

Leveraging Longitudinal Data for Personalized Prediction and Word Representations

by

Charles Welch

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
2021

Doctoral Committee:

Professor Rada Mihalcea, Chair
Professor Joyce Chai
Professor David Jurgens
Professor Ellie Pavlick

Charles Welch

cfwelch@umich.edu

ORCID iD: [0000-0002-3489-2882](https://orcid.org/0000-0002-3489-2882)

© Charles Welch 2021

To my parents, Robert and Cindy.

ACKNOWLEDGMENTS

I have many people to thank for how they have helped or supported me in writing this thesis. For supporting me no matter what, I thank my parents, Robert and Cindy. For helping keep me sane, I thank my friends and family, Riikka, Michal, Elliot, Sean, Ben, Veronica, Jacob, Sherman, Reed, Arjun, Leah, Zak, Miriam, Erin, John, and many many others. For helpful discussions and being part of a welcoming community, I thank my colleagues Maria, Harry, Kostas, Steve, Vero, Shibu, Aparna, Mohamed, Mahmoud, both Lauras, Chenxi, Siqi, Do June, Andrew, Ash, Allie, Oana, MeiXing, Santiago, Artem, Yiming, Yumou, Felix, Max, Mohini, Hui, Berrin, Yiqun, Ian, Michalis, Jonathan, and others who have worked as part of the LIT lab at the University of Michigan. Vero and Jonathan deserve extra thanks, as Vero has helped advise most of the work I was involved with in the past few years of my PhD, and Jonathan has helped advise me for most of my time in graduate school. I thank my committee, Joyce, David, and Ellie, for their helpful advice and Rada, who has been an amazing advisor, from helping me get accepted into the program to helping me pursue my research and teaching interests.

This material is based in part upon work supported by IBM under contract 4915012629 (Sapphire Project), by the Michigan Institute for Data Science, by the National Science Foundation (grant #1815291), by the John Templeton Foundation (grant #61156), and by DARPA (grants #HR001117S0026-AIDA-FP-045 and #D19AP00079).

TABLE OF CONTENTS

Dedication	ii
Acknowledgments	iii
List of Figures	vii
List of Tables	ix
Abstract	xii
Chapter	
1 Introduction	1
1.1 Research Questions	3
1.2 Thesis Outline	4
2 Related Work	6
2.1 Sentiment Analysis	6
2.2 Discourse and Dialog	8
2.3 Language Modeling	9
2.4 Word Embeddings	11
2.4.1 Learning Embeddings	11
2.4.2 Applying Embeddings	13
2.4.3 Embedding Bias	14
2.4.4 User Embeddings	14
2.5 Authorship Attribution	15
2.5.1 Author Profiling	15
2.6 Social Media	16
3 Individual Preferences	17
3.1 Introduction	17
3.2 Dataset	18
3.3 Targeted Sentiment Analysis	20
3.3.1 Entity Extraction	20
3.3.2 Entity-Centric Sentiment Analysis	23
3.4 Overall Evaluation and Discussion	25
3.5 Conclusions	28
4 Conversational Behavior	29

4.1	Introduction	29
4.2	Dataset	30
	4.2.1 Message and Speaker Distributions	32
	4.2.2 Message Production Across Time	33
4.3	Model	34
4.4	Features	35
4.5	Predicting Conversational Behavior	36
	4.5.1 Prediction of Next Message in the Conversation	36
	4.5.2 Prediction of Message Response Time	37
	4.5.3 Results	38
4.6	Deeper Dive into Personal Longitudinal Dialog Data	39
	4.6.1 Language Usage Patterns	40
4.7	Conclusions	41
5	Relationship Between Interlocutors	43
5.1	Introduction	43
5.2	Dataset	44
5.3	Message Content	45
5.4	Groups Over Time	47
5.5	Conversation Interaction	47
5.6	Model and Features	49
5.7	Experiments	51
5.8	Results	51
5.9	Conclusions	54
6	Personalized Word Representations	55
6.1	Introduction	55
6.2	Personalized Word Embeddings	57
6.3	Differences Across Individual Word Representations and Usages	58
6.4	Low Compute Language Modeling with In-Domain Initialization	60
6.5	When and Why Does Pretraining Help?	63
6.6	Language Modeling with Personalized Embeddings	70
6.7	Authorship Attribution	71
6.8	Limitations and Ethical Considerations	73
6.9	Conclusion	73
7	Personalization with Limited Data	75
7.1	Introduction	75
7.2	Demographic Embedding Models	77
7.3	Dataset	77
	7.3.1 Finding Demographic Information	78
	7.3.2 Preprocessing	80
	7.3.3 Post-processing	80
7.4	Generating Compositional Demographic Word Embeddings	84
	7.4.1 Demographic Attribute Vectors	84

7.4.2	Demographic Word Matrices	84
7.5	Language Modeling	85
7.5.1	Demographic Perplexity Evaluation	86
7.5.2	Comparison with User Representations	88
7.6	Demographic Word Associations	89
7.7	Experiments Without Demographic Information	92
7.7.1	Calculating User Similarity	92
7.8	Leveraging Similar Users	94
7.8.1	Weighted Sample Fine-tuning	94
7.8.2	Interpolation Model	94
7.9	Results	96
7.9.1	Small Anchor Set	96
7.9.2	Large Anchor Set	98
7.10	Analysis of Similarity and Personalized Words	101
7.10.1	Differences in Similarity Functions	101
7.10.2	Personalized Words	101
7.11	Limitations and Ethical Considerations	102
7.12	Conclusions	104
8	Conclusions	105
8.1	Revisiting the Research Questions	105
8.2	Future Work	107
8.3	Final Remarks	109
	Bibliography	110

LIST OF FIGURES

3.1	Example sentence, “I thought that CS 203 was going to be good, but it was awful”, showing the parse tree weighting for counts using the number of node hops between a given word and the target entity.	24
4.1	Distribution of number of messages and tokens between the (P) participants and their conversation partners (O) in our dataset.	31
4.2	Shared attributes between participants and their conversation partners	31
4.3	Distribution of messages over time. Months are grouped by season. Totals per season are listed in the inner circles with bars from 85k to 115k messages.	33
4.4	The model architecture encodes a context window as a sequence of tokens w_1 to w_n using a BiLSTM. The encoding is then used in combination with our other feature sets for classification by taking the softmax over q . In the case of speaker attribute decoding, separate classifier is used for each speaker attribute.	37
4.5	Author’s positive and negative emotion words over time shows an increase in positivity from 2012 to the end of 2017.	39
4.6	Normalized LIWC count differences between groups for the cognitive, emotion and social word categories. Group differences are represented by rows. The color scale shows larger differences in word percentages with lighter colors representing a higher proportion in the first group and darker colors representing a higher proportion in the second (using the groups defined in Section 4.2). Statistically significant differences are shown with * for $p < 0.1$, ** for $p < 0.05$, and *** for $p < 0.01$. Significance is tested using an unpaired t-test over the set of people with Holm-Bonferroni correction [70].	40
5.1	Distribution of messages over time. The top shows the distribution over the day of the week and the bottom shows hour of the day. The groups shown are those that vary the most from the aggregate trend over all speakers.	48
5.2	Language mirroring as a function of the number of messages exchanged within groups. The left figure shows the mirroring cumulatively over the first 5,000 messages averaged over people in each of the listed groups. The right figure shows these numbers averaged over a sliding window with the shaded region representing the interquartile range.	49

5.3	Speaker references for the top 20 conversation partners. The graph shows interactions with people from different groups: high school (rectangles), college (triangles), graduate school (rounded rectangles), family members (circles), and other people (ellipses). Shading is proportional to how long ago the author met the person. Edges below a threshold of 25 mentions are removed. Note that the clustering uses all 104 people, but only 20 are shown here.	50
6.1	Relationship between embedding similarity and distribution of word categories across most dissimilar words showing the five grouped POS tags, nouns (NN, NNS, NNP, NNPS), adjectives (JJ, JJR, JJS), verbs (VBD, VBG, VBN, VBP, VBZ, VB), adverbs (RB, RBR, RBS), prepositions and subordinating conjunctions (IN), with the highest frequency and the four LIWC categories with the highest concentration in the set of dissimilar words (i.e. function, relativity, social, and cognitive process (CogProc) words). We use a sliding window through the most dissimilar words where 100% means the window contains the most dissimilar of all words. We also include the distribution of words which have No LIWC category assigned (No LIWC). The y-axes are scaled separately for each sub-plot and the sliding window on the x-axis shows word types ordered by average dissimilarity across embedding spaces. Top row groups words by their part-of-speech and bottom groups words by LIWC categories. We show average and interquartile range for values calculated across all users. Dashed horizontal lines show the average percentage of words that fall into each category when the window is slid over a randomly shuffled list rather than the sorted one.	59
6.2	Hyperparameter search results with one point for each configuration. The line separates where our approach is better (left) or worse (right).	68
7.1	Distribution of the four demographic attributes in our two datasets, showing the set with all demographics known on the right and the random sample from those with at least two known on the left.	81
7.2	Change in perplexity for varying number of similar users considered in weighted fine-tuning for the three similarity metrics.	99
7.3	Heat maps showing normalized similarity for each metric on our 100 author anchor set.	100

LIST OF TABLES

3.1	Sample student utterances from our dataset along with annotations.	19
3.2	F-score figures for the identification of I and B tokens, for course (C) and instructors (I), where * indicates a that our system has a statistically significant improvement for the given token ($p < 0.01$)	22
3.3	Feature ablation for the identification of I and B tokens, for courses (C) and instructors (I). A feature that provides results significantly better than the base feature set is indicated with * ($p < 0.01$)	22
3.4	Precision, Recall and F-score measures for the identification of class and instructor entities	23
3.5	Sentiment accuracies for our system compared to a majority baseline, the Stanford sentiment analysis tool using recursive neural tensor networks, and the inter-annotator agreement.	25
3.6	Sentiment accuracies of different feature models where * indicates a feature whose difference from the default linear weighted bag-of-words is a statistically significant improvement ($p < 0.01$).	25
3.7	Micro-averaged Precision, Recall, and F-score for full targeted sentiment analysis, for both courses and instructors, using ground truth or automatically identified entities.	26
3.8	Accuracy and F-score for different versions of our system, as compared to previous work.	27
4.1	Distribution of messages and tokens (words, punctuation, emoticons) in conversations. Unique averages are computed at the participant level.	30
4.2	Two examples of five-message context windows (ctx_1 and ctx_2) in our dataset.	32
4.3	Prediction results averaged across participants. The majority baseline is compared to models that use embeddings only and a model which uses all features under a general and personal training setting.	38
4.4	Ablation results shown for each feature type and compared to a model that uses all features, as well as baselines obtained using the majority class or message embeddings only.	39
4.5	Manually labeled LDA topics extracted from heuristically segmented conversations. Sample words were chosen from the top 20 highest probability words for each topic. Words in quotes represent tokens replaced for clarity and anonymity.	41

5.1	Distribution of messages and tokens (words, punctuation, emoticons) in the conversations between the author and other individuals.	45
5.2	Distribution of speakers and messages in the corpus by speaker attributes (% of corpus). The values for <i>Age</i> represent ‘younger’, ‘older’, and ‘same age’, while the values for the other attributes represent ‘yes’ and ‘no’.	45
5.3	Dominant LIWC word classes for each attribute/value pair. The top ten classes are listed for each attribute in decreasing order.	46
5.4	Results are shown for the accuracy per person using leave-one-speaker-out cross validation. Individual models learn to classify each attribute in all cases except for the two ‘Joint’ rows, which jointly classify attributes. Feature ablations are shown for each of the single feature types, and compared to the model that uses all features, as well as the baselines obtained using the majority class or message embeddings (Emb) only. Additional improvements are shown when training single attribute classifiers and using the other six attributes as features.	52
5.5	Accuracy on context windows macro-averaged over speakers. The individual, joint, single attribute, and baseline models are defined the same way as in Table 5.4.	53
6.1	Nearest neighbors of “health” for two personalized embedding spaces and the generic space.	58
6.2	Perplexity on the PTB for all valid combinations of weight tying, freezing, and pretraining. Results are sorted by perplexity on Std and shown to three significant figures.	64
6.3	Results for various domains. All other results in this section are for NANC.	65
6.4	Varying the minimum frequency to not be converted into an UNK. The top half shows language model perplexity. The bottom half shows the percentage of word tokens and types that are replaced with UNK in each case.	66
6.5	Percentage of word types and tokens that occur five times or fewer in each dataset. The last two columns are the percentage of types/tokens in the training set that occur five or fewer times in the pretraining set. For PTB the pretraining set is Gigaword (as used in Table 6.2).	66
6.6	Varying similarity and size of pretraining data. Dataset size is shown below the name of each dataset.	67
6.7	Expanding the language model training set.	68
6.8	Final results, training with 8xNANC.	70
6.9	Results for Language Modeling (LM) and Authorship Attribution. Personalized word embeddings significantly improve performance (permutation test, $p < 0.0001$).	71
6.10	Perplexity results broken down by the type of target word (top showing part-of-speech and bottom showing high-level LIWC category), with the best result in bold. OTHER includes all other unlisted part-of-speech tags.	72
7.1	Examples of rules for filtering posts.	82

7.2	Statistics for two Reddit sets: with at least two demographic attributes (2+Dem), or all four demographic attributes (4Dem). Training, development, test splits used in the language modeling experiments are also shown. First row shows overall number of posts and users from the entire set of Reddit posts.	83
7.3	Examples of words with low overlap in nearest neighbors, showing how meaning can differ across the values of a demographic attribute.	83
7.4	Perplexity on the demographic data. Our demographic-based approach improves performance. The difference between the last row and generic words is significant ($p < 0.00001$ with a permutation test).	86
7.5	Perplexity for language models with no demographics (0D) or with all four demographic matrices (4D) with results broken down by demographic values.	87
7.6	Comparing our demographic-based approach with two user-specific approaches. Perplexities are generally lower than previous tables because the threshold for rare words being made UNK was higher.	89
7.7	Comparison of demographic-aware word association similarities for our embeddings using (G)eneric or (G)eneric+(D)emographic, and the best results of the two variants of the composite skip-gram model (C-SGM) from Garimella et al. (2017). We show improved results for USA, India, Male, and Female, and provide new results using age for Younger than 30 and Older.	89
7.8	Results on the 8 disjoint word association subsets for each combination of attributes. Similarities concatenate three embeddings that are each either generic, or specific to that demographic attribute. Overall, using age and gender in combination gives the best performance, though using all three is better on oo3. † indicates statistically significant improvement (permutation test, $p < 0.001$) over the next best model on the marked metric.	90
7.9	Difference in perplexity for our interpolated model and weighted fine-tuning results on the small anchor set. The baseline metrics are subtracted from our model, meaning that more negative perplexity and more positive accuracy are better. The baseline perplexity average is 64.3 for a model that uses standard fine-tuning. Bold indicates best performance.	95
7.10	Difference in perplexity for fine-tuning varying number of similar users on the large anchor set, first fine-tuning on similar users, and second on the new user’s data, as compared to a baseline that fine-tunes on new user data only with perplexity 89.7. Each similar user has 2k tokens and each new user has 2k.	95
7.11	Comparison of our interpolated user embedding similarity model on the large anchor set to a standard fine-tuned baseline measured in perplexity and accuracy @N. We show results for 2k and 6k tokens per anchor user, showing improved performance when more data per anchor is available. Bold indicates best performance.	97
7.12	Top 50 words for which our best model outperforms the baseline based on the frequency of word correctly predicted normalized by the word’s total frequency.	98
7.13	Spearman and Pearson correlation coefficients for each pair of similarity metrics (User Embeddings (UE), Authorship Attribution (AA), and Perplexity (PPL)) computed for each of our 100 anchor users similarity to each new user.	101

ABSTRACT

This thesis focuses on personalization, word representations, and longitudinal dialog. We first look at users expressions of individual preferences. In this targeted sentiment task, we find that we can improve entity extraction and sentiment classification using domain lexicons and linear term weighting. This task is important to personalization and dialog systems, as targets need to be identified in conversation and personal preferences affect how the system should react. Then we examine individuals with large amounts of personal conversational data in order to better predict what people will say. We consider extra-linguistic features that can be used to predict behavior and to predict the relationship between interlocutors. We show that these features improve over just using message content and that training on personal data leads to much better performance than training on a sample from all other users. We look not just at using personal data for these end-tasks, but also constructing personalized word representations. When we have a lot of data for an individual, we create personalized word embeddings that improve performance on language modeling and authorship attribution. When we have limited data, but we have user demographics, we can instead construct demographic word embeddings. We show that these representations improve language modeling and word association performance. When we do not have demographic information, we show that using a small amount of data from an individual, we can calculate similarity to existing users and interpolate or leverage data from these users

to improve language modeling performance. Using these types of personalized word representations, we are able to provide insight into what words vary more across users and demographics. The kind of personalized representations that we introduce in this work allow for applications such as predictive typing, style transfer, and dialog systems. Importantly, they also have the potential to enable more equitable language models, with improved performance for those demographic groups that have little representation in the data.

CHAPTER 1

Introduction

The work described in this thesis relates to conversational systems, personalization, and longitudinal dialog data. We aim to better understand conversational behavior, what makes language different for different individuals, and how personalized models can be applied. We gather data from surveyed individuals, Reddit, and personal text messages to perform experiments. This research is motivated by applications to dialog systems, though experiments implement only parts of full dialog systems.

A large portion of recent research in natural language processing focuses on dialog systems that allow us to interface with computers in spoken or written language. Often, these are task-oriented settings where a system is built to assist a user in accomplishing a task such as booking a flight or ordering food. Other research examines how to build more general conversational systems that can converse with users on any topic. Many types of dialog data exist, varying the number of participants, human or machine, whether the language is scripted, spontaneous, or constrained, and whether it is spoken or written. If one trains a dialog system on any or all of these corpora, the resulting system will predict and understand language patterns common to the whole, but likely will not capture and retain patterns of specific individuals. How then can we create personalized dialog systems? A wealth of data exists in peoples text messages, that is not often used in this research. We construct a corpus from this type of data and made tools to make the extraction process easier, facilitating future research.

In this thesis, we focus mainly on non-task-oriented conversations between humans in various settings and over time spans on the order of years in order to examine aspects of human communication and use this information to model behavior in conversations. These models can be applied to the personalization of conversational systems and help create systems that have an individual personality. This is opposed to methods in NLP in general, that are trained on data gathered from many people. In the general approach, biases from these many humans who produced the data are aggregated into a single corpus. We look not only at personal conversations in text messages, but also at public forum posts and

individuals who post frequently and have been active for many years. Previous work has tried learning representations of such individuals, their language, and how to personalize systems to individuals who have little data. This previous work mainly (1) learns single vector representations of speakers that are appended or added to word vectors, (2) requires social network graph structures to find additional speakers from which to augment data, and (3) uses data sets which have very few utterances per user.

We first experiment with a sentiment analysis application. Many automated conversational systems are linked to a user or customer database which stores information about that individual. For instance, a class recommendation dialog system may know which classes a student took, and when they took them, but does not know how the student liked the classes. This information is important to be able to identify in order to provide higher quality recommendations to the student. This motivates us to study the problem of targeted sentiment analysis, to identify the targets of sentiment in an utterance and value of the sentiment expressed toward them.

We then move to personal text-message conversations exchanged between pairs of individuals and analyze the differences between individuals and properties of their conversations over time. We derive linguistic, psycholinguistic, time, communication-based, and speaker-based features and show that they can be used for predicting the next utterance in a conversation, the response time, and for classifying attributes of speakers. Using personal data and a combination of these features, we show improved performance over models that do not use these features or use a same-sized random sample of data from other individuals.

Next, we study how differences in an individual's word usage leads to different word representations when generating word embeddings. Using public forum data, we examine the differences in word representations and which types of words vary more across individuals and thus are more important to model for the downstream task of predicting the next word in a conversation given a dialog history and part of a response. We build off of previous work to generate embeddings using information about the speaker and the context together to leverage additional data for the generation of these embeddings.

When we do not have a lot of data for a given speaker, for instance, when an individual is new to a given platform, we may want to model their language using what we know about similar speakers. We investigate two methods to solve this problem. The first is to find similar individuals by using a similarity metric. This can be done by looking at the errors of an authorship attribution classifier trained on a set of existing users or by training a language model to learn a user embedding as an additional input. We hope to show that using a language model trained on the clusters of existing speakers allows us to model a new individuals language more accurately than a model trained on all data. Second, we

hope to learn word representations for individuals given their demographics. By learning jointly a matrix of word embeddings and a matrix of demographic embeddings, we can compose embeddings (e.g. by adding an age vector to a gender vector to a generic vector for a word) and show that these serve as an approximation of a new individual's word representations.

The work contained in this thesis shows how personal and longitudinal conversations between individuals can be used to understand the differences in conversational behavior, an individuals sentiment expressions, and the relationship between speakers. We look at how to generate personalized word representations, and more accurately model a person's language. Using recent approaches to modeling language and words using neural networks we address these problems in order to further the creation of personalized language processing systems.

1.1 Research Questions

1. **Can we predict an individual's behaviors, sentiments, and relationships from their conversations?**

We address this question with two types of data. The first are statements made by students about their classes and instructors and is discussed in Chapter 3. We build models to extract the entity targets of sentiment expressions and the polarity of the sentiment expressed toward those targets. We then look at personal data in Chapters 4 and 5 and show how using an individual's conversational data allows us to better predict what people will say, when they will say it, and the relationship between speakers. We present a series of analyses of the data regarding the time messages are sent, the similarity of the style between individuals, the content of messages, psycholinguistic categories of words, and graph-based features derived from how often speakers in the dataset mention each other. We also provide tools to make it easier to extract and analyze personal data from multiple online platforms and perform the same experiments.

2. **Can we improve the prediction of what people will say by using personalized word representations from an individual or from a composition of demographic representations?**

We develop both personalized word embeddings for an individual, in Chapter 6 and

demographic embeddings for a set of demographic attributes, in Chapter 7. In place of generic embeddings we can use personalized word representations, representations from their demographics, if they are known, or a combination of the two. We then use a language model to predict what individuals will say given these representations of words as input. We discuss which types of words have the most different representations across individuals and demographics and the resulting differences in perplexity. These words are thus more important for personalized models to accurately represent.

3. How does the amount of data from a new individual, what we know about them, and ability to measure the similarity of individuals affect how well we can predict what a new individual will say?

