural language for aphasic readers.* PhD thesis, University of Sunderland, 1999.

[92] Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, 2019.

[93] Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, and Furu Wei. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*, 2023.

[94] Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, 2021.

[95] Chris Donahue, Mina Lee, and Percy Liang. Enabling language models to fill in the blanks. *arXiv preprint arXiv:2005.05339*, 2020.

[96] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. Learning to generate product reviews from attributes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 623–632, 2017.

[97] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[98] Finale Doshi-Velez and Been Kim. Considerations for evaluation and generalization in interpretable machine learning. In *Explainable and interpretable models in computer vision and machine learning*, pages 3–17. Springer, 2018.

[99] Vincent Dumoulin, Neil Houlsby, Utku Evci, Xiaohua Zhai, Ross Goroshin, Sylvain Gelly, and Hugo Larochelle. Comparing transfer and meta learning approaches on a unified few-shot classification benchmark. *arXiv preprint arXiv:2104.02638*, 2021.

[100] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jian, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D Hwang, et al. Faith and fate: Limits of transformers on compositionality. *arXiv preprint arXiv:2305.18654*, 2023.

[101] Sergey Edunov, Myle Ott, Marc'Aurelio Ranzato, and Michael Auli. On the evaluation of machine translation systems trained with back-translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2836–2846, 2020.

[102] Aparna Elangovan, Jiayuan He, and Karin Verspoor. Memorization vs. generalization: Quantifying data leakage in nlp performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, 2021.

[103] Noemie Elhadad and Komal Sutaria. Mining a lexicon of technical terms and lay equivalents. In *Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing*, pages 49–56. Association for Computational Linguistics, 2007.

[104] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

[105] Allyson Ettinger. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48, 2020.

[106] Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. Tweepfake: About detecting deepfake tweets. *Plos one*, 16(5):e0251415, 2021.

[107] Tobias Falke, Leonardo FR Ribeiro, Prasetya Ajie Utama, Ido Dagan, and Iryna Gurevych. Ranking generated summaries by correctness: An interesting but challenging application for natural language inference. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 2214–2220, 2019.

[108] Angela Fan, David Grangier, and Michael Auli. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54, 2018.

[109] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

[110] Angela Fan, Mike Lewis, and Yann Dauphin. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, 2019.

[111] William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: Better text generation via filling in the _. *arXiv preprint arXiv:1801.07736*, 2018.

[112] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, 2021.

[113] Jessica Ficler and Yoav Goldberg. Controlling linguistic style aspects in neural language generation. *EMNLP 2017*, page 94, 2017.

[114] Edgar C Fieller, Herman O Hartley, and Egon S Pearson. Tests for rank correlation coefficients. i. *Biometrika*, 44(3/4):470–481, 1957.

[115] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[116] Adam Fisch, Amal Rannen-Triki, Razvan Pascanu, Jörg Bornschein, Angeliki Lazaridou, Elena Gribovskaya, and Marc'Aurelio Ranzato. Towards robust and efficient continual language learning. *arXiv preprint arXiv:2307.05741*, 2023.

[117] Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. *Statistical methods for rates and proportions*. John Wiley & Sons, 2013.

[118] Rudolf Franz Flesch. *How to write plain English: A book for lawyers and consumers*. Harpercollins, 1979.

[119] Marina Fomicheva and Lucia Specia. Taking mt evaluation metrics to extremes: Beyond correlation with human judgments. *Computational Linguistics*, 45(3):515–558, 2019.

[120] Markus Freitag, David Grangier, and Isaac Caswell. Bleu might be guilty but references are not innocent. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 61–71, 2020.

[121] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

[122] Yao Fu and Yansong Feng. Natural answer generation with heterogeneous memory. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 185–195, 2018.

[123] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[124] Michel Galley, Chris Brockett, Alessandro Sordoni, Yangfeng Ji, Michael Auli, Chris Quirk, Margaret Mitchell, Jianfeng Gao, and Bill Dolan. deltableu: A discriminative metric for generation tasks with intrinsically diverse targets. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 445–450, 2015.

[125] Jianfeng Gao, Michel Galley, and Lihong Li. *Neural approaches to conversational AI: Question answering, task-oriented dialogues and social chatbots*. Now Foundations and Trends, 2019.

[126] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, 2021.

[127] Yang Gao, Christian M Meyer, Mohsen Mesgar, and Iryna Gurevych. Reward learning for efficient reinforcement learning in extractive document summarisation. *arXiv preprint arXiv:1907.12894*, 2019.

[128] Cristina Garbacea and Qiaozhu Mei. Why is constrained neural language generation particularly challenging? *arXiv preprint arXiv:2206.05395*, 2022.

[129] Albert Gatt and Emiel Krahmer. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170, 2018.

[130] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.

[131] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. Gltr: Statistical detection and visualization of generated text. In *Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL), 2019.

[132] Pablo Gervás. Computational approaches to storytelling and creativity. *AI Magazine*, 30(3):49–49, 2009.

[133] Daniela Gerz, Ivan Vulić, Edoardo Ponti, Jason Naradowsky, Roi Reichart, and Anna Korhonen. Language modeling for morphologically rich languages: Character-aware modeling for word-level prediction. *Transactions of the Association of Computational Linguistics*, 6:451–465, 2018.

[134] Marjan Ghazvininejad, Xing Shi, Yejin Choi, and Kevin Knight. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1191, 2016.