Given a new individual for which we have little data, we can find similar individuals from which to model their language using known demographics, which we define as categorical variables, or we can use the text they have written to measure their similarity to existing clusters. We cluster individuals by learning a user embedding as an additional input to a language model, or by training an authorship attribution model on the set of existing users and using the confusion matrix rows to represent points in a high dimensional space. We then look at the effect of the amount of data we have on our ability to model their language. Experiments related to these limited data settings are discussed in detail in Chapter 7.

1.2 Thesis Outline

In the next chapter we describe previous work related to the work in this thesis. It contains work on sentiment analysis, which relates to the understanding of a specific individuals preferences. It mentions dialog systems, for which there has been a wealth of research in recent years, including some work on personalization, though non-dialog work on personalization is also covered. We discuss recent work on language modeling, as it relates to predicting what someone will say. We cover word embeddings, and strategies for low-data settings.

Chapter 3 covers work done on targeted sentiment analysis, whose purpose was to be implemented in a dialog system that could provide advice to undergraduate students planning what courses to take next semester. We extract sentiment targets from student comments and identifying the sentiment expressed towards them.

In Chapters 4 and 5 we examine text messages from a small set of individuals who each have a very large number of personal messages. We extract features for predicting conversational behavior and the relationship between speakers. We perform various analyses to derive features useful for these tasks and include a deeper dive into the authors data.

Then, in chapter 6, we consider using large personal corpora for developing personalized word embeddings that could be helpful not only for modeling language, but for many other possible downstream tasks. We discuss which types of words are more personal and are thus more important for personalized systems to model. In the process, we examine configurations of low compute LSTM language models for using these personalized embeddings to predict user text.

Lastly, in Chapter 7, we examine a scenario where we want to develop a personalized model, or word representations for an individual, but we do not have a lot of data for that individual. We augment an individuals data with data from similar individuals and how to determine similarity given some known attributes, such as demographics, or latent variables, or similarity metrics derived from writing we do have from them.

CHAPTER 2

Related Work

This section covers work in fields most closely related to the work presented in this thesis including sentiment analysis, discourse and dialog, language modeling, word embeddings, authorship profiling and attribution, learning from limited data, and social media. Within each subsection, we discuss how each area relates to personalization. More generally, personalization has been extensively applied to marketing, webpage layout, product and news recommendation, query completion, and dialog [29, 40]. Personalization is at the core of this thesis and is an important aspect of emerging language processing systems, as we move away from large models trained on data from a huge number of individuals, toward specialized models that work for individuals or whose performance is optimized for a particular group of users [43].

2.1 Sentiment Analysis

Most work in sentiment analysis is done at one of three levels: document level, sentence level, and aspect level. These three levels of granularity are ordered from coarsest to finest, with the finer granularity tasks being less well studied. In general, an opinion can be represented by the following quintuple, $(e_i, a_{ij}, oo_{ijkl}, h_k, t_l)$ [203]. The value e_i here represents the i th entity and a_{ij} represents the aspect j of this entity. The k th holder of the opinion is represented by h_k and the time, l , that the opinion is expressed is given by t_l . Given the entity, aspect, holder, and time, one can reason about an opinion orientation oo_{ijkl} . This is usually a positive, negative, or neutral value, although occasionally a larger number of sentiment values are used (e.g., very positive, very negative).

Targeted sentiment is a relatively more recent sentiment task, with contributions focused primarily on settings with scarce resources [133, 204]. The work by Mitchell et al. (2013) introduced pipeline and joint model types and compared the performance of these models using a new set of Tweets. Pipeline models first extract sentiment targets and then

as a second step classify sentiment toward each entity, as opposed to joint models which attempt to solve both tasks simultaneously. The pipeline models have seen more success in recent work [76].

There were two follow-up papers to Mitchell et al. (2013) from the same research group [204, 205]. The first of these papers worked on improving the three models used in Mitchell et al. (2013) including the pipeline, joint, and collapsed models. They show some improvements but the pipeline mode, which is most similar to ours, does not greatly differ in performance. The latter paper used different neural network models on a combination of three data sets. Two of these data sets are derived from Twitter (including Mitchell’s) and the last is derived from MPQA. More recent work in this area has applied neural architectures to Mitchell et al.’s (2013) Twitter data, namely stacked LSTMs [107] and transformers [76] to further improve performance on this task. Both of these studies specifically address the issue of assigning multiple sentiments to an entity when the entity contains multiple tokens. Unlike this work, we avoid this complication by classifying sentiment for each labeled entity span, rather than for each token in the text.

The next most closely related work to ours are the tasks of sentiment slot filling and aspect-based sentiment analysis. Slot filling is the task of discovering information about a named entity and storing it in a knowledge source [179]. The 2013 Text Analysis Conference (TAC) had two similar tasks, which were slot filling and temporal slot filling [178]. For the slot filling task, systems had to determine the correct value for a set of slots for people and organizations. People contained slots such as “date of birth”, “age”, or “spouse”, while organizations contained slots such as “website”, “founded by”, and “country of headquarters”. For the task of temporal slot filling, a system must determine two time ranges, representing the range of times that a given fact is known to be true. Sentiment slot filling is the task of taking a query opinion holder and orientation and returning the set of entities that satisfy this condition. In terms of the quintuple we use to represent sentiment, these are related tasks because although slot filling and temporal slot filling are not exactly sentiment tasks, they are concerned with the entity, aspect, and time values. The sentiment slot filling task is concerned with the entity, orientation, and opinion holder.

Aspect-based sentiment analysis has been the focus of recent SemEval tasks as well as a TAC task [41, 151, 152]. The 2014 sentiment task was continued in 2015, and again in 2016. Researchers submitted a variety of models to evaluate the sentiment of aspects on sets of reviews for laptops, restaurants, and hotels. The highest scoring systems in the SemEval 2015 Task 12 used maximum entropy and support vector machine (SVM) models with bag of words (BoW), verb and adjective lemmas, bigrams after verbs, negation terms, punctuation, point-wise mutual information scores, part of speech tags, and other

features [109, 209]. The results presented were marked as either constrained or unconstrained systems. Unconstrained systems were allowed to use data outside the training data provided, while constrained systems could not. The top two scoring models were unconstrained but the top scoring constrained system used Brown clusters in addition to other features. These are counts of how many words in the sentence belong to semantic clusters of words derived in previous work [65]. Other entries used similar features with several entries using SVM models and a single entry that relied on an unsupervised model.

2.2 Discourse and Dialog

Dialog systems research has grown in recent years and examines methods for constructing systems that can hold conversations with a user, either for some particular purpose or end-goal, or for general purpose conversation. There is also a body of research related to discourse analysis and dialog that seeks to understand the meaning of utterances in conversation and the effect of context on their meaning [60].

Discourse analysis approaches have been used to examine language to reveal social behavior patterns. Discourse structure has been applied to chat communication to identify and visualize message content and interaction structures [71]. This work focused on visualizing aspects such as conversation complexity, overlapping turns, distance between messages, turn changes, patterns in message production and references. In addition, it also proposed graph-based methods for showing coherence and thread patterns during the messaging interaction. Other work inferred social structures in chat-room conversations, using heuristics based on participants' references, message response time, and dialog sequences and represented social structure using graph-based methods [184]. Similarly, other work looked at extracting networks of biographical facts from speech transcripts that characterize the relationships between people and organizations [84].

Speaker behavior in instant messaging services has been widely studied for tasks such as dialog act tagging and discourse analysis. Studies have attempted to classify messages into actions, such as 'greet', 'accept', or 'reject' for online messaging, customer service interactions, and many other settings [44,80]. Other work has focused on the understanding of speakers, and detecting the emotion they are expressing [116].

Many dialog corpora exist. Recent work on building task-oriented and end-to-end dialog systems has used corpora from Twitter [175] and specific types of chatrooms, such as the Ubuntu chat corpus [111]. The construction of such datasets is motivated by the desire to have more useful dialog systems. Although much can be learned from these corpora, systems often also require commonsense reasoning to be effective [177]. One of the most

relevant corpora to our work is the NUS SMS corpus, which contains publicly released text messages, however the authors could not collect messages received, restricting their analysis [22]. Switchboard, a corpus of about 2,400 phone conversations covering 70 topics, has been one of the most widely examined [55].

Recent work has looked at the quality of generated dialog responses and questioned automatic evaluation metrics [138]. Dialog systems trained to maximize response likelihood tend to favor short and vague responses that are applicable to a wide number of circumstances. More recent work attempts to encourage these systems to generate more diverse responses by modified objective functions and adversarial learning [48,208]. Recent work has also focused more on personalization. Most of this work is in the open-domain setting, though some does focus on goal-oriented systems [112]. Goal-oriented dialog has used demographics (i.e. age, gender) to condition system response generation, showing that this relatively coarse grained personalization improves system performance [85]. A recent open-domain study constructs an artificial dataset of personas containing random facts about the persona like ‘I love the beach’ or ‘I am on a diet now’ [207]. Crowd workers are then paired together and asked to have a conversation as if they were a person with those persona attributes. This is not the most realistic scenario, and the resulting dataset does not have a large volume of data per user, though it has led to other work in this area, which has attempted to construct more realistic corpora [121]. It has also led to personalization techniques which use meta-learning in order to quickly adapt to new users [114]. Previous work had not used meta-data about personas and had used movie scripts and a set of tweets with a small number of tweets per user [106].

2.3 Language Modeling

Many language models exist and recent work has greatly increased the performance of general language models and language model pretraining for downstream tasks. Past work has often used language models for speech recognition and translation, but in our work, we use them to evaluate word embeddings. Work on ELMo uses bidirectional language models to pretrain encoders that can provide contextualized word embeddings for downstream tasks. The input text is fed into the pretrained language model and hidden layers can be weighted for specific downstream tasks [148]. The work on BERT improves upon this by training the model while masking tokens in the input as a new way of pretraining that allows the model to learn from both the right and left contexts at the same time [33]. While transformer based models such as GPT-2 now dominate transfer learning, LSTMs continue to be competitive in language modeling without massive compute resources [25, 37, 108, 123–125, 127, 159].

These large transformer models bring additional complications. Fine-tuning models in low data settings is known to be difficult and highly variable [36]. Similar transformer models have been used for controlled generation. One of these models, Grover, was developed for news generation and conditioned on meta-data including domain, date, authors, and headline [201]. No ablation is performed, and though it would be interesting to compare to a transformer method that conditions on authors alone, we opted for a model that is faster and cheaper to train (Grover-Mega was trained for two weeks and cost around 25k USD [201]). Additionally, when fine-tuning models for new users, little data is available. Contextualized embedding models often require a large amount of data to train effectively and recent work has shown that they are competitive with static embeddings in [9]. Our ideas in Chapters 6 and 7 are orthogonal to this prior work and our findings may apply to transformers as well, but these remain open questions.

One of the recent state of the art language models is that of Merity et al. (2018) [127, 128]. This language model uses an ASGD weight-dropped (AWD-LSTM) architecture which takes advantage of recent advances in regularization. Many subsequent models were based on the AWD-LSTM, notably, FRAGE, which along with the Mogrifier LSTM obtain substantial improvements over AWD-LSTM [56]. The authors of FRAGE observe that rare and frequent words occur in different subregions of the embedding space, even when they are semantically similar and add a component to their system to discriminate between rare and frequent words jointly while training. The Mogrifier LSTM is a modification to the LSTM architecture that conditions the input on the recurrent cell state and achieve state-of-the-art results on several language modeling tasks [124]. In this thesis, we are concerned with improving language models with personal information and are not aiming for state-of-the-art performance on general datasets. We find that the AWD-LSTM is relatively fast to run (can be trained on a single GPU in a day) and that higher performing models tend to be slower, though the Mogrifier LSTM code was not available as of the time of experimentation. We therefore find the AWD-LSTM suitable for our experiments.

When personalizing, it is often the case that little or no data is available for a new user. This relates to work on fine-tuning models trained on larger data sets and to the cold-start problem. Recent work has explored training a model to predict what data will be most informative for fine-tuning [5]. The similarity metrics that we derive are used to select data for fine-tuning in one of our methods of leveraging similar user data, however we consider indivisible sets of data grouped by author. The cold start problem is a well-known problem in recommendation systems. A great amount of previous work addressed how to recommend items to new users, about whom the system has little or no history, often with a focus on matrix factorization methods [210]. One study approached language

modeling as a cold-start problem, in that they had no writing from a user, though they had a social network, from which they interpolated language models from users linked in their social graph, though for some applications this network may not exist, or may also provide little data as a new user may have few connections [77]. Personalized language models have also been used for query auto-completion by using factored recurrent RNN cells and learned single-vector user representations [81].

King and Cook (2020) considered interpolating, fine-tuning, and priming language models as methods of personalization [93]. They also analyzed model adaptation for models trained on users with similar demographic factors, inspired by previous work, that showed that these demographic factors could help model a variety of natural language processing tasks, and found that personalized models perform better than those adapted from similar demographics [113]. Other work on interpolating models for personalization has examined related topics, with a focus on handling OOV tokens [170].

2.4 Word Embeddings

A vast amount of work has been done on word embeddings. We focus on learning and applying embeddings, and then, of particular relevance to this thesis, we discuss the biases embeddings learn and methods for learning “user” embeddings.

2.4.1 Learning Embeddings

Much previous work in natural language processing has analyzed the usefulness of word representations of varying sparsity, dimensionality, interpretability, and computational costs. Two of the most salient of which are relevant to this work are word2vec and GloVe [130, 146]. Word2vec is a shallow neural network usually trained in one of two settings. The skip-gram architecture is trained on word pairs occurring within a prespecified distance from each other, taking one word as input and trying to predict the other, while the continuous bag-of-words setup uses the neighbors of an unknown word to predict that word. Many optimization techniques are usually applied to word2vec to speed up training. Being able to quickly train a shallow network allows researchers to apply it to larger datasets and learn higher quality representations. These can be extracted from the model from the input layer, or if a normal softmax function is used, they can be combined from the same-sized weight matrices at each layer. GloVe is another unsupervised method that derives word representations from factoring the co-occurrence matrix of terms in a corpus. They formulate the least squares regression over this matrix to enforce that words are not weighted to heavily

as a result of their frequency. This objective is similar to singular value decomposition used in latent semantic analysis [31]. Resulting embeddings can be used for the same tasks as word2vec and show competitive performance.

Many techniques for embedding words can be applied to other structures as well. Node2vec is an algorithm for learning distributed representations of nodes in a graph by performing biased random walks over the nodes and feeding the sequences to a skip-gram architecture [61]. Other work has examined learning word embeddings jointly with vector representations of users, also using a graph structure, provided by the social network [202].

There has been work in specializing word embeddings, by incorporating external knowledge. This is done either by modifying the skip-gram objective function to jointly learn word representations and enforce certain relationships between the words [92] or by a second optimization to adjust the pretrained embeddings, called retrofitting [53]. This usually involves knowledge that words should be closer together (e.g. synonyms), that words should be farther apart (e.g. antonyms), and that to some degree the original embedding space should be preserved.

Specializing word embeddings can be thought of in different formulations, depending on the type of data that is available, and the desired output. Work on geographically situated language has learned word embeddings from tweets with geotags by jointly learning a general word embedding matrix from all data, and a state-specific matrix of the same dimensions for each state. This is done by summing the general matrix output and the state matrix embeddings, choosing the state matrix by the geotag of the text and only updating that state-specific matrix during backpropagation [11]. Other work has looked at aligning word vectors learned in different embedding spaces by learning a projection based on known seeds, that should be the same in both spaces. The idea is that one embedding space can benefit from another trained on different data and has been shown to be effective across embedding spaces in different languages [86] and for fine-tuning from one domain to another [16].

Some work has attempted to learn word embeddings specific to groups of individuals who share a common attribute [50]. This work learns word embeddings for male and female speakers who live in the USA and India using a skip-gram architecture that learns a separate matrix for each location and gender. They learn embeddings on data from Google Blogger, and explore using them for word association tests. Related work learns personalized word embeddings using two methods. One retrains a word2vec model on each individual's data, while the other keeps the original trained skip-gram layers and inserts a user-adaptive layer in the middle that is tuned to an individual. The models show comparable performance on a sentence completion task [110].

Various subword encodings exist, including character-level and byte-pair encodings, which have been shown to improve language model results and our work could be extended to these encodings [168]. Other work has examined subword embeddings from unigram language models that often outperform byte-pair encoding [18]. Word-level embeddings are easier to interpret in the analyses we perform and to apply to downstream tasks we consider.

A recent framework to learn user embeddings was built on the sentence embeddings generated by BERT [197]. By using the learned user embeddings to predict gender, detect depression and classify MBTI personality, they concluded that their embeddings incorporate intrinsic attributes of users. In our work, user embeddings are learned in a different approach, and we focus on how to use similarity calculated from user embeddings to build better language models.

2.4.2 Applying Embeddings

Word vectors are frequently used in downstream tasks and recent work has shown that their effectiveness depends on domain similarity [9, 149]. For language modeling, previous work explored random and pretrained embeddings and found improvements, but did not consider tying and freezing [94]. In-domain data is also useful for continuing to train contextual embedding models before fine-tuning [62, 64], and for monolingual pretraining in machine translation [10, 135, 158]. This does not cover the interactions between freezing and tying we consider.

Tying input and output embedding matrices in language models has consistently increased performance while reducing the number of model parameters [79, 155]. The improvement is thought to be because otherwise only one input embedding is updated each step and the gradient has to propagate a long way through the model to reach it. Subsequent work has explored more advanced forms of tying, recognizing that the role of the input and output matrices are not exactly the same [141]. This asymmetry has been found in the actual embedding spaces learned and shown to have a negative effect on performance [32, 47]. These observations match the patterns we observe and provide theoretical justification for not tying embeddings when possible.

Part of learning and using embeddings involves handling rare words, which remain challenging even for large transformer models [165]. Recent work has explored copying mechanisms and character based generation [89], with some success. This thesis focuses on English, though for other languages, inflectional morphology and other factors may impact the effectiveness of approaches discussed later chapters [24, 172]. Our work is also

complementary to concurrent work on producing rare words as output [142].

2.4.3 Embedding Bias

Some of the recent work on embeddings has revealed and attempted to remove or mitigate racial, gender, religious, and other biases [17, 118]. An adversarial approach was applied while learning word embeddings to separate the denotation and connotation spaces [187]. The bias in our corpora and embeddings have a societal impact and risks exclusion and demographic misrepresentation [74]. This means that users of certain regions, ages, or genders may find natural language technologies more difficult to use. For instance, when using standard corpora for POS tagging, previous work has found that models perform significantly lower on younger people and ethnic minorities [73]. Similarly, results on text-based geotagging show best results for men over 40 [143].

Similar results are starting to be found in embeddings produced by contextual embedding methods [100, 120]. We focus on non-contextual embedding methods because of their computational efficiency, which is crucial if many separate representations are being learned. Additionally, there may not be a large amount of available data for underrepresented groups and these contextualized models require billions of tokens for training. Recent work has also shown that static embeddings are competitive with contextualized ones in some settings [9].

2.4.4 User Embeddings

The closest work to ours is an exploration of demographic-specific word embedding spaces [50]. They trained word embeddings for male and female speakers who live in the USA and India using skip-gram architectures that learn a separate word matrix for each demographic group (e.g., male speakers from the USA).

Another line of work used discrete [72] or continuous values [113] to learn speaker embeddings: a single vector for each user. The speaker embedding is appended to the input of the recurrent or output layer, and trained simultaneously with the rest of the model. This idea applies to any contextual information type and was introduced as a way to condition language models on topics learned by topic modeling [132]. It has since been used as a way of representing users in tasks such as task-oriented and open-domain dialog [106, 195], information retrieval based on book preferences [3], query auto-completion [81], authorship attribution [39], sarcasm detection [95], sentiment analysis [202], and cold-start language modeling [77].

2.5 Authorship Attribution

Classic authorship attribution is the task of determining which of a fixed set of users is the author of a given text [176]. We intend to test our personalized word representations on this task specifically, though authorship attribution has several related tasks. These include the task of authorship verification, where the classifier determines if a document is written by a given author or not, it's related formulation as authorship clustering, author profiling, where the task is to automatically determine certain characteristics or attributes of authors such as their age, gender, personality, or native language, and the task of author obfuscation, where the goal is to alter a piece of writing to mask the original author's identity [7, 19, 87]. Annual shared tasks are hosted by PAN and researchers compete for the best performance on digital forensics and stylometric tasks [153].

Early work on this task used lexical features like word frequencies and word n-grams [96, 176]. Recent approaches to authorship attribution include using character n-gram convolutional neural networks on tweets [173], syntactic trees encoded with convolutional networks [206], topic models on texts of various levels of formality [169], and a model that attends to syntactic and structural information to create document representations [82]. One study, for which we employ a similar model in Chapter 7, uses neural networks to model similarity between users and predict authorship [51].

2.5.1 Author Profiling

There have been many studies focusing on inferring author's characteristics from their writing, including their gender, age, educational, cultural background, personality, political affiliation, and native language [2, 67, 96]. This work has considered linguistic features to capture lexical, syntactic, structural, and style differences between individuals [96]. The language of Facebook users has been analyzed to identify aspects such as gender, age, and personality by looking at group differences on language usage in words, phrases, and topics [167]. Other work has derived useful information from Twitter profiles, such as Bergsma [13] who focused on gender classification using features derived from usernames, and Argamon [6] who found differences in part of speech and style when examining gender in the British National Corpus. Work in this area has used also used topical, and character-level features and includes comprehensive research analyzing and comparing the application of these features [34].

Some data, such as tweets, can be collected in such a way to capture a useful graph structure, where neighbors can be used to boost performance over textual features [2]. Rao [161] looked at classifying gender, age (older or younger than 30), political leaning,

and region of origin (north or south India) as binary variables using a few hundred or a few thousand tweets from each user. They used the number of followers and following users as network information to look at frequency of tweets, replies, and retweets as communication-based features but found no differences between classes. Hutto [78] analyzed sentiment, topic focus, and network structure in tweeting behavior to understand aspects such as social behavior, message content and following behavior.

2.6 Social Media

We use social media data with demographic attributes inferred from user posts. Prior work has explored extraction or prediction of attributes such as age, gender, region, and political orientation [160, 161]. Work on analyzing the demographics of social media users also includes race/ethnicity, income level, urbanity, emotional stability, personality traits [122], and life satisfaction [23, 38]. Every social media program has different biases in how data is crawled and the population of active users. Recent work has examined how to use demographics in analysis of social media while ensuring populations are well represented [186].

One particularly relevant study presented a corpus of Reddit users with personality information as well as some demographics for a subset of users [52]. Unlike our approach, which is based on text content, they extract information from Reddit flairs, a type of user tag. Out of their set of 10,295 users, 2,253 are also in our set of users (22% of theirs, 0.5% of ours) that have one or more demographic labels, confirming the speculation in their paper that extracting demographics from text is a complementary approach that captures more information about users in their data. Other work has used Reddit posts to identify users who were diagnosed with depression [198] and to construct personas for personalized dialog agents [121].

CHAPTER 3

Individual Preferences

In order to understand personal preferences, we address the task of targeted sentiment as a means of understanding the sentiment that students hold toward courses and instructors, as expressed by students in their comments. We introduce a new dataset consisting of student comments annotated for targeted sentiment and describe a system that can both identify the courses and instructors mentioned in student comments, as well as label the students' sentiment toward those entities. Through several comparative evaluations, we show that our system outperforms previous work on a similar task.

3.1 Introduction

Sentiment analysis is the computational study of people's opinions or emotions; it is a challenging problem that is increasingly being used for decision making by individuals and organizations [140]. There is a significant body of research on sentiment analysis, addressing entire documents [1], including blogs [4, 54] and reviews [20, 199]; sentences [137, 200] or otherwise short spans of texts such as tweets [97, 139]; and phrases [183, 196]. More recent work has also addressed the task of aspect sentiment [101, 151, 181], which aims to address the sentiment toward attributes of the target entity, such as the service in a restaurant [164], or the camera of a mobile phone [21].

In this chapter we address the task of *targeted sentiment*, defined as the task of identifying the sentiment (*positive*, *negative*) or lack thereof (*neutral*) that a writer holds *toward* entities mentioned in a statement. Targeted sentiment had been only recently introduced as a task, to our knowledge with contributions from only two research groups at the time our experiments were performed, that focused primarily on settings with scarce resources [133, 204]. While previous work on data sets such as product reviews can give an accurate measure of sentiment toward products (as explicit targets of the opinions being expressed in the reviews), some corpora include additional challenges. Targeted sentiment

addresses the challenge of identifying entities in running text (e.g., Twitter, student comments, utterances in a conversation), and attributing separate sentiment to each mentioned entity.

Unlike Mitchell et al. (2013), we do not use an artificially balanced data set. Instead we collected all the utterances from students who talked about whichever entities they chose. While we do limit the types of entities to only classes or instructors, we do not limit the specific entities themselves and students can talk about any entities that are relevant to their previous educational experience. Our method is also somewhat different in that we do not evaluate subjectivity: all the entities are assigned a positive, negative, or neutral sentiment, and there are no entities without sentiment.

In our work, we focus on an application-driven task, namely that of understanding students’ sentiment towards courses and instructors as expressed in their comments. As an example, consider the statement:

(1) *I thought that natural language processing with professor Welch was a great class.*

We want to recognize the targets “natural language processing” (a course) and “Welch” (an instructor), as well as a positive sentiment toward the course, and a neutral sentiment toward the instructor. We approach targeted sentiment as a pipeline of two tasks: (1) entity extraction, which aims to identify the entities of interest (in our case, courses and instructors); and (2) entity-centered sentiment analysis, which classifies the sentiment (positive, negative, neutral) held by the student writer toward those entities.

Section 3.2 describes the data used for our experiments. Section 3.3.1 shows how entities are extracted from text for use in targeted sentiment analysis, and Section 3.3.2 describes how the sentiment held toward these entities is classified. An overall evaluation of our system and comparison with previous work are presented in Section 3.4, followed by a discussion and conclusions in Section 3.5.

3.2 Dataset

As we are not aware of any dataset consisting of statements describing courses and instructors, and the sentiment that the writers (students) have toward them, we collected our own dataset. We extracted sentences from a Facebook student group where students describe their experience with classes in the Computer Science department at the University of Michigan, as well as from a survey run with students in the same department. The final

data set consists of 1,042 utterances written by both graduates and undergraduates, describing both classes and instructors that the students had/interacted with. Table 3.1 shows three statement examples drawn from our dataset.

Student utterance	Annotation
I thought that introductory programming concepts was a difficult class and I did not like it.	⟨class name=introductory programming concepts, sentiment=negative⟩
Professor Williams is my favorite teacher that I’ve had so far.	⟨instructor name=Williams, sentiment=positive⟩
I took CS 203 last Winter. Davis was teaching and I thought the class was excellent.	⟨class name = CS 203, sentiment=positive⟩ ⟨instructor name=Davis, sentiment=neutral⟩

Table 3.1: Sample student utterances from our dataset along with annotations.

All the utterances were first manually annotated by one of the authors to identify courses and instructors. As often done in entity extraction methods, we identify entities using an I(nside) O(utside) B(eginning) model. For instance, given the text “I am enrolled in CS 445.”, and assuming the entity to be extracted is a course name, the annotation would include the following labels “I_O am_O enrolled_O in_O CS_B 445_I.”, indicating that CS is at the beginning of the course name, 445 is inside a course name, and all the other tokens are outside the course name.

Classes can be mentioned by department and class ID as in “CS 484,” by ID alone as in “484,” or by name as in “introduction to artificial intelligence” or “intro to AI.” Instructors are mentioned by name, but could be mentioned by first, last, or first and last names. In total, the 1,042 utterances include 976 class mentions and 256 instructor mentions, for a total of 1,232 entities.

The perceived sentiment toward each entity was also manually labeled by one of the authors as either positive, negative, or neutral. When no explicit sentiment is expressed toward an entity, it is assumed to be neutral. If no sentiment is evident from a given utterance, it is assumed to be neutral. Table 3.1 shows the annotations for the three sample utterances from our dataset.

To calculate inter-annotator agreement for the identification of entities, a second annotator labeled 100 utterances from the data set, containing 1,263 tokens. Of these, 1,067 were mutually labeled as not being part of any entity. Of the remaining 196 tokens, 2% were not in agreement. Including all tokens, agreement was measured as 0.987 using Cohen’s kappa. These two percent were two instances where the human judges disagreed

on whether or not a sequence of tokens was a course name (i.e., an entity that needed to be annotated) or simply a course description. For example, in the sentence “I believe that databases are a crucial part of computer science and 520 was interesting,” while “databases” is part of the class name, one annotator decided that the word was simply a description of the content of the course and not an entity.

To calculate inter-annotator agreement for sentiment annotations, a second annotator individually labeled all 1,232 entities. The agreement between the two annotators was measured at 77.7%, which gives a Cohen’s kappa of 0.661 considered to be good agreement. Agreement was calculated as the percentage of entities for which both annotators assigned the same label. Of the annotator disagreements, 10.7% were neutral-negative disagreements, 11.2% neutral-positive disagreements, and 0.2% positive-negative.

3.3 Targeted Sentiment Analysis

We address this task as a pipeline of two steps. We first identify the target entities (i.e., courses and instructors), followed by a classification held by the student writer toward those entities. In the following, we describe and evaluate the method used for each step, and compare the results obtained against the state-of-the-art.

3.3.1 Entity Extraction

As mentioned before, we use an IOB model to identify entities in the text. We therefore apply a classification process to every token in the input text. For each token, we build a feature vector, using the following features:

Core features. These include the current word, the case and part-of-speech of the current word, the previous two words; features are also derived from the two words neighboring the current word, which are computed the same way as for the current word.

Lexicons. We record the presence/absence of words in two custom lexicons: one consisting of the professor names gathered from the University of Michigan; the second one including all the words used in the names of the classes offered in the Computer Science department at the same university. The lexicon features are generated for the current word as well as each neighboring word.

Professor titles. We use a list of titles, such as “Dr.” or “Prof.” to assist with the identification of professor names. The list was compiled manually, and consists of 15 tokens. A feature is generated to indicate whether a token belongs to this list or not. 38% of utterances in the corpus contain professor titles.

Sequence. Students often use a subset of the words in a class name to refer to it. The sequence feature is a binary feature that indicates whether the current word is inside a course or an instructor name sequence, where the courses and instructor names are drawn from the two lexicons described above.

Acronym. The acronym feature is another binary feature that indicates if the input token is an acronym of any class or instructor names in the lexicons. It takes the first letters of each of the words in a name and checks to see if the token matches the concatenated string of first letters for an entry in the lexicon. It subsequently checks if the removal of any number of letters, while retaining order, matches the given token. For instance, “AI” and “ITAI” both match “Introduction to Artificial Intelligence”.

Nearest entity. Sometimes class or instructor names are misspelled, and for such cases lexicon features may not be effective. We create a feature that checks if the current token has an edit distance less than three to a word in a class or instructor name in the lexicons. If a match is found, the feature is set to a value of “C” (course) or “I” (instructor) respectively. If no token exists, the feature is set to “N”.

As a machine learning algorithm, we use a conditional random field, as it has been previously shown to be highly effective for such entity extraction tasks [203]. We run a set of 67-33 train-test splits using stratified sampling. Table 3.2 shows the F-measure results obtained by our system, which makes use of all the features described above, for each of the four token types (B and I for courses and instructors). For comparison, we also show the results obtained with a basic setting, when only the core features are used, as well as the results obtained with a state-of-the-art entity extraction system available from the Stanford NLP group [42], which we have retrained using our corpus. Using our system, we see a statistically significant improvement over the core baseline for all four tokens ($p < 0.01$). We also find a statistically significant improvement over the Stanford system for I_C and B_I ($p < 0.01$) but no significant difference for the other two token types.¹

To gain a better understanding of the role played by each of the features considered, we also perform feature ablation, with results for the individual feature sets shown in Table 4.4. We also show the base feature set for comparison.

Interestingly, while lexicon features show the greatest improvement, the titles feature does not show any improvement over the base features. It is possible that this feature ends up being subsumed by the neighboring words, included in the base features. The sequence, acronym, and nearest entity features are all based on the provided lexicons so it is not surprising that sequence and nearest entity features work well. Among them, the acronym

¹Throughout this chapter, we measure the statistical significance of our results by using a paired t-test with Bonferroni correction using the same 67-33 train-test splits.

System	B_C	I_C	B_I	I_I
Our system	0.945	0.881	0.922	0.901
Baseline (core features)	0.940*	0.849*	0.863*	0.841*
Stanford (Finkel et al., 2005 [42])	0.944	0.848*	0.896*	0.908

Table 3.2: F-score figures for the identification of I and B tokens, for course (C) and instructors (I), where * indicates a that our system has a statistically significant improvement for the given token ($p < 0.01$)

feature appears to be less useful simply because many class names are not commonly abbreviated. The most frequently abbreviated name is “AI” for “artificial intelligence”. Classes are more often referred to by a subset of the words in the class name, which is a case covered by the sequence feature. This is why we see an improvement in I tokens for classes, whereas the instructor I tokens do not show an improvement for these features. There are also fewer I instructor tokens overall in the corpus, which could make it harder to learn the importance of these features.

Since classes can be identified by an ID number (e.g., “490”) or by a name (e.g. “Machine Learning”) we can examine the B_C token in more detail. If we separate the B_C token into a token for class IDs and a token for class name words, we find that the improvement using the lexicon, sequence, and nearest entity features is statistically significant only for the class name words ($p < 0.01$). There is no statistically significant improvement for the ID tokens by themselves, which is not surprising given that the identification of such IDs (most of the times consisting of numbers) is an easy task.

Features	B_C	I_C	B_I	I_I
Baseline (core features)	0.940	0.849	0.863	0.841
Lexicons	0.944*	0.875*	0.915*	0.896*
Titles	0.940	0.851	0.861	0.839
Sequence	0.945*	0.871*	0.858	0.832
Acronym	0.940	0.848	0.860	0.835
Nearest entity	0.944*	0.865*	0.910*	0.895*

Table 3.3: Feature ablation for the identification of I and B tokens, for courses (C) and instructors (I). A feature that provides results significantly better than the base feature set is indicated with * ($p < 0.01$)

We also run an entity-based evaluation, where we use the IOB tokens to construct full

class and instructor names. This is done by finding the B tokens that have the correct following sequence of I tokens. If any of the B or I tokens are missing, or are of the wrong type, the entity is not counted as correct. Table 3.4 shows the precision, recall, and F-score obtained by our system for the extraction of instructor and class entities, and compares our results with those obtained with the Stanford entity extraction system.

Set	Our system			Stanford [42]		
	Precision	Recall	F-score	Precision	Recall	F-score
Instructors	0.833	0.888	0.859	0.711	0.802	0.754
Classes	0.920	0.899	0.910	0.886	0.867	0.876
Both	0.900	0.897	0.900	0.845	0.853	0.849

Table 3.4: Precision, Recall and F-score measures for the identification of class and instructor entities

3.3.2 Entity-Centric Sentiment Analysis

Once the entities of interest are identified, the next step is to determine the sentiment held by the writer (student) toward those entities. This is performed as a classification task using three classes: positive, negative, and neutral. For each candidate entity, we build a feature vector using one of the following configurations:

Weighted bag-of-word. The default model is constructed using unigram counts. The first step is to extract a set of the words that exist in the training set. Using this vocabulary set, counts are constructed for every utterance. These counts are weighted based on their distance, in number of tokens, to the target entity in the statement. For each occurrence of each word, the feature is computed by $\sum_{i \in I} 1/d_{ie}$, where I is the set of occurrences of that word and d is the distance (in words) to the target entity e .

Tree weighted n-grams. A sentence is not linear in nature. A sentence contains clauses and phrases that can be grouped into a tree structure. Consider the sentence “I thought that CS 203 was going to be good, but it was awful”. In this sentence “CS 203” is the target entity and we find that a positive sentiment word “good” is closer (using linear distance in number of tokens) to the entity than the negative sentiment word “awful,” which represents the actual sentiment toward the entity. If we construct a constituency parse tree from this sentence, and calculate the distance as the number of hops between nodes in the tree, then the negative sentiment word is actually closer to the entity word. For each word in an utterance, we calculate this feature as the number of edges in the parse tree between that

word and the target entity. For instance, for the example shown in Figure 3.1, the distance between “awful” and the target entity “203” is six, while the distance between “good” and “203” is eight.

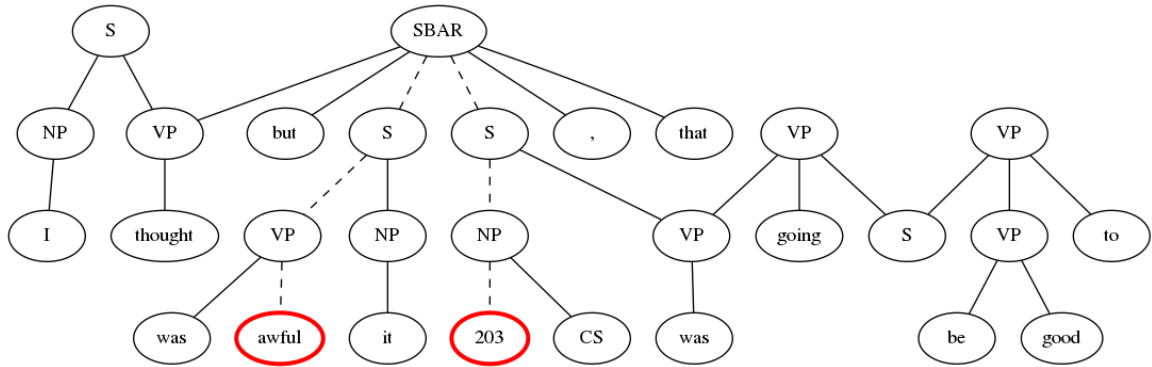


Figure 3.1: Example sentence, “I thought that CS 203 was going to be good, but it was awful”, showing the parse tree weighting for counts using the number of node hops between a given word and the target entity.

Weighted sentiment lexicons. We also implement a feature based on the presence/absence of words from two sentiment lexicons: Bing Liu’s lexicon [75], and the MPQA lexicon [163] [196]. These are two of the most commonly used lexicons in recent sentiment work, and contain 6,789 and 8,222 words respectively, labeled as positive, negative, or neutral. For each word in the utterance, we now generate four features: one simply reflecting the weight of the word (calculated as described before, as a distance to the target entity), and the other three reflecting whether the word appears as a positive, negative, or neutral word in any of the lexicons; these three latter features are again represented as weighted distance scores.

We use an SVM classifier, with a grid search for the SVM cost and gamma parameters performed using three-fold cross validation on the training set. Training and test splits contained approximately 67% and 33% of the data respectively, stratified as mentioned in Section 3.3.1, such that the two entity types (instructor or class) and each of the three sentiment class labels are roughly evenly spread across the train, development, and test sets.

Table 3.5 shows the results obtained with our sentiment analysis system. Note that all the experiments are run on the gold standard set of entities (i.e., manually annotated entities). For comparison, we also report a majority baseline, calculated as the percentage of instances in the entire data set that are neutral, as well as the inter-annotator agreement, as described in Section 3.2.

We also include the result obtained by using the Stanford sentiment analysis tool [174].

We do not retrain this model on our own data, as this would require additional node level annotation for the parse tree of each utterance; instead, we use their sentence level sentiment analysis, which assigns an integer score of 0-4 to each sentence, ranging from “very negative” to “very positive”. We assign the sentence level scores to each entity contained within that sentence. The five values can be mapped to the three values used in our data set in a number of ways, but the way that maximizes the accuracy over our entire data set maps 0 to our “negative,” 1 and 2 to our “neutral,” and 3 and 4 to our “positive.”

Feature	Accuracy
Our system	69.5%
Majority baseline	52.8%
Stanford (Socher et al., 2013 [174])	62.3%
Annotator agreement	77.7%

Table 3.5: Sentiment accuracies for our system compared to a majority baseline, the Stanford sentiment analysis tool using recursive neural tensor networks, and the inter-annotator agreement.

For a deeper analysis, Table 3.6 shows the results obtained by our various features.

Feature	Accuracy
Weighted bag-of-words	67.9%
Tree weighted n-grams	65.6%
Weighted sentiment lexicon	69.5%*

Table 3.6: Sentiment accuracies of different feature models where * indicates a feature whose difference from the default linear weighted bag-of-words is a statistically significant improvement ($p < 0.01$).

3.4 Overall Evaluation and Discussion

In the previous section, we described the methods used for each of the two stages of targeted sentiment analysis, along with results obtained at each stage. We now perform an overall evaluation of this task, and compare our system with previous methods for targeted sentiment analysis.

First, we evaluate the correctness of the sentiment at the entity level, where an entity is marked as correct only if both the entity and the writer’s sentiment toward that entity

are correct. Table 3.7 shows the precision, recall, and F-score obtained for instructor and classes individually, and for all the entities together, assuming: (1) ground truth identification of the entities (i.e., manual annotations); and (2) automatic annotation of the entities using our system from Section 3.3.1.

Entities	Precision	Recall	F-score
Ground Truth Instructors	0.643	0.643	0.643
Ground Truth Classes	0.710	0.710	0.710
Ground Truth Both	0.695	0.695	0.695
Extracted Instructors	0.581	0.578	0.580
Extracted Class	0.571	0.599	0.585
Extracted Both	0.573	0.600	0.586

Table 3.7: Micro-averaged Precision, Recall, and F-score for full targeted sentiment analysis, for both courses and instructors, using ground truth or automatically identified entities.

Second, we compare the results of our system with previous work by Mitchell et al. (2013) [133]. In their work, the authors use a dataset consisting of 2,350 English tweets containing 3,577 volitional entities, which include PERSON and ORGANIZATION entities. They evaluate the performance of the sentiment on entities by checking only the “B” token from the IOB annotation to see if the associated sentiment is correct. If so, it is counted as a true positive. Note that this is less constrained than our evaluation, which also requires that the subsequent I tokens be correct.

In order to allow for a comparison between our system and theirs, we train our pipeline model on their data, by using the same ten-fold cross validation that the authors provided. Note that for this comparison, in the entity extraction step of our system we do not use the lexicon, professor title, acronym, sequence, or nearest entity features because of their domain specificity (these features are specifically aimed at finding sentiment toward courses and instructors, and are not expected to be useful on a dataset of general Twitter data). The results of this comparison are shown in Table 3.8. Mitchell et al. (2013) examine targeted sentiment with only volitional entities and do not use “neutral” as a class for targeted sentiment. For these reasons we include the second and third rows in Table 3.8.

Additionally, because previous work had purposefully not used certain features so that their method could be applied to low resource languages, we also show the performance of the system when we remove the part-of-speech features from our entity extraction step. Note that some of the previous work used accuracy, while other work used F-score; we therefore report both.

We also compare our system to Zhang et al. (2015), who use a neural network model and report their F-score performance on the same corpus [204]. They perform two evaluations, one that uses only positive/negative sentiment, and one that includes the neutral class. We find that our model is comparable when part-of-speech tags are excluded, but outperform the neural models when they are included.

Method	Accuracy	F-score
Our system	68.3%	0.687
Our system, positive/negative sentiment only	68.6%	0.664
Our system, volitional entities, positive/negative sentiment only	70.8%	0.703
Our system, no part-of-speech features	28.9%	0.393
(Mitchell et al., 2013 [133])	30.8%	NA
(Zhang et al., 2015 [204])	NA	0.401
(Zhang et al., 2015 [204]) positive/negative sentiment only	NA	0.279

Table 3.8: Accuracy and F-score for different versions of our system, as compared to previous work.

Discussion. There are a number of errors that are made by our system. Some of the errors come simply from fully or partially missing entities in the first stage of the pipeline. For instance, we found that the named entity recognition fails on some professor names, mainly because some professors use names other than those listed in the online resources that we used to generate our lexicons. A few other less common errors included recognizing first and last names as separate people, and combining class names listed after each other, e.g. “natural language processing and compilers.”

Another batch of errors have correctly recognized entities, but incorrectly classified sentiment. The most common of these cases is incorrectly assigning the neutral class to an entity; the classifier may be somewhat bias toward this class given that it is assigned to 52% of entities in the corpus. Another error involves having multiple entities in a sentence and assigning the sentiment expressed to the wrong entity. For example, in the sentence “I think that John Smith was an interesting teacher in natural language processing”, positive sentiment is incorrectly assigned to “natural language processing.” Another type of error comes from unresolved pronouns. In the utterance, “John Smith taught the data mining class that I took. He was an amazing teacher and I wish that he would teach machine learning,” “John Smith” is classified as having neutral sentiment, rather than positive; coreference resolution could help if we reweighted the features taking into account the correct pronoun set as entity words.