[135] Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, 2017.

[136] Sayan Ghosh, Mathieu Chollet, Eugene Laksana, Louis-Philippe Morency, and Stefan Scherer. Affect-lm: A neural language model for customizable affective text generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 634–642, 2017.

[137] Gyslain Giguère and Bradley C Love. Limits in decision making arise from limits in memory retrieval. *Proceedings of the National Academy of Sciences*, 110(19):7613–7618, 2013.

[138] Kevin Gimpel, Dhruv Batra, Chris Dyer, and Gregory Shakhnarovich. A systematic exploration of diversity in machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1100–1111, 2013.

[139] Dimitra Gkatzia and Saad Mahamood. A snapshot of nlg evaluation practices 2005-2014. In *Proceedings of the 15th European Workshop on Natural Language Generation (ENLG)*, pages 57–60, 2015.

[140] Micah Goldblum, Liam Fowl, and Tom Goldstein. Robust few-shot learning with adversarially queried meta-learners. 2019.

[141] Micah Goldblum, Steven Reich, Liam Fowl, Renkun Ni, Valeriia Cherepanova, and Tom Goldstein. Unraveling meta-learning: Understanding feature representations for few-shot tasks. In *International Conference on Machine Learning*, pages 3607–3616. PMLR, 2020.

[142] Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 89–97, 2019.

[143] Olga Golovneva, Moya Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam Fazel-Zarandi, and Asli Celikyilmaz. Roscoe: A suite of metrics for scoring step-by-step reasoning. *arXiv preprint arXiv:2212.07919*, 2022.

[144] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2022.

[145] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[146] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[147] Yvette Graham and Timothy Baldwin. Testing for significance of increased correlation with human judgment. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 172–176, 2014.

[148] Yvette Graham, Barry Haddow, and Philipp Koehn. Translationese in machine translation evaluation. *arXiv preprint arXiv:1906.09833*, 2019.

[149] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[150] Jiatao Gu, Qi Liu, and Kyunghyun Cho. Insertion-based decoding with automatically inferred generation order. *arXiv preprint arXiv:1902.01370*, 2019.

[151] Jiatao Gu, Changhan Wang, and Jake Zhao. Levenshtein transformer. *arXiv preprint arXiv:1905.11006*, 2019.

[152] Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, 2018.

[153] Han Guo, Ramakanth Pasunuru, and Mohit Bansal. Dynamic multi-level multi-task learning for sentence simplification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 462–476, 2018.

[154] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[155] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, 2020.

[156] Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *Transactions of the Association of Computational Linguistics*, 6:437–450, 2018.

[157] Helena H. Lee, Ke Shu, Palakorn Achananuparp, Philips Kokoh Prasetyo, Yue Liu, Ee-Peng Lim, and Lav R Varshney. Recipegpt: Generative pre-training based cooking recipe generation and evaluation system. In *Companion Proceedings of the Web Conference 2020*, pages 181–184, 2020.

[158] Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. Warp: Word-level adversarial reprogramming. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4921–4933, 2021.

[159] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023.

[160] David Hardcastle and Donia Scott. Can we evaluate the quality of generated text? In *LREC*. Citeseer, 2008.

[161] Donna K Harman and Ellen M Voorhees. Trec: An overview. *Annual review of information science and technology*, 40(1):113–155, 2006.

[162] Tatsunori B Hashimoto, Hugh Zhang, and Percy Liang. Unifying human and statistical evaluation for natural language generation. *arXiv preprint arXiv:1904.02792*, 2019.

[163] Tianxing He, Jingyu Zhang, Tianle Wang, Sachin Kumar, Kyunghyun Cho, James Glass, and Yulia Tsvetkov. On the blind spots of model-based evaluation metrics for text generation. *arXiv preprint arXiv:2212.10020*, 2022.

[164] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[165] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[166] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[167] Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, 2017.

[168] Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, 2018.

[169] Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[170] Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. Instruction induction: From few examples to natural language task descriptions. *arXiv preprint arXiv:2205.10782*, 2022.

[171] Timothy M Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[172] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[173] Dirk Hovy and Shannon L Spruit. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 591–598, 2016.

[174] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

[175] Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

[176] Lee-Hsun Hsieh, Yang-Yin Lee, and Ee-Peng Lim. Enconter: Entity constrained progressive sequence generation via insertion-based transformer. *arXiv preprint arXiv:2103.09548*, 2021.

[177] J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, 2019.

[178] Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org, 2017.

[179] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[180] Ferenc Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

[181] Daphne Ippolito, Reno Kriz, Joao Sedoc, Maria Kustikova, Chris Callison-Burch, Reno Kriz, Eleni Miltsakaki, Marianna Apidianaki, Chris Callison-Burch, John Hewitt, et al. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*. Association for Computational Linguistics, 2018.

[182] Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 633–644, 2014.

[183] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, 2018.

[184] Muhammad Abdullah Jamal, Liqiang Wang, and Boqing Gong. A lazy approach to long-horizon gradient-based meta-learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6577–6586, 2021.

[185] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[186] Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, KIM Gyeonghun, Stanley Jungkyu Choi, and Minjoon Seo. Towards continual knowledge learning of language models. In *International Conference on Learning Representations*, 2021.

[187] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

[188] Ganesh Jawahar, Muhammad Abdul-Mageed, and VS Laks Lakshmanan. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, 2020.

[189] Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. Perplexity?a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63, 1977.

[190] Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. Shakespearizing modern language using copy-enriched sequence-to-sequence models. *EMNLP 2017*, 6:10, 2017.

[191] Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, 2019.

[192] Nitish Joshi and He He. An investigation of the (in) effectiveness of counterfactually augmented data. In *60th Annual Meeting of the Association for Computational Linguistics, ACL 2022*, pages 3668–3681. Association for Computational Linguistics (ACL), 2022.

[193] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[194] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[195] Tomoyuki Kajiwara. Negative lexically constrained decoding for paraphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6047–6052, 2019.

[196] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR, 2023.

[197] Sasikiran Kandula, Dorothy Curtis, and Qing Zeng-Treitler. A semantic and syntactic text simplification tool for health content. In *AMIA annual symposium proceedings*, volume 2010, page 366. American Medical Informatics Association, 2010.

[198] Bingyi Kang and Jiashi Feng. Transferable meta learning across domains. In *UAI*, pages 177–187, 2018.

[199] Anjuli Kannan and Oriol Vinyals. Adversarial evaluation of dialogue models. *arXiv preprint arXiv:1701.08198*, 2017.

[200] Jungo Kasai, Keisuke Sakaguchi, Lavinia Dunagan, Jacob Morrison, Ronan Le Bras, Yejin Choi, and Noah A Smith. Transparent human evaluation for image captioning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3464–3478, 2022.

[201] Marvin Kaster, Wei Zhao, and Steffen Eger. Global explainability of bert-based evaluation metrics by disentangling along linguistic factors. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8912–8925, 2021.

[202] David Kauchak. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 1537–1546, 2013.

[203] Pei Ke, Hao Zhou, Yankai Lin, Peng Li, Jie Zhou, Xiaoyan Zhu, and Minlie Huang. Ctrleval: An unsupervised reference-free metric for evaluating controlled text generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2306–2319, 2022.

[204] Zachary Kenton, Tom Everitt, Laura Weidinger, Iason Gabriel, Vladimir Mikulik, and Geoffrey Irving. Alignment of language agents. *arXiv preprint arXiv:2103.14659*, 2021.

[205] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[206] Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. A distributional approach to controlled text generation. *arXiv preprint arXiv:2012.11635*, 2020.

[207] Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. Discriminator-guided multi-step reasoning with language models. *arXiv preprint arXiv:2305.14934*, 2023.

[208] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

[209] Mitesh M Khapra and Ananya B Sai. A tutorial on evaluation metrics used in natural language generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Tutorials*, pages 15–19, 2021.

[210] Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A Smith, and Daniel S Weld. Genie: Toward reproducible and standardized human evaluation for text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11444–11458, 2022.

[211] Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Controlling output length in neural encoder-decoders. In *EMNLP*, 2016.

[212] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[213] Yoon Kim, Kelly Zhang, Alexander M Rush, Yann LeCun, et al. Adversarially regularized autoencoders for generating discrete structures. *arXiv preprint arXiv:1706.04223*, 2:12, 2017.

[214] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. 1975.

[215] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[216] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[217] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. *arXiv preprint arXiv:2301.10226*, 2023.

[218] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[219] Yuval Kirstain, Patrick Lewis, Sebastian Riedel, and Omer Levy. A few more examples may be worth billions of parameters. *arXiv preprint arXiv:2110.04374*, 2021.

[220] Rebecca Knowles and Philipp Koehn. Neural interactive translation prediction. In *Twelfth Conference of The Association for Machine Translation in the Americas*, pages 107–120. Association for Machine Translation in the Americas, AMTA, 2016.

[221] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[222] Xiang Kong, Bohan Li, Graham Neubig, Eduard Hovy, and Yiming Yang. An adversarial approach to high-quality, sentiment-controlled neural dialogue generation. *arXiv preprint arXiv:1901.07129*, 2019.

[223] Bernhard Kratzwald, Anna Eigenmann, and Stefan Feuerriegel. Rankqa: Neural question answering with answer re-ranking. *arXiv preprint arXiv:1906.03008*, 2019.

[224] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*, 2020.

[225] Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, 2018.

[226] Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, 2019.

[227] Victor Kuperman, Hans Stadthagen-Gonzalez, and Marc Brysbaert. Age-of-acquisition ratings for 30,000 english words. *Behavior research methods*, 44(4):978–990, 2012.

[228] Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, 2019.

[229] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR, 2015.

[230] Philippe Laban, Andrew Hsi, John Canny, and Marti A Hearst. The summary loop: Learning to write abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, volume 1, 2020.

[231] Philippe Laban, Wojciech Kryściński, Divyansh Agarwal, Alexander R Fabbri, Caiming Xiong, Shafiq Joty, and Chien-Sheng Wu. Llms as factual reasoners: Insights from existing benchmarks and beyond. *arXiv preprint arXiv:2305.14540*, 2023.

[232] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 131–138, 2019.

[233] Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609, 2016.

[234] Andrew Lampinen, Ishita Dasgupta, Stephanie Chan, Kory Mathewson, Mh Tessler, Antonia Creswell, James McClelland, Jane Wang, and Felix Hill. Can language models learn from explanations in context? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 537–563, 2022.

[235] Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. Multiple-attribute text rewriting. In *International Conference on Learning Representations*, 2018.