3.5 Conclusions

We addressed the task of targeted sentiment analysis in the context of understanding the sentiment that students hold toward courses and instructors. We introduced a new annotated dataset, collected from students at the University of Michigan, and proposed new features for the extraction of entities and the classification of the sentiment toward these entities. We performed evaluations of each of the two stages in our pipeline model, and showed that both our entity extraction method and the entity-centric sentiment analysis have performance that is competitive with the state-of-the-art. Moreover, in an overall evaluation of our pipeline, we showed that our system exceeds the performance of two previously proposed systems for targeted sentiment analysis [133, 204].

Through several feature ablation analyses, we found that lexicon features play an important role in this task. Further investigation of the use of such lexicons, as well as that of more advanced representations of domain-specific knowledge such as knowledge-graphs may yield stronger results.

The work described in this chapter was part of the Sapphire Project² and was used in the construction of a dialog system for undergraduate academic advising. The work described in this chapter was originally published in [190, 191].

²<http://sapphire.eecs.umich.edu/>

CHAPTER 4

Conversational Behavior

In order to look at how personal preferences affect conversational behavior, we need a source of personal conversations. In this chapter, we obtain and study personal messaging data from SMS and social media platforms. We explore the use of longitudinal dialog data for two dialog prediction tasks related to conversational behavior: next message, and response time prediction. We show that a neural model using personal data that leverages a combination of message content, style matching, time features, and speaker attributes leads to the best results on conversational behavior prediction, with error rate reductions of up to 15% compared to a classifier that relies exclusively on message content and to a classifier that does not use personal data. We then delve deeper into a single individual’s data in order to perform a more in-depth analysis.

4.1 Introduction

Most dialog research provides an overall view of speakers’ language and interaction behaviors based on data from recorded spoken conversations, movie scripts, social network messaging, forums, instant messaging, and audio subtitles [12, 27, 28, 83, 105]. These corpora contain a diverse set of speakers. Thus, the developed models are not tailored to individual speakers, who might have preferences and behaviors different than the consensus trends.

In this work, we address discourse analysis in personal dialog data. In particular, we seek to explore what can be learned from personal messaging history by analyzing language usage and communication patterns. We conduct our analyses over a large set of conversations obtained from the instant messaging history of several individuals. The conversation set contains 1.3 million messages from a five-year time span. We label speaker social relations using seven categories – gender, school, work, relationship status, family, age, and cultural background. We then use psycholinguistic-inspired analysis to analyze language usage within groups in these categories. We use the insights from these analyses

	Participant	Other	All
Total Messages	690,767	647,026	1,340,338
Average Unique Messages	63,039	62,907	123,568
Total Tokens	4,992,575	5,069,745	10,062,320
Average Unique Tokens	19,023	23,265	32,195
Average Tokens / Message	7.23	7.83	7.52

Table 4.1: Distribution of messages and tokens (words, punctuation, emoticons) in conversations. Unique averages are computed at the participant level.

to derive features that represent the message content, messaging frequency and messaging timing. We also derive several features to capture interaction behaviors, including word usage and language matching across conversational groups. We use these features in combination with standard word embeddings to conduct two classification tasks: (1) predicting the next message in the conversation (based on the most common utterances); and (2) predicting the message response time. For both tasks, models with our features and trained on personalized data perform best.

4.2 Dataset

To enable our experiments, we invited individuals to contribute their personal messaging history for a study on personal longitudinal data.¹ To ensure data privacy, we recruited participants who could run our code on their own computers, keeping message content private and sharing only aggregate statistics with us. We recruited eight participants and provided them with detailed instructions on how to prepare the data and run the scripts.

We define the following conversation units: A **message** consists of all the text written by a participant in a conversation right before they press the send key. A **turn change** occurs when the author of the current message differs from the participant in the previous message. Note that a turn can be composed of multiple messages. We define a **conversation** as a sequence of turns between two individuals. **Message response time** is the amount of time that has passed between a message from a user and the previous turn change. On these platforms conversations continue indefinitely, but shifts in response time can indicate when a synchronous exchange has ended.

All messages sent and received by participants via Google Hangouts, iMessage, and Facebook Messenger are considered, covering a range of short message service systems.

¹The study was approved by the Institutional Review Board (IRB) at the University of Michigan.

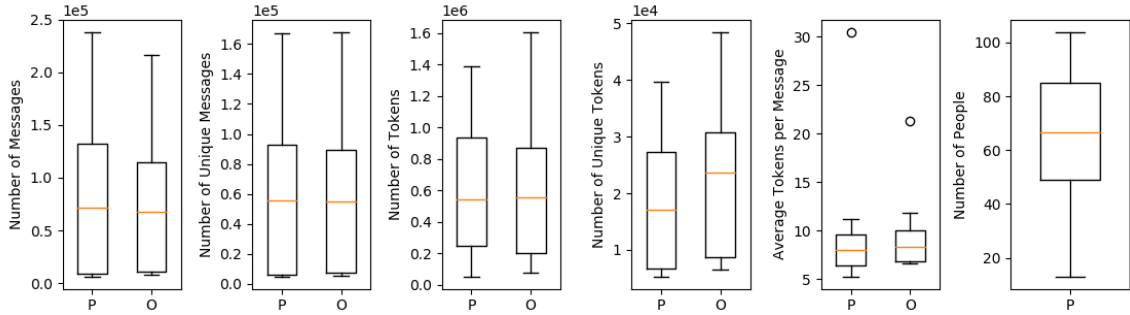


Figure 4.1: Distribution of number of messages and tokens between the (P) participants and their conversation partners (O) in our dataset.

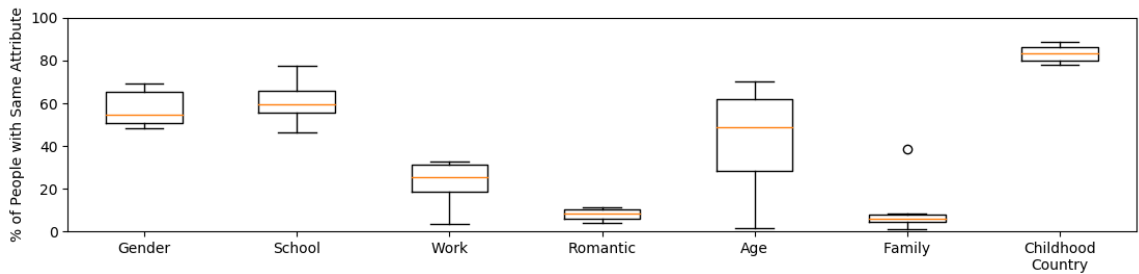


Figure 4.2: Shared attributes between participants and their conversation partners

The data spans a decade and contains about 1.3 million messages,² but we focus on a five year span containing the majority of messages: 2012 to 2017. We also exclude multi-party conversations and conversation partners with fewer than 100 messages. This leads to a final set of 508 interlocutor pairs and contains all the messages from conversations held between the participants and other individuals during 2012-2017. Table 4.1 shows corpus statistics. The data contains slightly more sent messages than received, but sent messages are slightly shorter.

Annotation of Social Interaction Categories

To enable our analyses, each participant manually labeled their conversation partners with seven attributes that describe their social relationship. We chose attributes that they were likely to know about the people they converse with and may impact the way they write. The attributes are defined as follows:

Family: The participant and the other speaker are related.

²There may be some overlap if participants spoke to each other though we cannot quantify it because we do not have access to the raw data.

Table 4.2: Two examples of five-message context windows (ctx_1 and ctx_2) in our dataset.

Message Number	Time	Message
ctx_1msg_0	15:45:06	Participant: Wanna grab coffee?
ctx_1msg_1	15:45:20	Author: yeah
ctx_1msg_2	15:45:25	Participant: Sweet!!!!
ctx_1msg_3	15:45:29	Participant: Meet in the lobby?
ctx_1msg_4	15:45:52	Author: okay
ctx_2msg_0	12:21:00	Participant: Perfect!!
ctx_2msg_1	15:56:22	Participant: Wanna go to get Thai?
ctx_2msg_2	16:01:18	Participant: I'll take it you're sleeping lol
ctx_2msg_3	16:19:59	Author: Yeah
ctx_2msg_4	16:20:08	Author: I mean yeah I was sleeping

Romantic Relationship: The participant and the other speaker's relationship was at some point not platonic.

Relative Age: The participant is older, younger or the same age (± 1.5 years) as the speaker.

Childhood Country: The participant grew up in the same country as the other speaker.

Same Gender: The participant is the same gender as the other speaker.

School: The participant and the other speaker met while attending school.

Work: The participant and the other speaker know each other because they worked together.

These attributes and their values are used during the analyses and experiments presented throughout this chapter. We analyze aggregate statistics of our corpus including total messages and tokens exchanged, the distribution of attributes, and message production across time.

4.2.1 Message and Speaker Distributions

Figure 4.1 shows the distribution of messages and tokens across participants. The leftmost plot shows participants had from a few thousand to a few hundred thousand messages. Distributions are similar for participants and their partners across the number of messages, tokens, and unique messages. The distribution of unique tokens differs, providing some evidence for variation in writing, as each value for O is based on a set of individuals, while each value for P is based on one individual (the participant). The average number of tokens per message ranges from 5-12 with the exception of one outlier, whose messages were significantly longer.

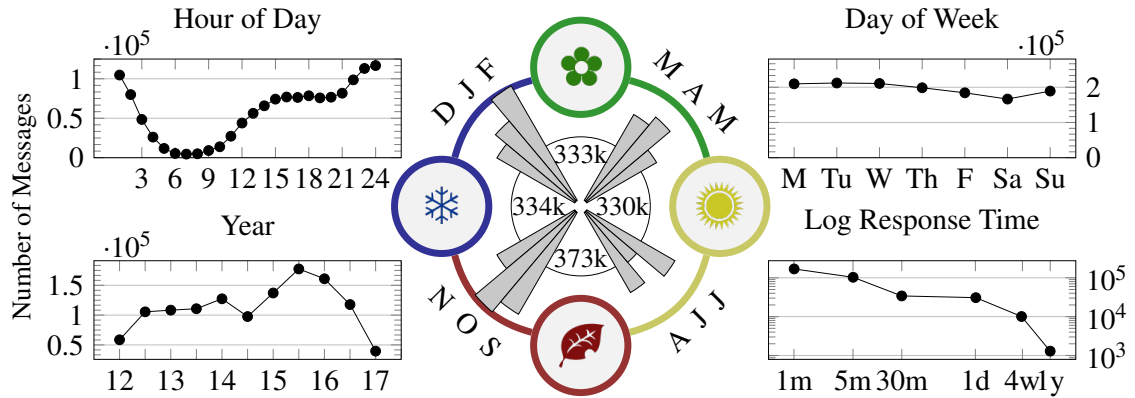


Figure 4.3: Distribution of messages over time. Months are grouped by season. Totals per season are listed in the inner circles with bars from 85k to 115k messages.

We also examine the distribution of speaker attributes over conversation partners and across participants. Figure 4.2 shows this by representing the values ‘yes’ (school, work, romantic, family) or ‘same’ (gender, age, childhood country) for each attribute. For instance, the gender plot shows that the median proportion of conversation partners of the same gender as the participant is 54%. Note that while *age* takes three values the plot shows only *Relative Age=same*. The range is similar for older conversation partners but ranges from 11-37% for those who are younger.

4.2.2 Message Production Across Time

To explore messaging behavior over time, we analyze message exchange trends during conversations based on the time they were sent and speaker response time.

Figure 4.3 presents the distribution of messages over various periods of time: hour of the day, day of the week, season of the year, and across years. Looking at the distribution over months and seasons (middle circle), there is a slight increase during autumn. Looking at the distribution over hour of the day (top left graph), there is an increase until midnight and then a dip in the morning. Looking at the distribution over days of the week (top right graph), there is a decrease as the weekend approaches. This may be instant messaging complementing real-life communication, picking up when real-life communication slows down (beginning of the week) and dropping down when real-life communication picks up (end of the week). Finally, looking at the distribution across years (bottom left graph), there is a peak in late 2015, which might be related to life events, such as starting a new job or starting school.

Figure 4.3 also shows the distribution of message response times with a log-log scale

(bottom right). The graph shows that usually responses occur within a half-hour interval, though there are many up to a day apart, and some a year or more apart.

4.3 Model

During our experiments, we use a bidirectional long-short term memory network (BiLSTM) as our baseline model. We use a sequence of five utterances to define a dialog *context window*.³ The input for our model is the first four messages in the context window, in which all utterances are concatenated but one token is used to represent the beginning of an author utterance and another token is used to represent the beginning of any other speaker’s utterance. The last utterance in the window is the message for which we perform a classification. We use the same implementation to incorporate additional features. For the BiLSTM at time t we use the following standard LSTM equations [68, 166]:

$$i_t = \sigma(W_{ii}w_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \quad (4.1)$$

$$f_t = \sigma(W_{if}w_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \quad (4.2)$$

$$g_t = \tanh(W_{ig}w_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \quad (4.3)$$

$$o_t = \sigma(W_{io}w_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \quad (4.4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (4.5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (4.6)$$

Where w_t is a message embedding obtained with the GloVe model. We use a projection layer q with an output dimension equal to the number of classes. To encode the other sets of features we use the equation for z . For each message of length m we obtain the output as follows:

$$p = W_{ph}h_m + b_p \quad (4.7)$$

$$z = W_{zh}z + b_z \quad (4.8)$$

$$q = W_{q(p+z)}(p \oplus z) + b_q \quad (4.9)$$

The model architecture is shown in Figure 4.4. To encode other feature sets, we use

³Context window size is fixed in our experiments but future work could explore prediction accuracy as a function of this variable.

another fully-connected layer whose output is concatenated with the LSTM output. In the baseline case the feature encoders are not used and the context encoding is passed directly to a classifier or fully-connected layer, from which the softmax of the output gives us a message prediction or response time.

Hyper-parameters for the network, including hidden layer sizes, learning rate, and number of epochs, were tuned on a validation set. For message and response time predictions we tune hidden size in multiples of two from 64 to 512. In our models that use one or more feature encoders, the concatenated ρ vector is used for decoding. The feature encoder sizes t will vary depending on which feature set is being encoded. The word embedding inputs to the context encoder are 300 dimensional.

4.4 Features

Our features are inspired by the group and message production analyses above as well as linguistic aspects in conversational analysis. Personality of the speaker would be a relevant feature but is not feasible for us to obtain ground truth as it would require each speaker to take a personality test. Future work could attempt to gather this data or use a pretrained model for extracting personality from documents [115]. We define several linguistic, time, frequency, and interaction features:

Word Embeddings: We obtain word vector representations for each message using the GloVe Common Crawl pre-trained model [146]. We chose GloVe over other frequently used off-the-shelf embeddings because its training data is more similar to our data and we observed a higher token coverage rate than embeddings such as word2vec trained on GoogleNews [131].

Speaker Attributes: These features aim to represent the relationship(s) between the participant and their conversation partner. We derive binary features representing the seven attributes listed in Section 4.2 for the current conversation partner. If all messages in the context window belong to the participant this vector contains only zeros. When we are predicting one of the seven speaker attributes this feature set represents the values of the other six attributes. Note that we cannot use this feature when training joint speaker attribute models.

Frequency: This set of features attempts to capture the message frequency patterns observed in Figure 4.3. Our features include the number of messages exchanged between conversation participants in the past day, week, month, and from all time. The vector also includes a list of binary values representing the turn change sequence in the context window.

Time: During our analyses we observed important differences in message timing across the day of the week, month, season, and year. To capture these, we define a set of features including the time elapsed during the first four messages in the context window, the number of seconds between each of the first four messages, and the day, month, year, season (winter, fall, summer, spring), and hour of the day of the last message.

LIWC: To capture the semantic categories of text we use the Linguistic Inquire and Word Count (LIWC) lexicon. For each speaker, we calculated normalized counts for the 73 categories. The feature set includes the vectors obtained from messages of individual conversation participants, the cosine similarity between them, and the vector sum of both speakers.

Style Matching: To incorporate information about how the interaction between the participant and their conversation partners changes over time, we calculated the degree to which the speakers match each others language. We use the Linguistic Style Matching (LSM) metric [57], which quantifies to what extent one speaker’s language matches the language of another using eight linguistic markers⁴ from the LIWC dictionary [180]. Specifically, we calculate LSM over the last hundred messages exchanged and the difference in LSM from the beginning to the end of the context window.

4.5 Predicting Conversational Behavior

We consider two prediction tasks related to conversation: 1) predicting the next message in a conversation, and 2) predicting message response times. Our experiments are conducted on context-windows consisting of one message written by a participant and the four preceding messages. Table 4.2 shows two examples of context-windows.

For each participant, we sample random contexts for training and testing. A separate personalized model is trained for each participant and evaluated on the same participant’s test data. For comparison with our personalized models, we also train and evaluate models on general data. For each participant, the data for the general model is sampled randomly from all other participants’ data. For both the general and personalized models the test data is the same. This allows us to measure the impact of having a user-specific model.

4.5.1 Prediction of Next Message in the Conversation

In this task, we must predict which of a small set of messages will occur next in a conversation. This is similar to services like Google’s Smart Reply,⁵ which suggests potential

⁴These are types of function words; quantifiers, conjunctions, adverbs, auxiliary verbs, prepositions, articles, personal pronouns and impersonal pronouns

⁵<https://allo.google.com/>

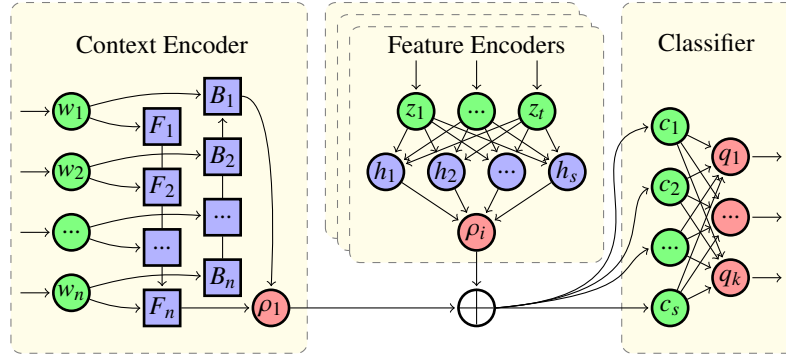


Figure 4.4: The model architecture encodes a context window as a sequence of tokens w_1 to w_n using a BiLSTM. The encoding is then used in combination with our other feature sets for classification by taking the softmax over q . In the case of speaker attribute decoding, separate classifier is used for each speaker attribute.

responses to email and text messages [88]. We structure the task as a multi-label classification problem. We use the top five most frequent utterances sent by each participant as classes. The classes vary slightly but typically include values like ‘yes’, ‘haha’, ‘okay’, ‘oh’, and ‘nice’. We also include an additional category ‘other’, which is a random sample of 1% of the messages sent by the participant (other than the most common five).

During feature extraction, we take the last message in the context window as the label to be predicted and use the previous four messages to generate features as described above. For instance, for the first example in Table 4.2 we assign the label ‘okay’, as it appears in the most common set, but for the second example we assign the label ‘other’, as this message is not one of the five most frequent messages.

4.5.2 Prediction of Message Response Time

In this task, we predict the time till the next message. This kind of information can be used to make conversational agents, such as Microsoft’s XiaoIce, feel more natural.⁶ We address this task as a four-class classification problem, where messages are categorized based on their response time as: (1) the response occurs within 90 seconds of the timestamp of the previous message; (2) between 90 seconds and 10 minutes; (3) more than 10 minutes but less than a day; and (4) longer than a day. For this task, the fifth message in the context window is used to determine the label, and the previous four messages are used to generate features. For example, the response time labels for the context windows shown in Table 4.2 are determined by the time elapsed between msg3 and msg4, which fall into the first category, i.e., the response occurred in under 90 seconds.

⁶<https://blogs.microsoft.com/ai/xiaoice-full-duplex/>

	Next Message	Response Time
Majority Class	32.5	65.8
General Message Embedding	38.0	68.0
General All Features	39.0	70.5
Personal Message Embedding	45.5	69.6
Personal All Features	48.3	73.4

Table 4.3: Prediction results averaged across participants. The majority baseline is compared to models that use embeddings only and a model which uses all features under a general and personal training setting.

The total number of utterances per person (P+O) ranges from 15,000 to 336,000. Two people had too few common utterances for the common utterance prediction task and were excluded from these experiments. Perhaps not surprisingly, we notice that there is a large overlap in common utterances across speakers. The utterance ‘yes’ is in the top two most frequent utterances for all speakers, and laughter (‘haha’) appears in the top two in six of the eight participants. We also consider general and personalized models for this task, with data prepared in the same way as in the message prediction task.

4.5.3 Results

The results for both prediction tasks are shown in Table 4.3. We use an average of 9,500 context windows for next message prediction and 88,000 for response time. We use 80% of the data for training, and 10% for validation and testing respectively. The results show that across the participants in our study, our neural model with all features and personal data performs best, improving over the classifiers that use only message embeddings or classifiers that do not use personal data.⁷

In follow up analysis, we found that as the number of messages in an individual’s dataset increased, the percentage that were short also increased. These messages tend to be fast and close together, leaving less room for improvement on the response time task. Future work could explore the relationship between the number of messages in an individual’s dataset and the accuracy of models trained on their data.

We also perform an ablation using data from the participant with the largest number of messages. Table 4.4 shows the results. For the next message prediction task, the *time*, *LIWC*, and *frequency* features give the largest improvement, increasing classification accuracy by 3.5% over the baseline message embeddings model. For response time predictions,

⁷The next message task excludes two participants who had too few messages.

	Next Message	Response Time
Majority Class	34.0	61.9
Message Embedding	46.5	64.4
Message Embedding + Time	47.0	67.5
Message Embedding + LIWC	47.6	64.4
Message Embedding + Style	46.7	64.4
Message Embedding + Freq	47.6	64.8
Message Embedding + Attributes	46.9	64.5
All Features	50.0	68.0

Table 4.4: Ablation results shown for each feature type and compared to a model that uses all features, as well as baselines obtained using the majority class or message embeddings only.