[236] Seemab Latif, Sarmad Bashir, Mir Muntasar Ali Agha, and Rabia Latif. Backward-forward sequence generative network for multiple lexical constraints. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 39–50. Springer, 2020.

[237] Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, et al. Mind the gap: Assessing temporal generalization in neural language models. *Advances in Neural Information Processing Systems*, 34:29348–29363, 2021.

[238] Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Sebastian Ruder, Dani Yogatama, et al. Pitfalls of static language modelling. *arXiv preprint arXiv:2102.01951*, 2021.

[239] Teven Le Scao and Alexander M Rush. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, 2021.

[240] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[241] Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, 2018.

[242] Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, 2016.

[243] Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

[244] Christoph Leiter, Piyawat Lertvittayakumjorn, Marina Fomicheva, Wei Zhao, Yang Gao, and Steffen Eger. Towards explainable evaluation metrics for natural language generation. *arXiv preprint arXiv:2203.11131*, 2022.

[245] Gondy Leroy, James E Endicott, David Kauchak, Obay Mouradi, and Melissa Just. User evaluation of the effects of a text simplification algorithm using term familiarity on perception, understanding, learning, and information retention. *Journal of medical Internet research*, 15(7):e144, 2013.

[246] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[247] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[248] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[249] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[250] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

[251] Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*, 2020.

[252] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[253] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pages 110–119, 2016.

[254] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and William B Dolan. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, 2016.

[255] Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, 2016.

[256] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, 2017.

[257] Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1865–1874, 2018.

[258] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

[259] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, 2021.

[260] Zekun Li, Baolin Peng, Pengcheng He, Michel Galley, Jianfeng Gao, and Xifeng Yan. Guiding large language models via directional stimulus prompting. *arXiv preprint arXiv:2302.11520*, 2023.

[261] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

[262] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[263] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[264] Bill Yuchen Lin, Ming Shen, Yu Xing, Pei Zhou, and Xiang Ren. Commongen: A constrained text generation dataset towards generative commonsense reasoning. *arXiv preprint arXiv:1911.03705*, 2019.

[265] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[266] Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. Adversarial ranking for language generation. In *Advances in Neural Information Processing Systems*, pages 3155–3165, 2017.

[267] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, 2022.

[268] Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. Deductive verification of chain-of-thought reasoning. *arXiv preprint arXiv:2306.03872*, 2023.

[269] Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. Dexperts: Decoding-time controlled text generation with experts and anti-experts.

[270] Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, 2016.

[271] Dayiheng Liu, Jie Fu, Qian Qu, and Jiancheng Lv. Bfgan: Backward and forward generative adversarial networks for lexically constrained sentence generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):2350–2361, 2019.

[272] Jiongnan Liu, Jiajie Jin, Zihan Wang, Jiehan Cheng, Zhicheng Dou, and Ji-Rong Wen. Retallm: A retrieval-augmented large language model toolkit. *arXiv preprint arXiv:2306.05212*, 2023.

[273] Nelson Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

[274] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[275] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*, 2018.

[276] Xin Liu, Muhammad Khalifa, and Lu Wang. Bolt: Fast energy-based controlled text generation with tunable biases. *arXiv preprint arXiv:2305.12018*, 2023.

[277] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*, 2023.

[278] Yu Lu Liu, Rachel Bawden, Thomas Scaliom, Benoît Sagot, and Jackie Chi Kit Cheung. Maskeval: Weighted mlm-based evaluation for text summarization and simplification. *arXiv preprint arXiv:2205.12394*, 2022.

[279] Zhiqiang Liu, Zuohui Fu, Jie Cao, Gerard de Melo, Yik-Cheung Tam, Cheng Niu, and Jie Zhou. Rhetorically controlled encoder-decoder for modern chinese poetry generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1992–2001, 2019.

[280] Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*, pages 5103–5113, 2018.

[281] Ekaterina Loginova, Stalin Varanasi, and Günter Neumann. Towards multilingual neural question answering. In *European Conference on Advances in Databases and Information Systems*, pages 274–285. Springer, 2018.

[282] Tania Lombrozo. The structure and function of explanations. *Trends in cognitive sciences*, 10(10):464–470, 2006.

[283] Adam Lopez. Putting human assessments of machine translation systems in order. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 1–9. Association for Computational Linguistics, 2012.

[284] Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. Towards an automatic turing test: Learning to evaluate dialogue responses. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1116–1126, 2017.

[285] Sidi Lu, Yaoming Zhu, Weinan Zhang, Jun Wang, and Yong Yu. Neural text generation: Past, present and beyond. *arXiv preprint arXiv:1803.07133*, 2018.

[286] Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Neurologic decoding:(un) supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, 2021.

[287] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, 2022.

[288] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. *CoRR, abs/1703.07511*, 2017.

[289] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*, 2016.

[290] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *Advances in neural information processing systems*, pages 700–709, 2018.

[291] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[292] Yiwei Lyu, Paul Pu Liang, Hai Pham, Eduard Hovy, Barnabás Póczos, Ruslan Salakhutdinov, and Louis-Philippe Morency. Styleptb: A compositional benchmark for fine-grained controllable text style transfer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2116–2138, 2021.

[293] Computing Machinery. Computing machinery and intelligence-am turing. *Mind*, 59(236):433, 1950.

[294] Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabás Póczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. Politeness transfer: A tag and generate approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1869–1881, 2020.

[295] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023.

[296] Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. Plug-and-play conversational models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2422–2433, 2020.

[297] Inbal Magar and Roy Schwartz. Data contamination: From memorization to exploitation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–165, 2022.

[298] Louis Martin, Éric Villemonte de la Clergerie, Benoît Sagot, and Antoine Bordes. Controllable sentence simplification. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4689–4698, 2020.

[299] Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. Hatexplain: A benchmark dataset for explainable hate speech detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 14867–14875, 2021.

[300] Nitika Mathur, Tim Baldwin, and Trevor Cohn. Tangled up in bleu: Reevaluating the evaluation of automatic machine translation evaluation metrics. *arXiv preprint arXiv:2006.06264*, 2020.

[301] Nitika Mathur, Timothy Baldwin, and Trevor Cohn. Putting evaluation in context: Contextual embeddings improve machine translation evaluation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2799–2808, 2019.

[302] Elijah Mayfield, Michael Madaio, Shrimai Prabhumoye, David Gerritsen, Brittany McLaughlin, Ezekiel Dixon-Román, and Alan W Black. Equity beyond bias in language technologies for education. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 444–460, 2019.

[303] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, 2020.

[304] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35, 2021.

[305] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. Coherent dialogue with attention-based language models. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[306] Chris Mellish and Robert Dale. Evaluation in the context of natural language generation. *Computer Speech & Language*, 12(4):349–373, 1998.

[307] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842, 2019.

[308] Yishu Miao and Phil Blunsom. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 319–328, 2016.

[309] Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Extensions of recurrent neural network language model. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 5528–5531. IEEE, 2011.

[310] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201, 2011.

[311] Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. *SLT*, 12:234–239, 2012.

[312] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.

[313] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018.

[314] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*, 2023.

[315] Ehsan Montahaei, Danial Alihosseini, and Mahdieh Soleymani Baghshah. Jointly measuring diversity and quality in text generation models. *arXiv preprint arXiv:1904.03971*, 2019.

[316] Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. *arXiv preprint arXiv:1607.00970*, 2016.

[317] Lili Mou, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. Backward and forward language modeling for constrained sentence generation. *arXiv preprint arXiv:1512.06612*, 2015.

[318] Jonas Mueller, David Gifford, and Tommi Jaakkola. Sequence to better sequence: continuous revision of combinatorial structures. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2536–2544. JMLR. org, 2017.

[319] Tsendsuren Munkhdalai, Xingdi Yuan, Soroush Mehri, and Adam Trischler. Rapid adaptation with conditionally shifted neurons. In *International Conference on Machine Learning*, pages 3664–3673. PMLR, 2018.

[320] Vaishnavh Nagarajan, Colin Raffel, and Ian J. Goodfellow. Theoretical Insights into Memorization in GANs.

[321] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[322] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[323] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, 2015.

[324] Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. Evaluating sentence compression: Pitfalls and suggested remedies. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97, 2011.

[325] Preksha Nema and Mitesh M Khapra. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, 2018.

[326] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[327] Nikola I Nikolov and Richard Hahnloser. Abstractive document summarization without parallel data. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6638–6644, 2020.

[328] Daiki Nishihara, Tomoyuki Kajiwara, and Yuki Arase. Controllable text simplification with lexical constraint loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 260–266, 2019.

[329] Tong Niu and Mohit Bansal. Polite dialogue generation without parallel data. *Transactions of the Association for Computational Linguistics*, 6:373–389, 2018.

[330] Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

[331] Hugo Gonçalo Oliveira. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 11–20, 2017.

[332] Luca Oneto and Silvia Chiappa. Fairness in machine learning. In *Recent trends in learning from data: Tutorials from the inns big data and deep learning conference (innsbddl2019)*, pages 155–196. Springer, 2020.

[333] OpenAI. Introducing chatgpt. *https://openai.com/blog/chatgpt*, 2022.

[334] OpenAI. Gpt-4 technical report, 2023.

[335] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in neural information processing systems*, 31, 2018.

[336] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 309–319. Association for Computational Linguistics, 2011.

[337] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL-HLT (Demonstrations)*, 2019.

[338] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[339] Gustavo Paetzold. Reliable lexical simplification for non-native speakers. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 9–16, 2015.

[340] Gustavo Paetzold and Lucia Specia. Semeval 2016 task 11: Complex word identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 560–569, 2016.

[341] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of NAACL-HLT*, pages 528–540, 2018.

[342] Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, 2021.

[343] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[344] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519. ACM, 2017.

[345] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[346] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

[347] Ramakanth Pasunuru, Han Guo, and Mohit Bansal. Dorb: Dynamically optimizing multiple rewards with bandits. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7766–7780, 2020.

[348] Kaustubh R Patil, Jerry Zhu, Łukasz Kopeć, and Bradley C Love. Optimal teaching for limited-capacity human learners. *Advances in neural information processing systems*, 27, 2014.

[349] David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

[350] Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*, 2023.

[351] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[352] Ellie Pavlick and Chris Callison-Burch. Simple ppdb: A paraphrase database for simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 143–148, 2016.

[353] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*, 2023.

[354] Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, 2018.

[355] Maxime Peyrard. Studying summarization evaluation metrics in the appropriate scoring range. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5093–5100, 2019.

[356] Maxime Peyrard, Teresa Botschen, and Iryna Gurevych. Learning to score system summaries for better content selection evaluation. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 74–84, 2017.