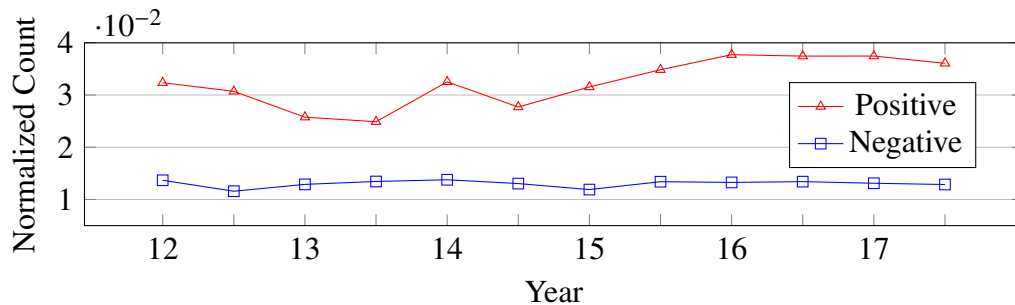


Figure 4.5: Author’s positive and negative emotion words over time shows an increase in positivity from 2012 to the end of 2017.

the previous response times are the most useful feature. However, we find that the combined features give an improvement of 3.6%, or a 10% error reduction. The next most useful features are the speaker attributes and the frequency of past communication.

4.6 Deeper Dive into Personal Longitudinal Dialog Data

Since personal information from the study participants might contain private information our initial analyses are conducted on aggregated metrics across participants. However, because one of the participants is also the author of this thesis we are able to use their subset of 450 thousand messages to illustrate how the analyses and features proposed in this work can provide important insights into personal communication behaviors.

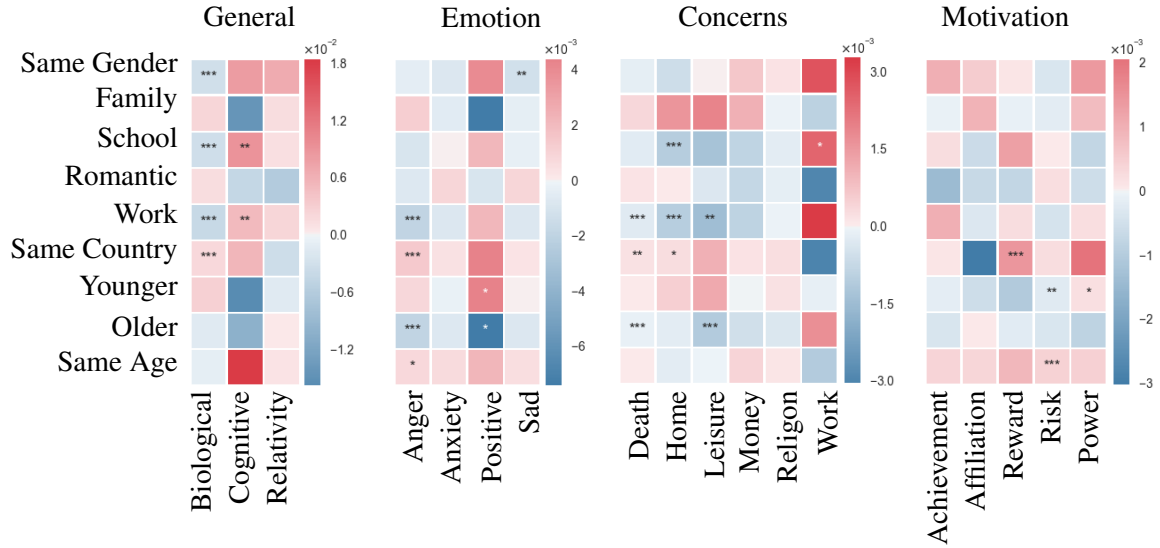


Figure 4.6: Normalized LIWC count differences between groups for the cognitive, emotion and social word categories. Group differences are represented by rows. The color scale shows larger differences in word percentages with lighter colors representing a higher proportion in the first group and darker colors representing a higher proportion in the second (using the groups defined in Section 4.2). Statistically significant differences are shown with * for $p < 0.1$, ** for $p < 0.05$, and *** for $p < 0.01$. Significance is tested using an unpaired t-test over the set of people with Holm-Bonferroni correction [70].

4.6.1 Language Usage Patterns

We examine language usage across the groups the author interacts with using the LIWC word categories over five years of their conversation history. This analysis can reveal patterns on salient language in conversations with the different groups and provide information about the nature of their relationship. For instance, in Figure 4.5 we look at the change in positive and negative words over time and find a steady increase in positive word usage over time and a relatively constant usage of negative words. This finding is consistent with previous studies reporting that with increasing age, individuals use more positive words [145].

To look into salient word categories used while talking to different groups, we plot the use of LIWC classes across the seven groups for the *general*, *emotion*, *concerns* and *motivation* dimensions as shown in Figure 4.6. From this plot, we observe interesting patterns. For instance, conversations with coworkers and school mates are more cognitive in nature. Also, conversations with younger people show more positive emotion, while conversations outside work express more anger. Moreover, conversations happening between coworkers do not include topics such as death, home and leisure, which are more personal in nature.

Another valuable piece of information are the topics discussed during the conversation,

Topic	Sample Words
Communication	phone, email, send, number, text, message, internet, chat
Positive Affect	lol, haha, yeah, like, good, just, nice, dude, fun, man, pretty
Negative Affect	don't, know, feel, people, weird, mean, really, talk, bad
Entertainment	game, play, picture, video, buffy, buy
Food & Time	going, home, eat, work, food, time, today, gonna, cool, want
Planning	go, come, tomorrow, party, tonight, weekend, think, cool, home
Sharing Media	youtube, like, good, music, new, make, imgur, site
Read & Write	read, paper, write, learning, book, think, interesting, learn, language
Travel	pizza, food, airport, russian, english, mom, cat, eat, 'restaurant'
Living & Money	'city', car, buy, live, pay, house, drive, places, money, 'hometown'

Table 4.5: Manually labeled LDA topics extracted from heuristically segmented conversations. Sample words were chosen from the top 20 highest probability words for each topic. Words in quotes represent tokens replaced for clarity and anonymity.

which can help to better understand the author interests and habits. We use Latent Dirichlet Allocation (LDA) [14] to analyze the topics frequently discussed during the messaging history. LDA is applied at conversation level and conversations are obtained by splitting the messages based the time elapsed between the speakers' messages.⁸ The topics obtained from 23,970 conversations are shown in Table 4.5.⁹ The topics include expressing positive and negative feelings (*Positive Affect*, *Negative Affect*), activities (*Sharing Media*, *Reading and Writing*, *Food and Time*, *Entertainment*), *Planning*, and *Travel*.

The 'Travel' topic has words for food, pets, and family, as traveling often includes visiting family and pets, and going out to dinner. Conversations are often about shared media, or activities, such as reading, writing, computing, or entertainment (sample words include games, and references to the TV show *Buffy the Vampire Slayer*). The 'begin/end/sleep' category includes greetings, and things said at the end of conversations (as sometimes conversations end when one person is going to sleep).

4.7 Conclusions

In this chapter, we studied a corpus of personal conversations consisting of the instant messaging history from eight individuals. The analyses were conducted over 1.3 million messages written over a five-year time span.

⁸We consider that a new conversation starts when the time elapsed between messages from both parties is longer than five hours. This number was obtained empirically by testing different segmentation values.

⁹We extract 10 topics, with the number of topics being selected based on the choice of topics in previous work.

We developed several linguistic features inspired by conversational and interaction behaviors we observed in the longitudinal data. Our features include message content, style matching, time features, and speaker attributes. These features were used to address two classification tasks: predicting common messages and message response times. While the most common utterances and distribution of response times vary across speakers, we found that a classifier that relies on a combination of all proposed features and uses personal data leads to error reductions of up to 15% compared to classifiers that exclusively rely on message content or are trained on messages randomly selected from other speakers in the corpus.

Our code is publicly available¹⁰ so that others may perform similar analyses and experiments on their own personal longitudinal data or other data, to discover patterns in messaging behavior and train models for dialog prediction tasks. This work was originally published in [193].

¹⁰https://github.com/cfwelch/longitudinal_dialog

CHAPTER 5

Relationship Between Interlocutors

An important detail of personalization concerns the nature of the relationship between interlocutors. The way we speak to others depends on how well we know them, past interactions, whether it is a professional relationship or not, and many other factors. In this chapter, we examine a large dialog corpus obtained from the conversation history of a single individual with 104 conversation partners. The corpus consists of half a million instant messages, across several messaging platforms. We focus our analyses on seven speaker attributes, each of which partitions the set of speakers, namely: gender; relative age; family member; romantic partner; classmate; co-worker; and native to the same country. In addition to the content of the messages, we examine conversational aspects such as the time messages are sent, messaging frequency, psycholinguistic word categories, linguistic mirroring, and graph-based features reflecting how people in the corpus mention each other. We present two sets of experiments predicting each attribute using (1) short context windows; and (2) a larger set of messages. We find that using all features leads to gains of 9-14% over using message text only.

5.1 Introduction

People spend a significant amount of time using social media services such as instant messaging to communicate and keep in touch with others. Over time, conversation history can grow quickly, thus becoming an abundant source of personal data that provides the opportunity to study an individual's communication patterns and social preferences. Analyzing conversations from a single individual rather than conversations from multiple individuals can enable identification of social behaviors that are specific to that individual. Moreover, longitudinal analyses can help us better understand an individual's social interactions and how they develop over time.

In this work we look further at the collection of personal conversations of this thesis'

author. These messages are from over a five-year span, consisting of nearly half a million messages shared with 104 conversation partners. We focus our analyses on seven speaker attributes: a ternary attribute for relative age (younger, older, or same age); and six binary attributes reflecting whether somebody is the same gender; a family member; a romantic partner; a classmate; a co-worker; and a native of the same country. We explore the classification of speaker attributes, i.e, the group(s) the speaker belongs to, using a variety of linguistic features, message and time frequency features, stylistic and psycholinguistic features, as in the previous chapter, with one additional set of graph-based features. In addition, we examine the performance increase gained by using six of the attributes as features to try to classify the seventh.

We analyze linguistic variation in messages exchanged between the author and the other speakers. We also conduct analyses that look at speaker interaction behaviors, considering aspects such as time, messaging frequency, turn-taking, and linguistic mirroring. Next, we apply graph-based methods to model how people interact with each other by representing people as nodes and speaker mentioning each other as directed edges. We then apply clustering methods to identify groups that naturally occur in the graph. Finally, we conduct several classification experiments to quantify the impact of features derived from these analyses on our ability to determine who a speaker is.

Identifying speaker attributes has important applications within the areas of personalization and recommendation [49, 161]. While a large number of conversations that occur online are short, such as interactions on Twitter, there are also many social media platforms where personal dialog may span thousands of utterances. For this reason, we conduct evaluations at the level of small context windows, as well as at the speaker level using a large set of messages from each speaker. To the best of our knowledge, this is the first study on speaker attribute prediction using personal longitudinal dialog data that focuses on one person’s dialog interactions with many other speakers.

5.2 Dataset

We use a corpus of text messages from the author’s personal conversations on Google Hangouts, Facebook Messenger, and SMS text messages. The message set contains nearly half a million messages from conversations held between the author and 104 individuals. Aggregate statistics describing the corpus are shown in Table 5.1.

We use the seven speaker attributes described in Chapter 4 that describe the relationship between the author and their conversation partner. Table 5.2 shows the distribution of people and messages for each attribute in the dataset.

	Author	Others	All
Total Messages	237,300	216,766	454,066
Unique Messages	165,536	168,041	326,243
Total Tokens	1,370,916	1,602,607	2,973,523
Unique Tokens	38,937	48,005	68,985
Average Tokens / Message	5.78	7.39	6.55

Table 5.1: Distribution of messages and tokens (words, punctuation, emoticons) in the conversations between the author and other individuals.

Table 5.2: Distribution of speakers and messages in the corpus by speaker attributes (% of corpus). The values for *Age* represent ‘younger’, ‘older’, and ‘same age’, while the values for the other attributes represent ‘yes’ and ‘no’.

	Family	Romantic Relationship	Relative Age	Childhood Country	Gender	School	Work
	Y/N	Y/N	Y/O/S	Y/N	Y/N	Y/N	Y/N
%Speakers	6/94	9/91	26/30/44	78/20	51/49	62/38	33/67
%Messages	8/92	22/78	24/24/52	88/11	53/47	75/25	54/46

5.3 Message Content

We start by exploring linguistic differences in the messages exchanged between the author and each of the groups defined by the seven attributes described above. We obtain the most dominant semantic word classes [156] in messages exchanged with people sharing each attribute using the LIWC [180] lexicon, which contains psycholinguistic categories of words. The top ten dominant classes for each attribute-value pair are shown in Table 5.3.

Not surprisingly, the ‘Family=Yes’ group talks more about family and home than the ‘Family=No’ group. Interestingly, people who are not family members seem to use more emotion related words. Word categories related to feelings are also very dominant for the ‘Romantic Relationship=Yes’, ‘Relative Age=Same’, ‘Childhood Country=Same’ and ‘Gender=No’ groups; however they seem to focus on negative emotions such as anxiety and sadness. In fact, those two are in the top three classes for conversations with romantic partners ‘Romantic Relationship=Yes’, which also includes death words. This suggests that more serious conversations occur between the author and this group as compared to the ‘Romantic Relationship=No’ group (although, words related to death are also often used in hyperbole, e.g. “I didn’t eat lunch and I’m dying”).

Several of the attributes clearly separate the set of speakers into those who speak about

Table 5.3: Dominant LIWC word classes for each attribute/value pair. The top ten classes are listed for each attribute in decreasing order.

Attribute	Top Classes
Family	Yes: Family, Money, Home, Swear, Death, Leisure, Filler, Anger, Female, Health No: Anxious, Insight, Feel, Risk, Sad, Positive Emotion, Non-fluencies, Causality, Affect, Work
Romantic Relationship	Yes: Anxious, Death, Sad, Feel, Body, Filler, You, Family, Perception, Health No: Swear, Female, Money, Friend, Anger, She-He, Work, Leisure, Informal, Male
Relative Age	Younger: Netspeak, Ingest, Swear, Friend, Biological, Home, Anger, Informal, Body, Leisure Same: Female, Swear, Anger, She-He, Anxious, Negative Emotion, Friend, Sad, Negate, Money Older: See, We, Work, Number, Article, Home, Perception, Space, Motion, Relativity
Childhood Country	Same: Death, Family, Anger, Swear, Feel, Female, Negative Emotion, Body, Anxious, Health Other: We, Work, You, Male, Focus Future, Social, Affiliation, Friend, Assent, Time
Gender	Yes: Money, Female, Swear, Work, Friend, Netspeak, She-He, Article, Power No: Sad, Anxious, Family, Health, Death, Body, Biological, Negative Emotion, Ingest, Home
School	Yes: Work, Non-fluencies, Insight, Risk, Anxious, Quantify, Focus Past, Causality, Tentative, Compare No: Family, Money, Health, Home, Netspeak, Death, Swear, Leisure, Biological, Anger
Work	Yes: Work, Article, Number, We, Non-fluencies, Quantify, Compare, Insight, Achievement, Assent No: Family, Health, Money, Death, Anger, Swear, Anxious, Home, Biological, Sad

work and those who do not. People who talk the most about work are those who grew up in other countries ('Childhood Country=Other'), people from work ('Work=Yes'), people older than the author ('Relative Age=Older'), people with the same gender ('Gender=Yes') and people from school ('School=Yes'). However, there are some differences between these groups which can be seen mostly in the family, health, time, and gender specific words they use.

People from school use more words referring to the past, while people from other coun-

tries focus more on the future. Interestingly, people not from work ('Work=No') and the people not from school ('School=No') are very similar, and both use a lot of family, health, and money words. The similarity of these two attributes is also interesting in that people from work ('Work=Yes') and/or school ('School=Yes') use more quantifying words (e.g. sampling, percent, average) and disfluencies (e.g. umm, hmm, sigh). We also see that those who grew up in other countries use more male words, while speakers that are the same age, from the same country, or of the same gender use more female words.

5.4 Groups Over Time

To understand the role that time has in the author's interactions with different groups we look at patterns in message volume over different intervals. Most notably, we find interaction differences given the day of the week, and the hour of the day. In Figure 5.1 we plot the attribute/value pairs that differ the most from the trend over all people, marked 'All'. The difference was calculated as the sum of differences on each of the seven days of the week and each of the 24 hours of the day.

We see that the overall trend for the day-of-week plot (top) is that there are more conversations during the first days of the week. The number of conversations drops until Sunday where it jumps back up and peaks on Monday. Throughout the week, most of the conversations occur between family members and people that grew up in other countries (co-workers mainly). In contrast, there are many more conversations with people outside of work on the weekend.

The hour-of-day plot (bottom) indicates that most of the interactions happen between 9AM and 6PM. Though this is a trend aggregated over all days in the corpus it shows that the author is least likely to be talking to people in the 7-8AM range. The author tends to speak more to people later in the day, with a peak at midnight. People who grew up in other countries converse more with the author during the day. The dominant 'Work' category for 'Childhood Country=Other' in Table 5.3 shows this trend, as this group may converse with the author more about work during work hours. We also find that family members speak to the author more during the day and romantic partners speak to the author more after midnight but before noon.

5.5 Conversation Interaction

Linguistic mirroring is a behavior in which one person subconsciously imitates the linguistic patterns of their conversation partner. Increased linguistic mirroring can be an indicator

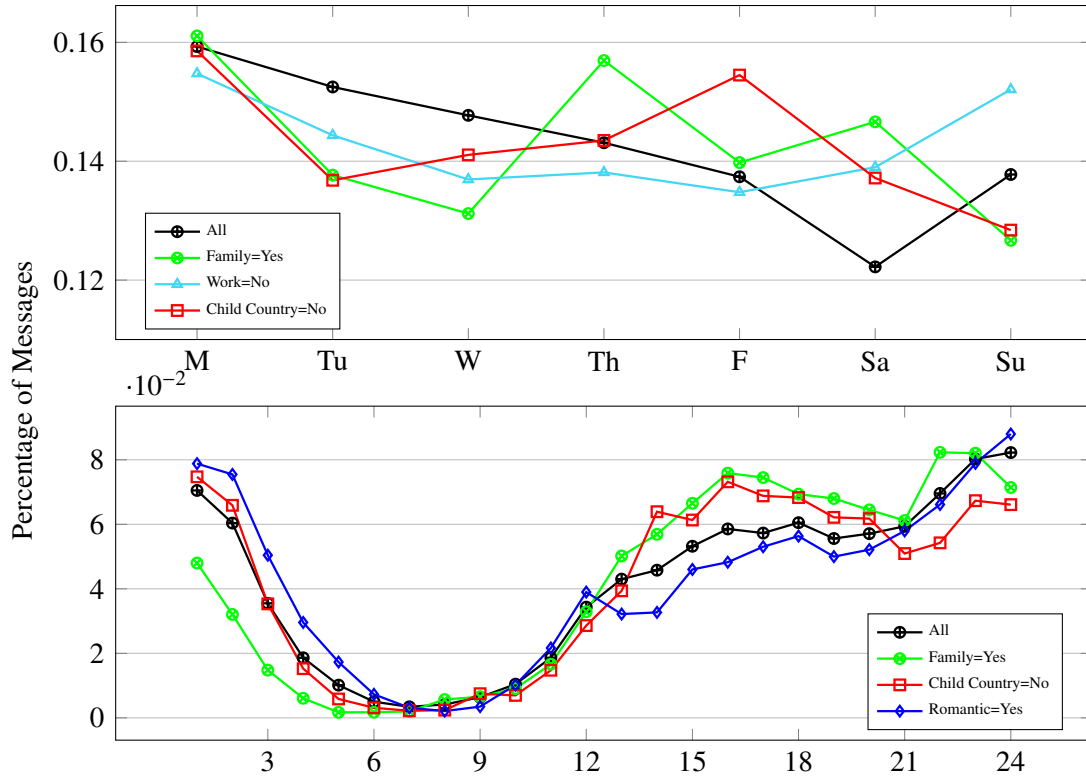


Figure 5.1: Distribution of messages over time. The top shows the distribution over the day of the week and the bottom shows hour of the day. The groups shown are those that vary the most from the aggregate trend over all speakers.

of an individual building rapport with others and thus forming better interpersonal relationships. We study linguistic mirroring in our dataset to analyze how relationships change over time. We calculate linguistic style matching (LSM) as the similarity of the normalized counts of nine types of function words [57], as the main metric for our analyses. In Figure 5.2 we show style matching over the first 5,000 messages with people in five specific groups. The trend here is cumulative over these messages. We also tried plotting a sliding window, but the trend was much harder to see because style matching fluctuates over time. We see that although the general trend is to match language style more over time, this trend levels off after 3k messages, potentially because at this point relationships start to consolidate, however, in interactions with many individuals, the relationship is already well established.

Next, we examine interactions between groups of people by constructing a graph where nodes represent speakers and edges between nodes represent speakers mentioning each other. Speakers who mention each other also tend to know each other. They might mention another person when planning to meet up with others or when talking about an interaction

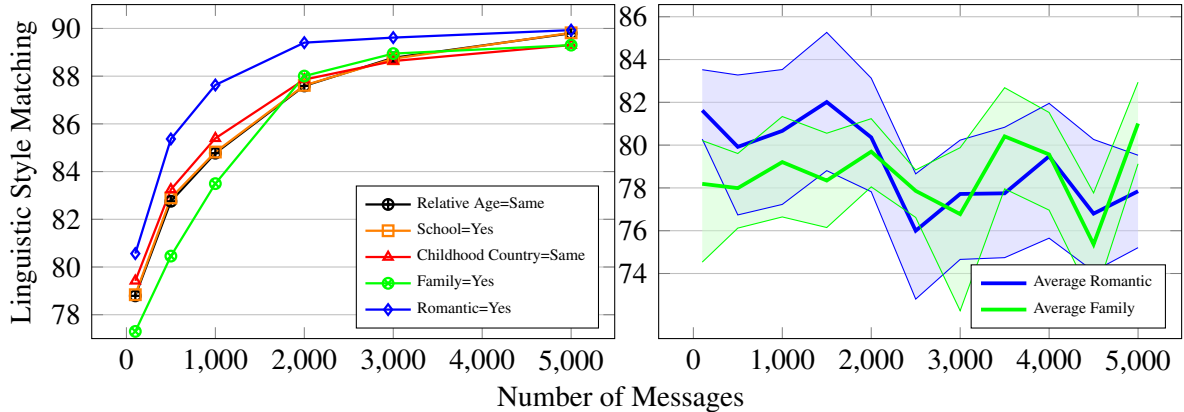


Figure 5.2: Language mirroring as a function of the number of messages exchanged within groups. The left figure shows the mirroring cumulatively over the first 5,000 messages averaged over people in each of the listed groups. The right figure shows these numbers averaged over a sliding window with the shaded region representing the interquartile range.

they had with this person in the past. We clustered the graph of people using Louvain clustering [15] to maximize the modularity of the network. This gave four clusters, one of which only contained two people. The remaining clusters roughly evenly split the set of people. The top twenty most frequent conversation partners are shown in Figure 5.3. Interestingly, the clusters resemble groups of speakers that the author spoke most to at three periods of time contained in the corpus i.e, conversations before attending graduate school (Cluster 3), the beginning of graduate school (Cluster 2), and later in graduate school (Cluster 1). We also see that people who spoke to the author more at a particular time were also more likely to know each other.