[357] Thang Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1145–1160, 2021.

[358] Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018.

[359] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34, 2021.

[360] Maja Popović. On reducing translation shifts in translations intended for mt evaluation. In *Proceedings of Machine Translation Summit XVII: Translator, Project and User Tracks*, pages 80–87, 2019.

[361] Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, 2018.

[362] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

[363] Mark A. Przybocki, Kay Peterson, Sebastien Bronsart, and Gregory A. Sanders. The NIST 2008 metrics for machine translation challenge - overview, methodology, metrics, and results. *Machine Translation*, 23(2-3):71–103, 2009.

[364] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2021.

[365] Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5046–5056, 2019.

[366] Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35:9538–9551, 2022.

[367] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*, 2018.

[368] Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications. *OpenAI blog*, 1(2), 2019.

[369] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:8, 2019.

[370] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[371] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

[372] Aniruddh Raghu, Jonathan Lorraine, Simon Kornblith, Matthew McDermott, and David K Duvenaud. Meta-learning to improve pre-training. *Advances in Neural Information Processing Systems*, 34, 2021.

[373] Sai Rajeswar, Sandeep Subramanian, Francis Dutil, Christopher Pal, and Aaron Courville. Adversarial generation of natural language. *arXiv preprint arXiv:1705.10929*, 2017.

[374] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019.

[375] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[376] Sudha Rao and Joel Tetreault. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 129–140, 2018.

[377] Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot reasoning. *arXiv preprint arXiv:2202.07206*, 2022.

[378] Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. Comet: A neural framework for mt evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, 2020.

[379] Ehud Reiter and Anja Belz. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558, 2009.

[380] Ehud Reiter, Roma Robertson, A Scott Lennox, and Liesl Osman. Using a randomised controlled clinical trial to evaluate an nlg system. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 442–449. Association for Computational Linguistics, 2001.

[381] Luz Rello, Ricardo Baeza-Yates, Stefan Bott, and Horacio Saggion. Simplify or help?: text simplification strategies for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*. ACM, 2013.

[382] Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2021.

[383] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

[384] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020.

[385] Irina Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM, 2001.

[386] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[387] Ananya B Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M Khapra. Perturbation checklists for evaluating nlg evaluation metrics. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7219–7234, 2021.

[388] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, pages 5228–5237, 2018.

[389] Abdelrhman Saleh, Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, and Rosalind Picard. Hierarchical reinforcement learning for open-domain dialog. *arXiv preprint arXiv:1909.07547*, 2019.

[390] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[391] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. In *ICLR 2022-Tenth International Conference on Learning Representations*, 2022.

[392] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[393] Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240*, 2022.

[394] Carolina Scarton and Lucia Specia. Learning simplifications for specific target audiences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 712–718, 2018.

[395] Jérémy Scheurer, Jon Ander Campos, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training language models with natural language feedback.

[396] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, 2021.

[397] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[398] Jürgen Schmidhuber and Sepp Hochreiter. Long short-term memory. *Neural Comput*, 9(8):1735–1780, 1997.

[399] Stephanie Schoch, Diyi Yang, and Yangfeng Ji. "this is a problem, don't you agree?" framing and bias in human evaluation for natural language generation. In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, pages 10–16, 2020.

[400] Moritz Schubotz, Philipp Scharpf, Kaushal Dudhat, Yash Nagar, Felix Hamborg, and Bela Gipp. Introducing mathqa: a math-aware question answering system. *Information Discovery and Delivery*, 2018.

[401] Elliot Schumacher, Maxine Eskenazi, Gwen Frishkoff, and Kevyn Collins-Thompson. Predicting the relative difficulty of single sentences with and without surrounding context. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1871–1881, 2016.

[402] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017.

[403] Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. What makes a good conversation? how controllable attributes affect human judgments. In *Proceedings of NAACL-HLT*, pages 1702–1723, 2019.

[404] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020.

[405] Stanislau Semeniuta, Aliaksei Severyn, and Sylvain Gelly. On accurate evaluation of gans for language generation. *arXiv preprint arXiv:1806.04936*, 2018.

[406] Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, 2016.

[407] Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[408] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*, 2015.

[409] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[410] Lei Sha. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703, 2020.

[411] Darsh J Shah, Tal Schuster, and Regina Barzilay. Automatic fact-guided sentence modification. In *AAAI*, pages 8791–8798, 2020.

[412] Amr Sharaf, Hany Hassan, and Hal Daumé III. Meta-learning for few-shot nmt adaptation. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 43–53, 2020.

[413] Matthew Shardlow. A comparison of techniques to automatically identify complex words. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*, pages 103–109, 2013.

[414] Matthew Shardlow. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, 4(1):58–70, 2014.

[415] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv preprint arXiv:1706.09799*, 2017.

[416] Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841, 2017.

[417] Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, 2019.

[418] Emily Sheng, Kai-Wei Chang, Prem Natarajan, and Nanyun Peng. Towards controllable biases in language generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 3239–3254, 2020.

[419] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*, 2023.

[420] Weiyan Shi, Ryan Shea, Si Chen, Chiyuan Zhang, Ruoxi Jia, and Zhou Yu. Just fine-tune twice: Selective differential privacy for large language models. *arXiv preprint arXiv:2204.07667*, 2022.

[421] Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Toward diverse text generation with inverse reinforcement learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4361–4367. AAAI Press, 2018.

[422] Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Toward diverse text generation with inverse reinforcement learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4361–4367. AAAI Press, 2018.

[423] Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. Ruse: Regressor using sentence embeddings for automatic machine translation evaluation. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 751–758, 2018.

[424] Seongjin Shin, Sang-Woo Lee, Hwijeen Ahn, Sungdong Kim, HyoungSeok Kim, Boseop Kim, Kyunghyun Cho, Gichang Lee, Woomyoung Park, Jung-Woo Ha, et al. On the effect of pretraining corpora on in-context learning by a large-scale language model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5168–5186, 2022.

[425] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, 2020.

[426] Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. Prompting gpt-3 to be reliable. *arXiv preprint arXiv:2210.09150*, 2022.

[427] Advaith Siddharthan. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4(1):77–109, 2006.

[428] Koustuv Sinha, Prasanna Parthasarathi, Jasmine Wang, Ryan Lowe, William L Hamilton, and Joelle Pineau. Learning an unreferenced metric for online dialogue evaluation. *arXiv preprint arXiv:2005.00583*, 2020.

[429] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[430] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200, 2006.

[431] Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*, 2019.

[432] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi and weakly supervised semantic segmentation using generative adversarial network. *arXiv preprint arXiv:1703.09695*, 2017.

[433] Charles Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.

[434] Akash Srivastava, Lazar Valkoz, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pages 3310–3320, 2017.

[435] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[436] Sanja Štajner and Ioana Hulpuș. When shallow is good enough: Automatic assessment of conceptual text complexity using shallow semantic features. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 1414–1422, 2020.

[437] Amanda Stent, Matthew Marge, and Mohit Singhai. Evaluating evaluation methods for generation in the presence of variation. In *international conference on intelligent text processing and computational linguistics*, pages 341–351. Springer, 2005.

[438] Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*, pages 5976–5985, 2019.

[439] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

[440] Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. Transforming delete, retrieve, generate approach for controlled text style transfer. *arXiv preprint arXiv:1908.09368*, 2019.

[441] Elior Sulem, Omri Abend, and Ari Rappoport. Bleu is not suitable for the evaluation of text simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 738–744, 2018.

[442] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 403–412, 2019.

[443] Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. How to compare summarizers without target length? pitfalls, solutions and re-examination of the neural summarization literature. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29, 2019.

[444] Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. Principle-driven self-alignment of language models from scratch with minimal human supervision. *arXiv preprint arXiv:2305.03047*, 2023.

[445] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[446] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[447] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*, 2016.

[448] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.

[449] Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. Evaluating the factual consistency of large language models through summarization. *arXiv preprint arXiv:2211.08412*, 2022.

[450] Jian Tang, Yifan Yang, Sam Carton, Ming Zhang, and Qiaozhu Mei. Context-aware natural language generation with recurrent neural networks. *arXiv preprint arXiv:1611.09900*, 2016.

[451] Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634, 2019.

[452] Zecheng Tang, Pinzheng Wang, Keyan Zhou, Juntao Li, Ziqiang Cao, and Min Zhang. Can diffusion model achieve better performance in text generation? bridging the gap between training and inference! *arXiv preprint arXiv:2305.04465*, 2023.

[453] Guy Tevet and Jonathan Berant. Evaluating the evaluation of diversity in natural language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 326–346, 2021.

[454] Guy Tevet, Gavriel Habib, Vered Shwartz, and Jonathan Berant. Evaluating text gans as language models. *arXiv preprint arXiv:1810.12686*, 2018.

[455] L Theis, A van den Oord, and M Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR 2016)*, pages 1–10, 2016.

[456] L Theis, A van den Oord, and M Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR 2016)*, pages 1–10, 2016.

[457] Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2062–2068, 2019.

[458] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[459] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

[460] Laurens Van den Bercken, Robert-Jan Sips, and Christoph Lofi. Evaluating neural text simplification in the medical domain. In *The World Wide Web Conference*, pages 3286–3292. ACM, 2019.

[461] Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, 2019.

[462] Judith van Stegeren and Mariët Theune. Narrative generation in the wild: Methods from nanogenmo. In *Proceedings of the Second Workshop on Storytelling*, pages 65–74, 2019.

[463] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[464] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[465] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

[466] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[467] Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. Exploring and predicting transferability across nlp tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, 2020.

[468] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[469] Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. Towards understanding chain-of-thought prompting: An empirical study of what matters. *arXiv preprint arXiv:2212.10001*, 2022.

[470] Chengyu Wang, Minghui Qiu, Jun Huang, and Xiaofeng He. Meta fine-tuning neural language models for multi-domain text mining. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3094–3104, 2020.

[471] Di Wang, Nebojsa Jojic, Chris Brockett, and Eric Nyberg. Steering output style and topic in neural response generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2140–2150, 2017.

[472] Hong Wang, Xuan Luo, Weizhi Wang, and Xifeng Yan. Bot or human? detecting chatgpt imposters with a single question. *arXiv preprint arXiv:2305.06424*, 2023.

[473] Jiaan Wang, Yunlong Liang, Fandong Meng, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*, 2023.

[474] Wenlin Wang, Zhe Gan, Hongteng Xu, Ruiyi Zhang, Guoyin Wang, Dinghan Shen, Changyou Chen, and Lawrence Carin. Topic-guided variational auto-encoder for text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 166–177, 2019.