5.6 Model and Features

Using the messages in a conversation between two speakers, we wish to be able to identify the value of each of the speaker attributes of whom the author is conversing with. In order to do this, we can encode part of the conversation and additional features as before in Chapter 4, and output the value of an attribute. In this setup, the size for both the feature encoders and attribute decoders were manually tuned in preliminary experiments, however, due to the computational requirements of our evaluation, we use a smaller LSTM hidden size of 64.

We use the same size context windows as Chapter 4 (see 4.2 for examples). As before, the context window messages are concatenated with speaker tokens between utterances to

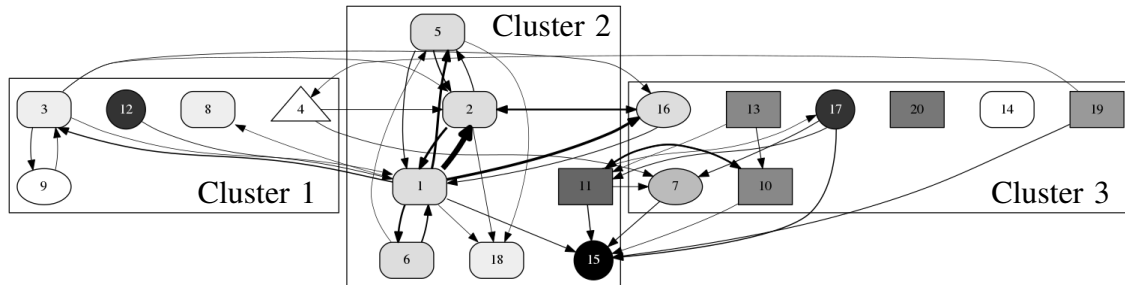


Figure 5.3: Speaker references for the top 20 conversation partners. The graph shows interactions with people from different groups: high school (rectangles), college (triangles), graduate school (rounded rectangles), family members (circles), and other people (ellipses). Shading is proportional to how long ago the author met the person. Edges below a threshold of 25 mentions are removed. Note that the clustering uses all 104 people, but only 20 are shown here.

represent either the author, or the other speaker, and fed into a BiLSTM. The model concatenates the BiLSTM output with encoded feature vectors, to be passed to an attribute decoder. A separate decoder is used for each speaker attribute and has k outputs, where k is two for every case except ‘relative age’, which has three possible values. We use the same features as the previous chapter, which include word embeddings from GloVe Common Crawl, and vectors representing the LIWC categories of words, timing of messages, frequency of communication, and style matching, but also include one additional feature based on conversation interaction.

Graph-based: Our graph-based features use the training set of messages to generate a graph where nodes represent people and weighted, directed edges represent how often one person mentions another person when speaking to the participant. This graph is used to generate features by finding the shortest path between users where edge weights are smaller when they have more mentions. We then use the adjacency matrix to find the shortest paths between nodes and use each row as a feature set, representing a speaker i conversing with this person. Given a graph of mentions, where $M_{i,j}$ represents how often person i mentions person j , we compute weights using the following equation:

$$W_{i,j} = 1 - \frac{w_{max} - M_{i,j}}{w_{max} - w_{min}}$$

Intuitively, if we assume that people who know each other better mention each other more often, and that the probability of people knowing each other is transitive, then a vector in this matrix represents how relatively well a given speaker knows each other speaker.

5.7 Experiments

Using the features described in Section 4.4 we run experiments using leave-one-speaker-out cross validation. We take the 104 speakers in our dataset and hold out all context windows containing dialog with one of the speakers as a test set and use the rest for training and validation with a 90% and 10% split. This means that we train and tune parameters on context windows from all 103 other speakers and update the model based on its predictions on each individual context window. During test time we examine the context-level and speaker-level accuracy. Context-level accuracy is calculated by macro-averaging the context window accuracy over all speakers. To calculate accuracy at speaker level, we first obtain the attribute prediction at context-window level for the held-out speaker and assign the attribute value most frequently predicted by the classifier.

We run experiments using a baseline model which only uses word embeddings and compare it to a model that uses all of our features. Additionally, we perform an ablation to examine the effectiveness of each feature set for predicting each speaker attribute by running the model using the word embeddings plus one of the other feature sets at a time. While we vary the number of feature encoders we use (see Figure 4.4), each model always uses one attribute decoder. The loss for each model is calculated as the cross-entropy loss for that model’s attribute decoder.

Since this evaluation is computationally expensive we run our experiments on a subset of the original corpus. Thus, we obtain a sample of 27,316 context windows, distributed as evenly as possible, from each speaker in the dataset to ensure that all people and attributes are represented. Experiments using this dataset took 3-4 days to run on a cluster with 12 NVIDIA GeForce GTX TITAN X GPUs.

During our experiments we consider single attribute models, which use only one attribute decoder, and joint models, which learn to predict all attributes at the same time using all decoders. In the single attribute setting we train a separate model for each attribute and calculate the cross-entropy loss for the decoder, while in the joint case we take the sum of the losses for all decoders.

5.8 Results

The results obtained for each attribute, when using different combinations of features are shown in Table 5.4 and Table 5.5. The first table shows accuracies at the person-level while the latter shows performance macro-averaged over context-windows. Overall, the combination of all features improves the prediction performance for all the attributes over

	Family	Rom. Rel.	Relative Age	Child. Country	Gender	School	Work
Baselines							
Majority Class	94.2	91.3	44.2	77.9	51.0	61.5	67.3
Emb	94.2	91.3	45.2	79.8	86.5	73.1	80.8
Single Attribute Decoder Ablation							
Emb + Time	94.2	91.3	44.2	79.8	85.6	76.0	85.6
Emb + LIWC	94.2	91.3	46.2	80.8	82.7	73.1	84.6
Emb + Style	94.2	91.3	49.0	78.8	86.5	76.0	85.6
Emb + Frequency	94.2	91.3	44.2	80.8	83.7	75.0	86.5
Emb + Graph	93.3	91.3	43.3	77.9	80.8	76.0	87.5
Single Attribute Decoder All Features vs Joint Decoder Models							
All Features	92.3	91.3	45.2	81.7	76.0	76.9	83.7
Joint + Emb	94.2	91.3	48.1	78.8	85.6	71.2	83.7
Joint + All	92.3	91.3	51.9	84.6	77.9	75.0	84.6
Single Attribute Decoder with Attribute Features							
Emb + Attributes	94.2	91.3	48.1	87.5	83.7	73.1	84.6
All + Attributes	93.3	91.3	50.0	88.5	78.8	78.8	85.6

Table 5.4: Results are shown for the accuracy per person using leave-one-speaker-out cross validation. Individual models learn to classify each attribute in all cases except for the two ‘Joint’ rows, which jointly classify attributes. Feature ablations are shown for each of the single feature types, and compared to the model that uses all features, as well as the baselines obtained using the majority class or message embeddings (Emb) only. Additional improvements are shown when training single attribute classifiers and using the other six attributes as features.

a baseline model that only uses word embeddings, with the exception of the gender attribute. The largest context-window level improvements are obtained for the *Relative age*, *Childhood country*, *Gender* and *Work* attributes. The largest speaker-level improvements are similar with the addition of *School* and without *Gender*.

Although in some cases the accuracy of attribute prediction at speaker-level is not improved by the different set of features, we still observe an improvement on the prediction accuracy at the context window level. For instance, the *Family* and *Romantic* attributes improve by 2.1% and 6% respectively. We also see that the *Gender* attribute improves up to 6.8% by this metric.

Using the other six speaker attributes as features to classify the seventh proved to be beneficial in all cases. The graph features also proved useful for all attributes showing

	Family	Rom. Rel.	Relative Age	Child. Country	Gender	School	Work
Baselines							
Majority Class	94.2	91.3	44.2	77.9	51.0	61.5	67.3
Emb	92.0	86.0	39.2	75.7	63.7	64.6	69.5
Single Attribute Decoder Ablation							
Emb + Time	91.7	86.8	40.5	77.4	63.4	64.4	73.1
Emb + LIWC	91.9	86.4	39.6	76.7	62.6	63.8	69.4
Emb + Style	92.0	86.0	38.9	76.2	62.8	65.1	69.2
Emb + Frequency	91.3	87.9	39.2	76.0	62.4	65.5	71.3
Emb + Graph	92.1	86.2	41.7	76.9	61.4	67.2	73.3
Single Attribute Decoder All Features vs Joint Decoder Models							
All Features	92.0	88.1	42.7	78.9	61.2	67.0	76.0
Joint + Emb	93.9	90.9	43.4	78.0	64.2	65.5	69.3
Joint + All	92.1	90.2	47.2	80.8	61.8	68.7	78.4
Single Attribute Decoder with Attribute Features							
Emb + Attributes	92.6	86.4	41.5	84.1	68.6	72.7	78.4
All + Attributes	92.0	88.2	44.3	85.7	67.1	74.3	83.4

Table 5.5: Accuracy on context windows macro-averaged over speakers. The individual, joint, single attribute, and baseline models are defined the same way as in Table 5.4.

gains of up to 6.7% in speaker-level performance and up to 7% in context-window level performance. The frequency features gave the biggest performance increase to the *Romantic*, *Childhood country*, and *Work* attributes. Time features improve performance most on *Romantic*, *Gender*, *School*, *Work*.

The overall trend we found in Section 5.5 showed that the most distinct groups when looking at language mirroring were ‘Family=Yes’ and ‘Romantic=Yes’. However, we found that the language mirroring features that we used, which use a sliding window, were most useful for *Relative age*, *School*, and *Work*. Similarly, LIWC features help for *Relative age* and *Work*, but they also improve prediction performance for *Childhood country* and *Gender*.

At the speaker level, classification is more difficult and we do not see improvement for all attributes when using the additional features or joint decoders. However, at the context-window level we found that joint decoders improved over single attribute decoders in all cases, though using the additional features did not help for *Romantic*, *Family*, and *Gender*. When using single attribute decoding with the other attributes as features we found even

higher performance for four of the attributes. Interestingly, *Gender* still does not benefit from using extra features and simply knowing the values of the other speaker attributes gives the best result. The lowest accuracy overall is obtained for relative age, this can be partly explained by the lower baseline as compared to the other attributes, which is influenced by the fact that it has three possible values instead of two.

5.9 Conclusions

In this chapter, we addressed the task of classifying the attributes of an individual based on their conversations in a longitudinal dataset. We conducted analyses of several interaction aspects, including message content, speaker groups over time, and interaction during the conversation. We developed a bidirectional LSTM architecture that, in addition to message content, includes a variety of features derived from our analyses, covering the time-stamp of the messages, messaging frequency, psycholinguistic word categories, linguistic mirroring, and graph-based representations of interactions between people. Additionally, to account for scenarios where some attributes are known, we present experiments that evaluate the use of the other six speaker attributes when classifying the seventh.

Our experiments evaluated the accuracy of predictions at the context-window level, which used only a sequence of five messages for message content, as well as at the speaker level using a larger set of context windows from each speaker. We observed improvements in speaker level accuracy up to 8.7% and up to 13.9% accuracy on context windows. We explored the usefulness of each feature with an ablative study and compared two different methods of decoding. For the case of predicting someone’s relative age or whether or not they are a co-worker, classmate, or native from the same country, we see improvement at both levels. Our evaluations show improvement over a system that only uses one of these features at a time, as well as over a baseline system that relies exclusively on message content.

To the best of our knowledge, this is the first study on speaker attribute prediction using personal longitudinal dialog data that focuses on one persons’ interactions with many users. The code used to extract the conversations from social media, to interactively annotate speakers, and to perform the experiments presented in this chapter is publicly available,¹ so others can conduct analyses on their own data. This work was originally published in [194].

¹https://github.com/cfwelch/longitudinal_dialog

CHAPTER 6

Personalized Word Representations

In the previous two chapters, we saw the benefit of personalized data and showed how training models on personal data was more effective than training on data from random individuals. In this chapter, we examine how to learn word embeddings using personal data. Using these embeddings we can look at differences in word representations across people and further explore language prediction.

6.1 Introduction

Word embeddings have become ubiquitous in natural language processing applications. Usually, embeddings are trained from a large corpus of news or web data that contains writing from many sources and authors [130, 146]. These embeddings capture syntactic and semantic properties of the language of all authors who contributed to this corpus. In this chapter, we introduce personalized word embeddings, and examine their value for language modeling and authorship attribution.

Multi-source corpora provide large volumes of data, but they may not lead to the ideal representations for individuals. For instance, the word “hometown” may have a different representation for different individuals. For some, it may relate to words such as “hills,” “trees,” and “family,” whereas for others may be more strongly connected to “ocean,” “beach,” and “friends.” These personalized representations differ among individuals, and also differ from a more generic representation that often tends to capture words that are semantically related at concept level, such as “city,” “town,” or “place.”

In this chapter, we explore the idea of personalized word embeddings. We explore differences in personalized word representations using a corpus of English Reddit posts that contains a large number of posts per author. We use the embeddings to initialize a language model and show that personalization leads to better results than generic embeddings, and explore how embeddings differ across individuals, and which types of words capture

personal meaning the most.

Through exploring the use of personalized embeddings for language modeling, we make additional contributions regarding the initialization of language models using pre-trained embeddings. Many NLP applications, such as biomedical data and technical support, have 10-100 million tokens of in-domain data and limited computational resources for learning from it. How should we train a language model in this scenario? Recent work has focused on (1) small constrained datasets, such as the Penn Treebank [119] and WikiText-103 [129], and (2) vast resources with billions of words from the web used to train enormous models with significant computational requirements [159]. This leaves a gap: when a substantial amount of in-domain data is available, but computational power is limited. We show that for our target setting in English, initializing and freezing input embeddings using in-domain data can improve language model performance by providing a useful representation of rare words, and this pattern holds across several different domains. In the process, we show that the standard convention of tying input and output embeddings does not improve perplexity when initializing with embeddings trained on in-domain data.

We explore how initializing word embeddings using in-domain data can improve language modeling in English. Testing all valid configurations of weight tying, embedding freezing, and initialization, we find that the standard configuration is not optimal when rare words are present. Instead, the best approach is to initialize with in-domain data, untie the input and output, and freeze the input.

To understand this difference, we run a series of experiments to measure the impact of changing (a) the threshold for replacing rare words with a special symbol; (b) the source of data for initialisation; (c) the amount of training data for the language model; and (d) the hyperparameters for both the baseline and our proposed approach. We find that the improvement comes from improved representation of rare words. These findings are confirmed through experiments on four additional domains, with similar trends.

We also compare our approach to an n-gram language model and a large-scale transformer model. We find that if a large-scale transformer is inappropriate either for computational or modeling reasons, it is best to train an LSTM-based language model with as much data as possible and initialize the embeddings on all available in-domain data. In this chapter, we apply this LSTM-based approach to personalized language modeling.

One motivation for this work is personalized text generation, and its application to text auto-completion, speech recognition, and translation. In systems where a user types a text input, such as a phone keyboard, it can be used to suggest future words to the user so that text can be entered more quickly. Another application of personalized embeddings is dialog systems, where the systems could be trained to produce text that follows the

style of certain professionals (e.g., counselors, advisors). In addition, one can measure cooperation in dialog by the similarity of speaking style, suggesting that building a system that can match the style of an individual may lead to a more successful system. Finally, such personalized word representations and the applications they enable can bring about a better understanding of the personalized models that can be built and used by the large companies that collect and store a significant amount of personal data.

6.2 Personalized Word Embeddings

Definition. Personalized word embeddings are vector representations of words derived from the text produced by a single author. We use the text produced by a Reddit user s in their posts C_s to create their word embeddings. We apply the method described below to this set and produce an embedding matrix, $C_s \mapsto W_s^{|V| \times k}$, where V is the vocabulary and k represents the embedding dimension.

Joint Learning of Personal and Generic Word Embeddings. We jointly learn a generic embedding matrix and an embedding matrix for each author, inspired by Bamman et al. (2014) [11]. Each matrix $W \in \mathbb{R}^{|V| \times k}$ has a row for each vocabulary word and a k -dimensional vector for each embedding. The hidden layer is calculated as $h = w^\top W_{generic} + w^\top W_s$ where w represents the one-hot encoding of a word and s represents an author. This is a modified skip-gram architecture [130], which sums two terms so that back-propagation updates the generic matrix and a author-specific matrix. It allows the generic matrix to benefit from all data while learning author-specific deviations in the same space.

We use the set of messages from all 100 speakers to generate embeddings for all words that occur at least five times across all users. This yields a vocabulary of 177 thousand words. We learn 100-dimensional embeddings with an initial learning rate of 0.025 and a window size of five, using L2 regularization due to the increased number of parameters.

Dataset. We use data for the 100 most active users¹ in a corpus collected from Reddit.²

These users have from 49k to 249k posts, with 73k on average. Posts contain 29 tokens on average and come from 3.6k subreddits. The largest fraction (18.6%) belong to the

¹We excluded users that appear on a public list of bots (<https://www.reddit.com/r/autowikibot/wiki/redditbots>) or who appear to be automated based on manual inspection, as well as posts from the counting subreddit (<https://www.reddit.com/r/counting>), as these posts are mostly single numbers.

²https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/

User Example	Use	Nearest Neighbors
A	doctors think this is bad for her health ...	preventative, insurance, reform, medical, education
B	it is usually bad for your health ...	professional, mental, conduct, experiences, online
All	N/A	medical, preventative, insurance, safety, healthcare

Table 6.1: Nearest neighbors of “health” for two personalized embedding spaces and the generic space.

subreddit *AskReddit*; the next two largest are *blog* and *politics* with 4.8% and 4.7% of the posts respectively.

We use the set of messages from all 100 authors to generate embeddings for all words that occur at least five times (across all users). This yields a vocabulary of 177k words. We learn 100-dimensional embeddings with an initial learning rate of 0.025 and a window size of five, using L2 regularization due to the increased number of parameters (tuned in preliminary experiments). Using this method, we learn 101 embeddings for each word – a generic representation, and a separate representation for each user.

Reddit users have been found to be primarily male, young adults (under 30), located in the USA and primarily identify as christian or atheist [188]. It is possible that results presented in this chapter do not generalize to populations that differ significantly from the population of Reddit users. Future work may consider isolating the effects of topics and style by modeling subreddits for comparison though in this work we consider a personalized embedding as a representation that may capture both.

6.3 Differences Across Individual Word Representations and Usages

Individuals use the same word in different ways in different contexts. Examining these differences can give insight into individual topic and style preferences, or their word associations. To illustrate these differences, in Table 6.1, we show different ways that two users in our dataset use the word “health.” Although these words may be used in similar contexts, the meaning of, and topics associated with these words is often different for each user, which affects the words we would expect to come after it. These preferences are reflected in the top neighbors for the word “health” for each user.

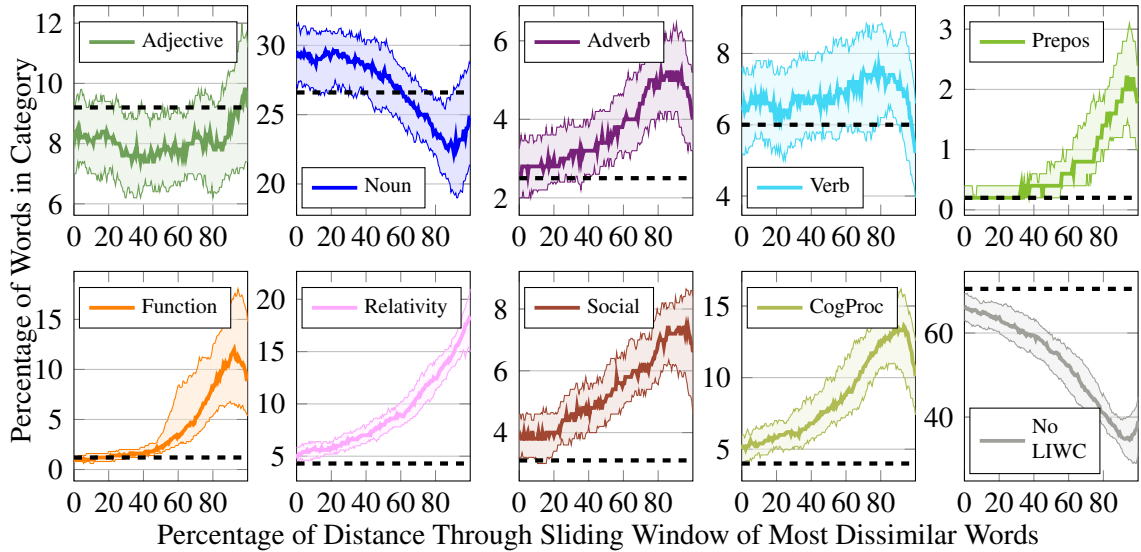


Figure 6.1: Relationship between embedding similarity and distribution of word categories across most dissimilar words showing the five grouped POS tags, nouns (NN, NNS, NNP, NNPS), adjectives (JJ, JJR, JJS), verbs (VBD, VBG, VBN, VBP, VBZ, VB), adverbs (RB, RBR, RBS), prepositions and subordinating conjunctions (IN), with the highest frequency and the four LIWC categories with the highest concentration in the set of dissimilar words (i.e. function, relativity, social, and cognitive process (CogProc) words). We use a sliding window through the most dissimilar words where 100% means the window contains the most dissimilar of all words. We also include the distribution of words which have No LIWC category assigned (No LIWC). The y-axes are scaled separately for each sub-plot and the sliding window on the x-axis shows word types ordered by average dissimilarity across embedding spaces. Top row groups words by their part-of-speech and bottom groups words by LIWC categories. We show average and interquartile range for values calculated across all users. Dashed horizontal lines show the average percentage of words that fall into each category when the window is slid over a randomly shuffled list rather than the sorted one.

To gain a deeper understanding of these differences, we analyze personal and generic embeddings for specific word groups based on the Linguistic Inquiry and Word Count (LIWC) lexicon [144] and part-of-speech tags. This analysis can help us understand what types of words tend to have different representations across users and are therefore more personal in nature. We part-of-speech tag the messages with the Stanford CoreNLP tagger [117, 182]. For each word in the vocabulary, we assign a tag if the tagger gives the same tag at least 95% of the time, otherwise the word is ignored. LIWC categories are looked up in the lexicon that contains words and word stems and a word may have multiple categories, in which case it counts toward each.

We look at the proportion of word types in the 5k most dissimilar words for each user. We define word similarity as the cosine distance between a word’s generic embedding and its author-specific embedding. Note that these are unique words and that the x-axis is the percentage of the way through the 5k dissimilar words, with the most dissimilar at 100%. We break this into subsets and look at how the distribution changes as we approach the most dissimilar words. A visualization is provided in Figure 6.1. We have added horizontal dashed lines to show the average number of words for each category when the words are random, rather than the most dissimilar. We find the set of most dissimilar words includes more function words, words relating to space and time (Relativ), cognitive processes (CogProc), and social words, as well as more adjectives and nouns. This suggests that these types of words may tend to have more personal usage than other types. These results are consistent with prior work that has found function words are effective for recognizing style and measuring style similarity [57] and for authorship attribution [8, 46, 134]. Though there are about a thousand occurrences of function words in the window of most dissimilar words, the three that occur most frequently are *become*, *near*, and *minus*. The concreteness of the words that accompany these words varies. The word *become* can refer to anything that is changing and can refer to physical objects, emotions, or ideas. *Near* can refer to physical proximity, but also in assessing any value (e.g. “anywhere near”, “nowhere near”). Similarly, the word *minus* can refer more concretely to subtracting a quantity but can also refer to the similarity of concepts (e.g. “it’s like X minus the Y”).

6.4 Low Compute Language Modeling with In-Domain Initialization

Normally, language models are initialized randomly, but in order to leverage the embeddings we learned, we propose initializing the language model’s word embeddings with

vectors trained on additional in-domain data. To make this most effective, we make two other key changes to training. First, we prevent embeddings from shifting during training. Without this, the embedding space could become inconsistent as vectors for words seen in training shift while those for words seen only in the additional data stay the same. Second, we do not tie the weights of the input embeddings and final output layer. To understand the impact of these factors, we train models with every valid combination of weight tying, freezing, and pretraining.³

We use AWD-LSTM language model from previous work as our baseline [127, 128]. It is an autoregressive language model that had state-of-the-art results by combining regularization techniques, has been widely used, and can be trained in under a day on a single GPU (without fine-tuning). Although more recent models can achieve better perplexities on standard benchmarks [25, 124], we find that not all models have code available or that they take far more time to run than Merity et al. (2018)’s model.

We are interested in experimenting with personalized embeddings, but for exploring the ideal low compute language modeling scenario, we experiment with several other domains. First, we train embeddings using GloVe on Gigaword. We use an embedding size of 400 and rare word cutoff of 5, the same as in the original AWD-LSTM model and GloVe respectively.⁴ All other GloVe hyperparameters were set as specified in the original GloVe paper and trained using the released code. For our main evaluation, we consider two versions of the Penn Treebank. *Std* is the standard version used in language modeling, with words of frequency less than five converted to UNK, all words lowercase, numbers replaced with a special symbol, and punctuation removed. *Rare* has the same preprocessing but without replacement of rare words.⁵ All of the hyperparameters were simultaneously varied, sampling all uniformly at random. We selected the final set based on validation perplexity. Following Dodge et al. (2019), we report the mean and variance as a function of the number of hyperparameter trials [35]. The result is shown in the following plot, where the orange line is the baseline and our approach is the blue line:

³Note, for frozen output embeddings the bias is not frozen.

⁴We ran experiments with 200 dimensional embeddings and found the same trends, but all results were slightly worse.

⁵ The script to generate our *Rare* data from the LDC release is available at: <http://jkk.name/emnlp201m/>.

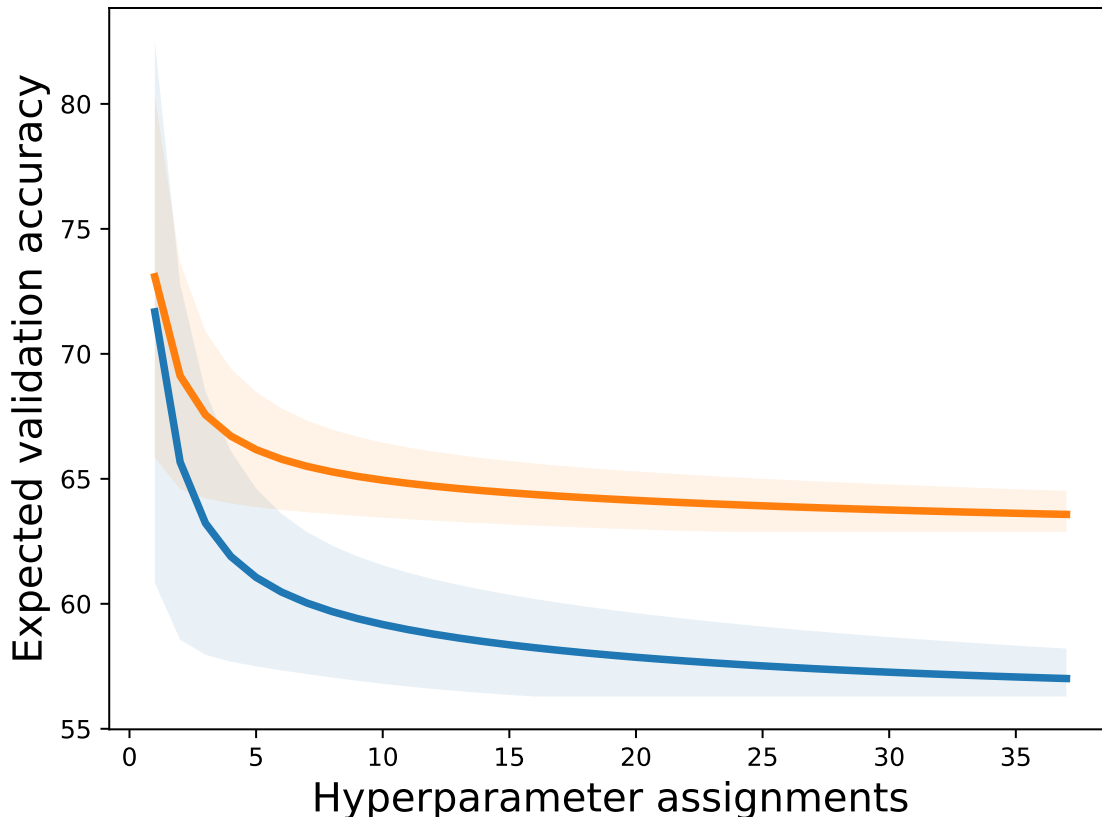


Table 6.2 shows the results, with icons to concisely describe the different configurations.⁶ Looking first at the standard evaluation set, we can see the value of pretrained embeddings by considering pairs where the only difference is whether the embeddings are random or pretrained. Pretrained embeddings are better in all but one case (comparing the fourth last and second last rows), and there the difference is only 0.5. As for freezing the pretrained input embeddings, keeping all other aspects the same, it is always better to freeze them. When we untie the embeddings it does increase the number of parameters.

There are also four clear sections of performance in the table: (a) frozen random output embeddings; (b) frozen pretrained output embeddings; (c) frozen random input embeddings; (d) various configurations. These results have an asymmetry. Freezing the output embeddings consistently leads to poor performance, even with pretrained embeddings pretrained. In contrast, freezing with pretrained input embeddings leads to some of the best results. We expected freezing with random initialisation to perform poorly, but the drop is modest for input freezing and dramatic for output freezing. This suggests that the two embedding matrices are serving different purposes in the model. The results do support the practice of tying when the input embeddings are random, but the benefit is half as large

⁶ Dice Icon by Andrew Doane from the Noun Project. Fire and Snowflake Icons by Freepik from www.flaticon.com.

when they are pretrained.











For the dataset with rare words we see mostly the same trends. The exception is the bottom six rows. Once rare words are present, random initialisation of the input embeddings is considerably worse than pretraining (third last row). Again, there is an asymmetry between input and output, with the top five models all using pretrained input embeddings, but only three of them using pretrained output embeddings. Tying is also no longer the best approach, with the top three models not tying. Our proposed approach, using pretrained untied embeddings and freezing the input, has the best results.

The only difference between Std and Rare is the lack of UNKs in Rare. This impacts 5.1% of tokens in the validation set (33% of types). While our pretrained embeddings do not cover all of these rare words, they do cover most. The vocabulary from Gigaword that we build vectors for covers 99.5% of the validation word tokens in Std (98% of word types), and 98.8% of the validation word tokens in Rare (84% of word types).

6.5 When and Why Does Pretraining Help?

To understand the strengths and limitations of this new approach, we consider a series of experiments, each probing a specific variable. To simulate our target scenario, we use 44 million words of Wall Street Journal data from the North American News Corpus (NANC) [58]. This provides enough data for pretraining, training, validation, and test sets all in the exact same domain (not even varying the newspaper). We apply similar preprocessing as in the previous section, but break the data down into articles rather than sentences and keep rare words.

We compare the six best configurations from Table 6.2. In all cases, output embeddings are not frozen, so we leave out the symbol. We use only one symbol for pretraining/random because both embeddings are the same in most cases. The exceptions have $\frac{\text{Ⓢ}}{\text{Ⓡ}}$ to indicate pretrained input and random output.

-   Standard approach.
-  $\frac{\text{Ⓢ}}{\text{Ⓡ}}$ Our approach, but with random output embeddings and without freezing.
-   Standard approach + pretraining.
-   Our approach, but without freezing.
-   Our approach.
-  $\frac{\text{Ⓢ}}{\text{Ⓡ}}$ Our approach, but with random output embeddings.

	Tied	Embeddings		Dev PPL			
		Input	Output	Std	Rare		
(a)	■	❄️	📦	❄️	📦	680	1120
	▣	❄️	📦	❄️	📦	680	1120
	▣	🔥	📦	❄️	📦	680	431
	▣	🔥	🌐	❄️	📦	220	372
	▣	❄️	🌐	❄️	📦	218	360
(b)	▣	❄️	📦	❄️	🌐	121	202
	▣	🔥	📦	❄️	🌐	95.0	170
	▣	🔥	🌐	❄️	🌐	91.3	147
	■	❄️	🌐	❄️	🌐	90.7	136
	▣	❄️	🌐	❄️	🌐	90.7	136
(c)	▣	❄️	📦	🔥	📦	82.2	143
	▣	❄️	📦	🔥	🌐	81.4	142
(d)	▣	🔥	📦	🔥	📦	65.3	120
	▣	🔥	📦	🔥	🌐	64.1	113
	▣	🔥	🌐	🔥	📦	62.5	105
	▣	🔥	🌐	🔥	🌐	61.7	98.5
	▣	❄️	🌐	🔥	🌐	61.6	97.1
	■	🔥	📦	🔥	📦	61.3	112
	▣	❄️	🌐	🔥	📦	61.1	98.1
	■	🔥	🌐	🔥	🌐	59.8	98.7

■ = Tied parameters ▣ = Untied parameters
 ❄️ = Frozen in training 🔥 = Unfrozen in training
 📦 = Random init. 🌐 = Pretrained init.

Table 6.2: Perplexity on the PTB for all valid combinations of weight tying, freezing, and pretraining. Results are sorted by perplexity on Std and shown to three significant figures.

Train Config	Domain				
	NANC	Cord	IRC	Reddit	Wiki
■ 🔥 📦	106	135	41.3	186	206
■ 🔥 📊	103	125	41.1	166	174
■ 🔥 🌐	97.2	121	39.8	154	142
■ 🔥 🌐	95.7	111	39.2	152	141
■ ❄️ 🌐	90.8	109	37.3	146	144
■ ❄️ 📊	90.5	112	37.6	152	161

Table 6.3: Results for various domains. All other results in this section are for NANC.

Other Domains Show the Same Pattern. First we consider varying the domain to make sure this is not an artifact of news data. Table 6.3 shows results on Covid-19 research [185], Ubuntu IRC chat [99], Reddit, and Wikipedia, tokenized with either Scispacy [136] or Stanza [157]. Pretraining consistently helps, while freezing is best on all but Wikipedia. Our approach is consistently either the best or very close to the best.

The Improvement is Due to Rare Words. To probe the impact of rare words, we explore replacing them with UNK (using the same UNK symbol as used in embedding pretraining). We consider four variations, each constructed in two steps. First, we make a list of the words in the original training set and how many times each one occurs. Second, we make modified versions of the training and validation sets, replacing words with UNK if their count in our list is lower than K . For this step, any word that does not appear in our list is treated as if it has a count of zero. We consider $K = 0, 1, 2$ and 5 . K is 0 for all other experiments in this section, which means that no words are replaced with UNK. When K is $1, 2,$ and 5 , the introduction of UNKs means all words in the validation set are seen during language model training.

Table 6.4 shows a clear trend: the benefit of our approach grows as more rare words are present (i.e., K is smaller). Note, it may seem odd that perplexity is higher when $K=1$ than when $K=0$ since we have removed rare words. This is probably because when K is 1 there are UNKs in the validation set but not in the language model training set.

Table 6.5 shows statistics about rare words in the datasets. 71-83% of word types in the training sets occur fewer than five times, but most of these appear frequently in the pretraining sets (compare the first column with the second last column). The same pattern occurs for word tokens. Comparing the statistics for the training set and the pretraining set, the percentage of rare word types is fairly consistent while the percentage of rare tokens

Train Config	Frequency Cutoff			
	0	1	2	5
■ 🔥 🎲	106	106	70.6	55.4
■ 🔥 📊	103	104	72.5	56.8
■ 🔥 🌐	97.2	99.9	68.1	54.1
■ 🔥 🌐	95.7	97.8	70.2	56.0
■ ❄️ 🌐	90.8	92.1	66.5	54.5
■ ❄️ 📊	90.5	91.5	65.8	54.0
UNK Types Dev	0%	13%	21%	33%
UNK Tokens Dev	0%	2.3%	3.4%	5.5%
UNK Types Train	0%	0%	40%	68%
UNK Tokens Train	0%	0%	1.4%	4.1%

Table 6.4: Varying the minimum frequency to not be converted into an UNK. The top half shows language model perplexity. The bottom half shows the percentage of word tokens and types that are replaced with UNK in each case.

Dataset	Train		Pretrain		Train in Pre	
	Type	Tok	Type	Tok	Type	Tok
PTB	73	5.3	77	0.11	14	1.3
NANC	71	4.8	63	0.49	13	0.63
Sci	78	6.3	85	1.2	23	1.6
IRC	83	4.2	90	1.3	37	1.4
Reddit	81	6.1	86	0.69	15	0.71
Wiki	78	7.3	78	0.36	5.6	0.43

Table 6.5: Percentage of word types and tokens that occur five times or fewer in each dataset. The last two columns are the percentage of types/tokens in the training set that occur five or fewer times in the pretraining set. For PTB the pretraining set is Gigaword (as used in Table 6.2).

Train Config	Pretraining Source			
	NANC 43M	Gigaword 5B	GloVe 6B 42B	
■ 🔥 🌐	97.2	90.9	93.1	93.3
■ 🔥 📰	103	99.3	98.3	99.5
■ 🔥 🌐	95.7	90.0	91.2	93.6
■ ❄️ 🌐	90.8	90.6	91.1	91.5
■ ❄️ 📰	90.5	90.7	90.7	91.9

Table 6.6: Varying similarity and size of pretraining data. Dataset size is shown below the name of each dataset.

consistently goes down.

Pretraining Data Needs to be from a Similar Domain. We would expect that the effectiveness of pretraining will depend on how similar the data is. Table 6.6 shows results with different embeddings, and indicates the number of words used in pretraining. We see that the value of additional data depends on the domain. Gigaword is also news text and is able to improve performance. The larger GloVe datasets use Wikipedia and Common-Crawl data, which is a poorer match and so does not improve performance. For GloVe we did have to change the embedding dimensions from 400 to 300, which may impact performance slightly.

The Effect Persists When Language Model Training Data is Increased. So far we have only used the additional in-domain data for pretraining. In this experiment, we expand the training set for the language model. We try two variations, one where the data is an exact domain match (NANC) and one where it is also news, but from different newspapers and from a different year (Gigaword). Table 6.7 shows that as we increase the amount of data our approach and the variant with random output embeddings continue to do best, but the margin shrinks between them and the standard approach. Note, however, that these results are with hyperparameters tuned for the baseline configuration. With tuning the 0.7 gap between our proposal and the baseline for 4xNANC widens to 6.6.

Hyperparameter Tuning Further Improves Results. All of the previous experiments were slightly tipped in favour of the baseline as we used the hyperparameters from [128]. We do not have the resources to tune for every condition, so instead we focus on a final set of experiments with the 4xNANC condition from Table 6.7. We run 37 configurations with

Train Config	NANC WSJ			Gigaword		
	1x	2x	4x	1x	2x	4x
■ 🔥 📦	106	81.0	67.5	106	92.5	86.3
■ 🔥 📦 /	103	83.3	68.7	99.3	91.7	87.2
■ 🔥 🌐	97.2	80.4	67.8	90.9	88.6	85.7
■ 🔥 🌐	95.7	80.0	68.1	90.0	86.4	85.5
■ ❄️ 🌐	90.8	73.7	66.8	90.6	84.8	82.5
■ ❄️ 📦 /	90.5	72.9	66.1	90.7	83.8	83.7

Table 6.7: Expanding the language model training set.

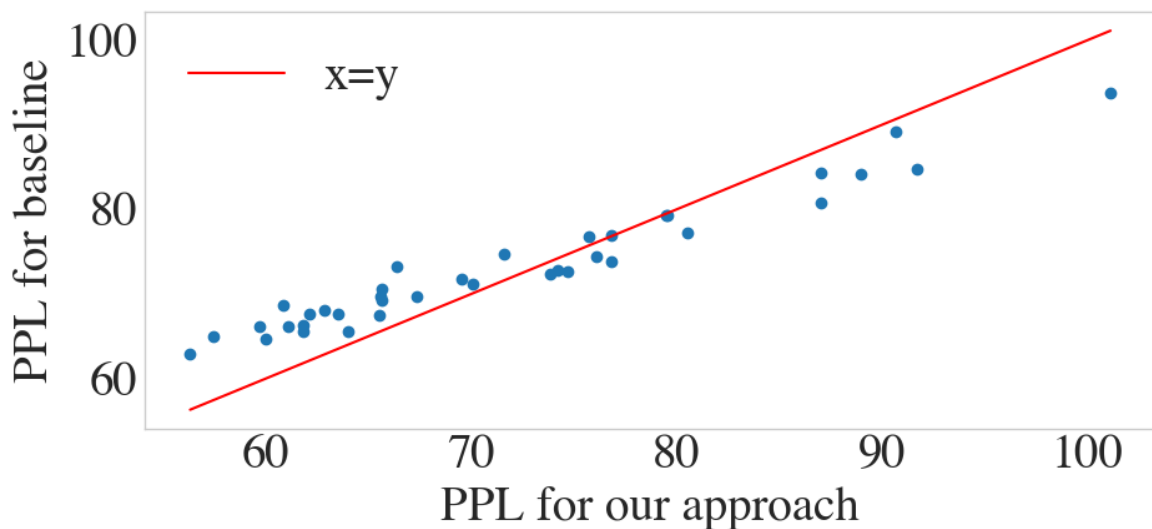

















Figure 6.2: Hyperparameter search results with one point for each configuration. The line separates where our approach is better (left) or worse (right).

randomly sampled hyperparameters, using the same configurations for the baseline and our proposed approach (see the supplementary material for details). Figure 6.2 shows that our approach is even stronger after tuning, with a score that is 6.6 better than the baseline. Comparing the baseline and tuned hyperparameters, some shifted substantially more than others: the learning rate was halved; word dropout was halved; and the number of layers was increased from 3 to 4. The other parameters shifted by 15-30%.

Initialization Without Additional Data This experiment considers a variation where we pretrain the embeddings only on the PTB (i.e., there is no additional pretraining data). LM uses the embeddings produced by training a baseline model (i.e., train an LM, then reset all parameters except the embeddings and train again).

Tie	Input	Pretraining Method	Val PPL
		GloVe	64.5
		GloVe	66.2
		LM	61.7
			61.3
		GloVe	60.5
		LM	60.3
		LM	59.4

While pretraining on the training data improves performance here, the improvement does not persist through the finetuning stage.

Test Results Confirm Our Observations. Using the best configuration we train the baseline and our proposed approach using 8xNANC (the most our GPU could support). We compare to an n-gram language model trained on all of the NANC data [66], and a transformer based model trained on a massive dataset, GPT-2 [159]. While GPT-2 cannot be retrained in a low-compute scenario, it can be used. We compare to GPT-2 without fine-tuning. We evaluate byte-pair encoding (BPE) separately because with BPE tokenisation models have additional information when predicting the second or later piece of a token [126].

Table 6.8 shows that for word-level prediction, our approach improves over the baseline and an n-gram language model. BPE breaks up rare words, leading to no improvement over the baseline and while we do better than the 112m parameter GPT-2, we do not do as well as the 774m parameter one (both untuned). Overall, this indicates that for users who require

Model	Words		BPE	
	Dev	Test	Dev	Test
N-Gram	92.3	95.0	56.7	55.3
GPT-2 (112m)	-	-	46.4	43.8
Baseline AWD-LSTM	52.8	53.5	37.8	36.7
Our approach	49.0	49.4	38.3	37.2
GPT-2 (774m)	-	-	32.5	33.7

Table 6.8: Final results, training with 8xNANC.

word-level scores and have limited computational resources our approach is an effective way to use additional data when training LSTM language models.

6.6 Language Modeling with Personalized Embeddings

To use our personalized embeddings, we modify the architecture to take as input the concatenation of the personalized user-specific embedding and the generic embedding for each word. We use the same hidden layer sizes and drop-out rates as in their original AWD-LSTM experiments, but untie the weights of the encoder and decoder, and verified that this gave better performance in preliminary experiments. Embedding dropout is a type of dropout that randomly removes entire word vectors from the input, rather than dimensions, and scales the remaining embeddings by $\frac{1}{1-e}$, where e represents the embedding dropout probability [45]. In our setting, we copy the mask and apply it to both generic and personalized embeddings.

The embeddings are trained on all available user data, but the more computationally expensive language models are not. We use a subsample of our dataset with 1,000 posts for each user and an 80/10/10 split for training, validation, and testing. The same splits are used for generic and personalized models, varying only the embedding layer.