[475] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[476] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

[477] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[478] Sean Welleck, Kianté Brantley, Hal Daumé III, and Kyunghyun Cho. Non-monotonic sequential text generation. *arXiv preprint arXiv:1902.02192*, 2019.

[479] Georg Wiese, Dirk Weissenborn, and Mariana Neves. Neural domain adaptation for biomedical question answering. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 281–289, 2017.

[480] John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. Beyond bleu: Training neural machine translation with semantic similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, 2019.

[481] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[482] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[483] Genta Indra Winata, Andrea Madotto, Zhaojiang Lin, Rosanne Liu, Jason Yosinski, and Pascale Fung. Language models are few-shot multilingual learners. *arXiv preprint arXiv:2109.07684*, 2021.

[484] Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

[485] Kristian Woodsend and Mirella Lapata. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the conference on empirical methods in natural language processing*, pages 409–420. Association for Computational Linguistics, 2011.

[486] Danny TY Wu, David A Hanauer, Qiaozhu Mei, Patricia M Clark, Lawrence C An, Joshua Proulx, Qing T Zeng, VG Vinod Vydiswaran, Kevyn Collins-Thompson, and Kai Zheng. Assessing the readability of clinicaltrials. gov. *Journal of the American Medical Informatics Association*, 23(2):269–275, 2016.

[487] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. Errudite: Scalable, reproducible, and testable error analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 747–763, 2019.

[488] Yuxiang Wu and Baotian Hu. Learning to extract coherent summary via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[489] Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv preprint arXiv:2307.02477*, 2023.

[490] Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 1015–1024. Association for Computational Linguistics, 2012.

[491] Joern Wuebker, Spence Green, John DeNero, Saša Hasan, and Minh-Thang Luong. Models and inference for prefix-constrained machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, 2016.

[492] Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. Language models meet world models: Embodied experiences enhance language models. *arXiv preprint arXiv:2305.10626*, 2023.

[493] Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie. Decomposition enhances reasoning via self-evaluation guided decoding. *arXiv preprint arXiv:2305.00633*, 2023.

[494] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. Topic augmented neural response generation with a joint attention mechanism. *arXiv preprint arXiv:1606.08340*, 2(2), 2016.

[495] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[496] Wei Xu, Chris Callison-Burch, and Courtney Napoles. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297, 2015.

[497] Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415, 2016.

[498] Ying Xu, Xu Zhong, Antonio Jose Jimeno Yepes, and Jey Han Lau. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[499] Kevin Yang and Dan Klein. Fudge: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, 2021.

[500] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*, 2022.

[501] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32:5753–5763, 2019.

[502] Denis Yarats and Mike Lewis. Hierarchical text generation and planning for strategic dialogue. In *International Conference on Machine Learning*, pages 5587–5595, 2018.

[503] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[504] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2021.

[505] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.

[506] Weizhe Yuan, Graham Neubig, and Pengfei Liu. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34, 2021.

[507] Rowan Zellers, Ari Holtzman, Elizabeth Clark, Lianhui Qin, Ali Farhadi, and Yejin Choi. Turingadvice: A generative and dynamic evaluation of language use. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021.

[508] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. Defending against neural fake news. *Advances in neural information processing systems*, 32, 2019.

[509] Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, 2016.

[510] Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*, 2022.

[511] Michael Zhang and Eunsol Choi. Situatedqa: Incorporating extra-linguistic contexts into qa. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387, 2021.

[512] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. *Advances in neural information processing systems*, 31, 2018.

[513] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, 2018.

[514] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[515] Xingxing Zhang and Mirella Lapata. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680, 2014.

[516] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, 2017.

[517] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, 2017.

[518] Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. Generating informative and diverse conversational responses via adversarial information maximization. In *Advances in Neural Information Processing Systems*, pages 1810–1820, 2018.

[519] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 4006–4015. JMLR. org, 2017.

[520] Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. Pointer: Constrained text generation via insertion-based generative pre-training. *arXiv preprint arXiv:2005.00558*, 2020.

[521] Zhirui Zhang, Shuo Ren, Shujie Liu, Jianyong Wang, Peng Chen, Mu Li, Ming Zhou, and Enhong Chen. Style transfer as unsupervised machine translation. *arXiv preprint arXiv:1808.07894*, 2018.

[522] Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M Rush, and Yann LeCun. Adversarially regularized autoencoders. In *35th International Conference on Machine Learning, ICML 2018*, pages 9405–9420. International Machine Learning Society (IMLS), 2018.

[523] Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M Meyer, and Steffen Eger. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 563–578, 2019.

[524] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021.

[525] Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*, 2019.

[526] Chunting Zhou, Junxian He, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Prompt consistency for zero-shot task generalization. *arXiv preprint arXiv:2205.00049*, 2022.

[527] Chunting Zhou and Graham Neubig. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 310–320, 2017.

[528] Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, 2021.

[529] Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[530] Wangchunshu Zhou, Ronan Le Bras, and Yejin Choi. Commonsense knowledge transfer for pre-trained language models. *arXiv preprint arXiv:2306.02388*, 2023.

[531] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

[532] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[533] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[534] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100. ACM, 2018.

[535] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*, pages 1353–1361. Association for Computational Linguistics, 2010.

[536] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[537] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[538] Xu Zou, Da Yin, Qingyang Zhong, Hongxia Yang, Zhilin Yang, and Jie Tang. Controllable generation from pre-trained language models via inverse prompting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2450–2460, 2021.