To measure the ability of our models to predict the next word, we use two metrics: (1) mean reciprocal rank (MRR), calculated as one divided by the rank of the correct word choice in the descending list of next word probabilities and averaged over all instances, and (2) perplexity.

Single User Embeddings. We also consider an approach in which just one vector is learned for each user (rather than one for each user-word pair). This is an approach widely used in previous work [95, 106]. This user vector is concatenated to the generic word embedding.

The results in Table 7.6 suggest that using the combined personalized and generic em-

Model	LM		Attribution	
	MRR	PPL	MRR	Accuracy
Merity et al. (2018)	0.364	65.53	0.452	34.8%
Single User Vectors	0.361	66.70	0.450	34.9%
Personalized Embeddings	0.371	62.43	0.462	36.1%

Table 6.9: Results for Language Modeling (LM) and Authorship Attribution. Personalized word embeddings significantly improve performance (permutation test, $p < 0.0001$).

beddings improves performance significantly over single vector user representations and over generic embeddings. We note that the number of parameters of the LSTM input is not the same when comparing the baseline Merity et al. (2018) model to the other cases. We ran an additional experiment doubling the size of the embeddings for the baseline and found that the perplexity improved to 64.21, although the personalized embeddings still significantly outperformed this baseline.

We can also analyze the accuracy when predicting words belonging to particular parts of speech and LIWC categories. Our intuition is that a model that uses personalized word embeddings would be better at predicting words belonging to the four LIWC categories whose words are most distant from the generic space. Table 6.10 shows that for almost all categories, the personalized word embeddings lead to the best performance, although for relativity words, single user vectors performs slightly better and for drives, our methods do not outperform the baseline.

6.7 Authorship Attribution

We also use a language model trained with personalized word embeddings to perform the task of authorship attribution.⁷ We build a language model for each author using a sample of 10k posts for training and 1k for validation. We then hold out another sample of 1k posts to use for authorship attribution. The language models for all authors are separately run on the held out set, and the model with the lowest perplexity is then chosen as the assigned author. Table 7.6 shows there is a statistically significant improvement for our personalized embeddings method. This is a difficult task with 100 classes, so the accuracy is low, but the MRR suggests that the correct author is usually in the top 3 model choices.

⁷Note that we do not consider datasets such as [90] because they do not provide the volume of data needed for our approach and our goal is to compare generic and personalized embeddings, not to set a new state-of-the-art.

Part-of-Speech	Merity et al. (2018)	Single User Vectors	Personalized Word Embeddings
DT	19.1	19.4	18.9
IN	30.9	34.2	30.7
JJ	703.6	708.2	681.2
NN	632.5	621.4	597.7
PR	23.4	23.7	22.6
RB	146.6	148.7	143.7
VB	65.4	65.8	62.1
PUNCT	10.9	11.1	10.2
OTHER	72.6	73.9	70.2

LIWC Category	Merity et al. (2018)	Single User Vectors	Personalized Word Embeddings
Affect	88.3	85.6	82.7
Biology	93.9	95.7	92.4
Cognitive Process	73.4	75.8	73.3
Drives	69.8	75.0	70.7
Function	68.8	71.2	67.6
Informal	35.3	36.0	35.3
Perception	95.9	90.7	85.4
Relativity	28.7	28.1	28.5
Social	48.7	48.9	46.7

Table 6.10: Perplexity results broken down by the type of target word (top showing part-of-speech and bottom showing high-level LIWC category), with the best result in bold. OTHER includes all other unlisted part-of-speech tags.

6.8 Limitations and Ethical Considerations

Our work in this chapter treats an individual user’s style as determined by both the subject matter as well as the way that it is expressed. Previous work has treated these as separate aspects of an individual’s language [26, 57] and although we did analyze function word usage, further work could explore how well personalized language is captured by testing on held-out subreddits or otherwise more carefully controlling the distribution of topics during training.

In applying our method to text prediction systems, users may experience unintended negative effects. For instance, embeddings may become unintentionally biased toward language that becomes inappropriate when later suggested in another context. Additionally, users who are learning a language may bias embeddings toward improper language use, reinforcing errors and making it more difficult for the user to learn the language. It may be appropriate to use our embeddings if users consent and are made aware of the possible consequences of doing so.

It is possible that our method could be used for authorship attribution and surveillance of individuals online [176]. Such an application risks potential discrimination, coercion, and threats to intellectual freedom [162]. Personalized language models could also be used to develop a tool that tells the user who their writing most resembles, or if their writing resembles their past writing, with the objective of obfuscating the author’s identity [154]. A tool like this could also be used maliciously to impersonate a particular author. Although we believe the difficulty of this task currently makes these minor risks, we advocate against the use of our methods for these tasks.

Our method requires more computation and memory than the baseline method we compared to. The additional computation is relatively small, as learning the embeddings takes around 3 hours using 30 threads on a machine with 16 Intel Xeon Silver 4108 CPUs. The memory required to store embeddings for N users is $N + 1$ times the amount of storage required by a generic matrix only.

6.9 Conclusion

In this chapter, we explored personalized word embeddings. Using a large corpus of Reddit posts, we generated personalized word embeddings for 100 individuals, and performed analyses of the differences between personalized and generic embeddings for specific groups of words. We showed that using personalized word embeddings to initialize a language model improves perplexity over a model that uses generic word embeddings, or a

model that only learns single vectors for each user as has been frequently done in previous work. We showed that initializing embeddings with vectors trained on in-domain data can improve performance by providing better representations for rare words and that this effect persists even as more in-domain data is used to train the language model. Our work also suggests that standard model components like embedding tying should be retested as we continue to explore the space of language modeling. Further, we showed that the embeddings can be used to improve performance on authorship attribution. We cannot release the data due to licensing restrictions but our code is available online with instructions for how to obtain and process the data in order to support future work on personalization. This work was originally published in [189, 192].

CHAPTER 7

Personalization with Limited Data

The previous chapter explored the production of personalized word representations by using large quantities of written text from each individual. In this chapter, we look at how to model an individual’s language when not as much text is available. We attempt this with two different methods. The first is an explicit modeling of the differences between speakers using demographic variables. These values can be used to group speakers who may speak in similar ways and gives us a way to find related data for a given individual. Second, if a small amount of initial text is available for a new user, we can use that data to find similar users. Our task is then to determine which similarity metric to use and, once we have determined which users are similar, how to leverage data from these similar users in order to construct a personalized language model for the new user. Both methods use data related to a given individual’s writing to assist in personalized language modeling.

7.1 Introduction

As discussed in previous chapters, word embeddings are used in many natural language processing tasks as a way of representing language. These embeddings capture syntactic and semantic properties of the language of all individuals who contributed to the corpus. However, they are unable to account for user-specific word preferences (e.g., using the same word in different ways across different contexts), particularly for individuals whose usage deviates from the majority. These individual preferences are reflected in the word’s nearest neighbors. The example from the previous chapter, in Table 6.1 shows the way two users use the word “health” and the word’s five nearest neighbors in their respective personalized embedding spaces. The word is used in similar contexts, where contextual embeddings may give similar representations, but it has different salient meanings in the personal space of each user. User A tends to talk more about preventative care and insurance, while user B tends to talk about people’s experiences affecting their mental health.

The typical approach in natural language processing (NLP) is to use one-size-fits-all language representations, which do not account for variation between people. This may not matter for people whose language style is well represented in the data, but could lead to worse support for others [100, 120, 143]. While the way we produce language is not a direct consequence of our demographics or any other grouping, it is possible that by tailoring word embeddings to a group we can more effectively model and support the way they use language.

A static representation of a word’s meaning is limited, as meaning varies given context and extra-linguistic information. Hofmann et al. (2020) find that in some contexts “testing” refers to seeing if a device works and “sanitation” refers to a pest control issue, while in another context both refer to conditions of the COVID-19 pandemic [69]. Garimella et al. (2017) look at location and gender and how they affect associations with words like “health” and many other stimulus words like “stack”— does it make you think of books or pancakes? Similarly, “wicked” may mean “evil” or may function as an intensifier depending on where you live [11].

These factors all contribute to word representations that determine word associations and thus affect how language models predict what someone will say next. In this chapter, we refer to personalized language modeling as the task of constructing a language model for an individual that captures these effects and better predicts what that individual will say. This task has applications to predictive text, authorship attribution, and can be used to model the style of an individual or profession (e.g. therapist, counselor) for dialog systems.

Additionally, personalized embeddings can be useful for applications such as predictive typing systems that auto-complete sentences by providing suggestions to users, or dialog systems that follow the style of certain individuals or professionals (e.g., counselors, advisors). They can also be used to match the communication style of a user, which would signal cooperation from a dialog agent.

To implement and evaluate our proposed method, we build a large corpus of Reddit posts from 61,981 users for whom we extract self-reported values of up to four demographic properties: age, location, gender, and religion. We examine differences in word usage and association captured by the demographics we extracted and discuss the limitations and ethical considerations of using or drawing conclusions from this method. We explore the value of compositional demographic word embeddings on two English NLP tasks: language modeling and word associations. In both cases, we show that our proposed embeddings improve performance over generic word representations.

Similarly, language models do not take into account the differences between individuals and their language patterns, and are not optimized for personalized use. Approaches like

fine-tuning can be used to tailor a pretrained model to an individual, but perform well only when enough data is available, which is often not the case.

We propose compositional demographic word embeddings as a way of building personalized word embeddings by leveraging data from users sharing the same demographic attributes (e.g., age: *young*, location: *Europe*). Our proposed method has the benefits of personalized word representations, while at the same time being applicable to users with limited or no data, as long as demographic information is available.

Then, we consider the case of users with a small number of available tokens and propose ways to (1) find similar users in our corpus and (2) leverage data from similar users to build a personalized language model for a new user. We hypothesize that data from similar users should be able to outperform standard fine-tuning or tuning on a similar amount of random data. We explore the trade-offs between the amount of available data from existing users, the number of existing users and new users, and how our similarity metrics and personalization methods scale. We then show an analysis to explore what types of words our method predicts more accurately and are thus more important to consider when applying personalization methods.

7.2 Demographic Embedding Models

We examine two methods for creating demographic embeddings. The first uses the joint method discussed in Chapter 6, where we learn generic word representations, and separate word embedding matrices for each demographic value [11]. We run this once for each of our demographic attributes. Secondly, we want to see if we can instead learn a single vector representation of each demographic value, rather than a separate vector for each value for each word, as a more memory efficient representation of demographic shifts of the meanings of words. We do this in a similar joint learning setup where we take an authors text and construct skip-gram windows with knowledge of the demographic attributes of the author. We sum the input word embedding with a vector representing the authors age (i.e. young, old), with a location vector (e.g. USA, UK), with a religion vector (e.g. christian, atheist), and with a gender vector (i.e. male, female), and use the resulting vector to predict the output word.

7.3 Dataset

We use Reddit comments as in the previous chapter but also focus on finding speakers with demographic information. The 100 authors discussed in the previous chapter can be used as

anchor users from which we can calculate similarity to new users. For experiments on users without demographic information, we experiment with two settings: In the **small anchor** setting, there are 100 *anchor* users, each having 200k tokens as training data, 25k tokens as validation data and 25k tokens as test data, and 50 *new* users, each having 2k tokens as training data, and 25k tokens for each of validation and test. In the **large anchor** setting, there are 10k *anchor* users and 100 *new* users, each having 2k tokens each for training and validation and 20k tokens for test. We use anchor users to construct the personalized language models and new users to validate and test our models.

Using Reddit comments is advantageous in that we are able to get a large amount of data for each speaker which will allow us to compare a model trained on an individual's text to models which only use their demographic information. The downside is that users do not have profiles as they do on other social media websites and there are no categorical fields from which to extract this information as other sites like Twitter and Blogger.

To resolve this, we manually come up with patterns to extract demographic information from Reddit data. This has successfully been done in previous work to construct a set of Reddit users who were diagnosed with depression [198] and to construct personas for personalized dialog agents by extracting statements users make about themselves [121].

We use English Reddit comments as they are publicly available, are written by many users, and span multiple years.¹ We extract demographic properties of users from self-identification in their text.

7.3.1 Finding Demographic Information

Reddit users do not have profiles with personal information fields that we could scrape. Instead, we developed methods to extract demographic information from the content of user posts.

In order to determine what kind of information we can extract about users, we performed a preliminary analysis. We manually labeled a random sample of 132 statements that users made about themselves. We specifically searched for statements starting with phrases such as 'i am a' or 'i am an'. In our sample: 36% clearly stated the user's age, religion, gender, occupation, or location; 34% contained descriptive phrases that were difficult to categorize like 'i am a big guy' or 'i am a lazy person'; and 30% mentioned attributes such as sexual orientation, dietary restrictions, political affiliations, or hobbies that were rare overall.

¹https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/

Based on our analysis, we decided to focus on age, religion, gender, occupation, and location as the main attributes.² These were extracted as follows:

Age. We extracted the user’s age using a regular expression.³ During this process, we found users that were matched to different ages due to the corpus covering user activity across several years. In those cases, we removed users whose age difference was greater than the time span of our corpus. Additionally, we excluded users who said they were less than 13 years of age, as this violates the Reddit terms of service. We decided to split the age into two groups, young and old at a threshold of 30, as this split was used in previous work [161], and it gave a reasonable split for our data and the data we used for testing word associations [50].

Gender. Gender was extracted by searching for statements referring to oneself as a ‘boy’, ‘man’, ‘male’, ‘guy’, for male, or ‘girl’, ‘woman’, ‘female’, ‘gal’, for female. Manual inspection revealed some users indicated that they were of both genders. In that case, if one gender occurred less than one fifth of the time we took the majority of the reported gender, otherwise we removed the user from our dataset. We acknowledge that this approach excludes transgender, gender fluid, and a range of non-binary people, and may misgender people as well (see § 7.11 for further discussion of these issues).

Location. To obtain location information, we searched for phrases such as ‘i am from’ and ‘i live in.’ Next, whenever either the next token is (1) tagged as a location by a named entity recognizer [117], (2) a noun, or (3) the word ‘the’, we select all subsequent tokens in the phrase as the user location. Manual inspection of matches showed that Reddit users are not consistent in the granularity of reported location. Statements included cities, state, province, country, continent, or geographical region. Based on the number of users per country, we decided to merge some countries into region labels while leaving others separate. This resulted in the following set of regions: USA, Asia, Oceania, UK, Europe, Canada. We further matched location statements to lexicons to resolve the location to one of these regions, removing common relative location words.⁴ For larger population regions of Canada and the USA, we match statements using state abbreviations, province names, highest population cities, and in the USA we also match the capital cities. For other regions we only match the highest population cities as there were too many cases to cover.

Religion. To extract religion, we searched for the five largest global religious populations,⁵

²We attempted to extract occupations, but found they were difficult to identify and group because there are many different occupations, many ways of stating one’s occupation, and many ways to describe the same occupation.

³`.*?(i am|i\’m) (\d+) (years|yrs|yr) old[^\e].*?`

⁴northern, western, eastern, southern, downtown, suburbs

⁵From <https://www.adherents.com/>, although note that since our study the domain name has been hijacked by a payday loans service. The site is archived by the Library of Congress at <https://www.>

counting ‘secular’, ‘atheist’, and ‘agnostic’ as one non-religious group. We used a regular expression⁶ and filtered users who stated beliefs in more than one of these five groups.

7.3.2 Preprocessing

Reddit data can be noisy, containing URLs, structured content (e.g. tables, lists), Subreddit-specific emoticons, generated, or deleted content. We first extract all posts for a each user in our dataset. During this process we removed noisy posts following these rules: (1) it contains more than 20 tokens but the average token length is less than 3; (2) it contains a long token whose length is greater than 30; (3) it contains less than 8 tokens among which more than 3 are URLs; (4) it contains more than 3 math related symbols, such as “[”, “+” and “=”; (5) it contains coding related symbols like “{”, “}” and “()” with only white spaces in the parentheses; (6) it contains less than 5 tokens and the last token is ”*” (This kind of post is usually a spelling correction to a previous post); (7) there are less than 4 unique tokens in every sequence of 8 adjacent tokens; (8) it contains hashtags, indicated by: [] (/ / #; (9) it is a duplicate of another post in the user’s data; and (10) more than 60% of the characters are non-alphabetical. After the filtering step, we removed markup for emojis and hyperlinks from the remaining posts (keeping the posts themselves). We took these steps to ensure that we were capturing language used by the authors, rather than reposts, collections of links, ASCII tables and art, equations, or code. Tokens that occur fewer than 5 times are replaced with “UNK,” which results in a vocab size of 55k for the small anchor set and 167k for the larger one. Additional examples are shown in Table 7.1.

7.3.3 Post-processing

The resulting dataset was further filtered to remove known bots.⁷ For the demographic data we consider two subsets. First, the set of users for which all four attributes are known (4Dem). With this set we perform ablation experiments on the number of known attributes in a controlled manner. However, it is important to note that this set may not be representative of most users on Reddit, as it focuses on users willing to divulge a range of demographic attributes. Our second sample addresses this by including users for whom we identify two or more of the demographic attributes (2+Dem). Statistics for these sets are described in Table 7.2, along with the training, development, and test splits used for the language modeling experiments.

[loc.gov/item/lcwaN0003960/](https://www.reddit.com/r/autowikibot/wiki/redditbots)

⁶.*(?i am|i\`m) (a)?(christian | muslim | secular | atheist | agnostic | hindu | buddhist).*?

⁷<https://www.reddit.com/r/autowikibot/wiki/redditbots>

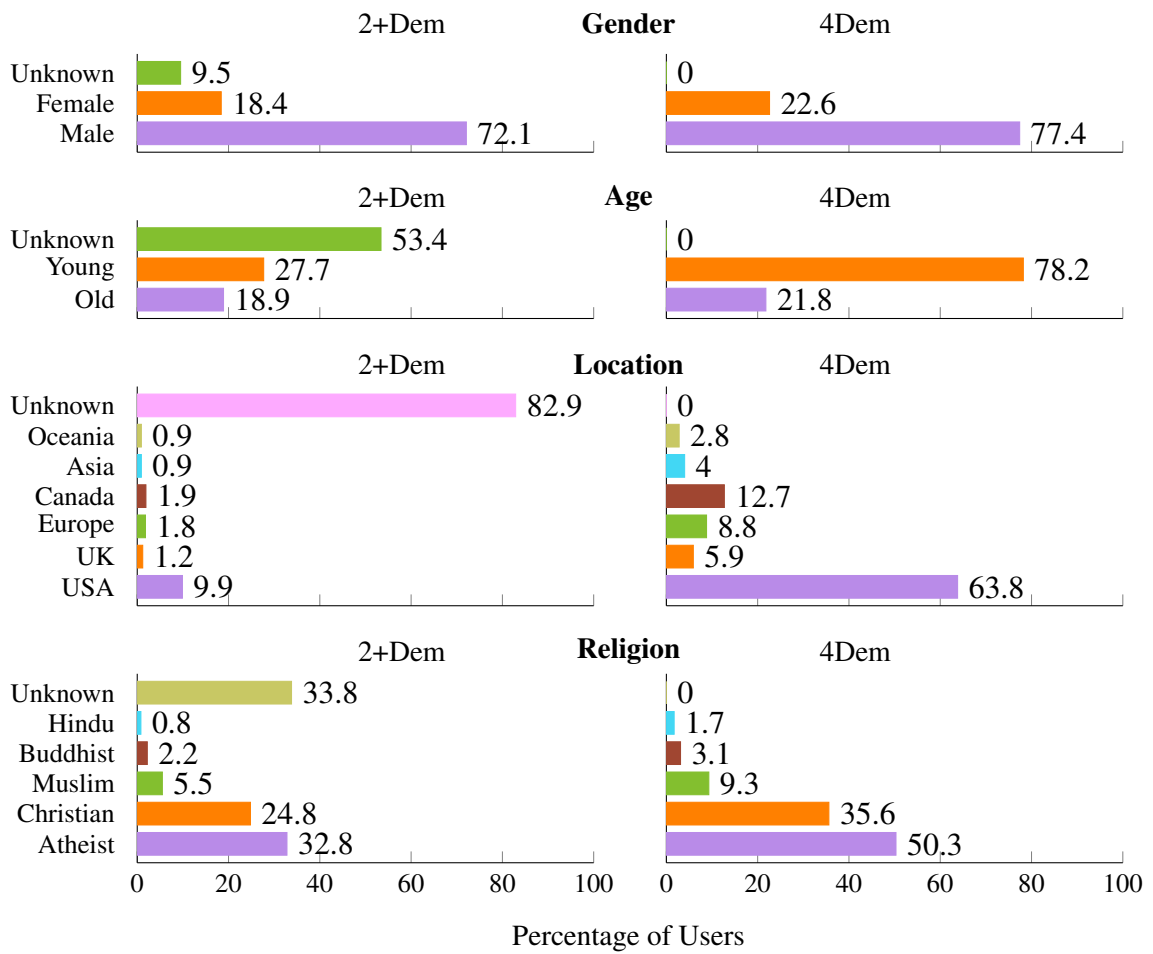


Figure 7.1: Distribution of the four demographic attributes in our two datasets, showing the set with all demographics known on the right and the random sample from those with at least two known on the left.

Rule	Example
(1) it contains more than 20 tokens but the average token length is less than 3	" i " " " " " w " " i " " l " " l " " " " " n " " e " " v " " e " " r " " " " " g " " i " " v " " e "
(2) it contains a long token whose length is greater than 30	(There is usually duplication inside this kind of post)
(3) it contains less than 8 tokens among which more than 3 are URLs	URL URL URL URL
(4) it contains more than 3 math related symbols, such as "=", "+", and "!"	before humanity , maybe 2 +2 = 5 . no , before humanity 2 +2 = 4 did not exist .
(5) it contains symbols like "{", "}" and "(" with only white spaces in the parentheses	we specialize in () () () () () () ()
(6) it contains less than 5 tokens and the last token is "*"	(This kind of post is usually a spelling correction to a previous post)
(7) there are less than 4 unique tokens in every sequence of 8 adjacent tokens	w , w , w , w , would n't it be better if we just bend over and follow their rules ?
(8) it contains hashtags, indicated by: [] (/ / #	[** if i were a rich man ... **] (/ / #ggj)
(9) it is a duplicate of another post in the user's data	
(10) more than 60% of the characters are non-alphabetical.	

Table 7.1: Examples of rules for filtering posts.

The distribution of demographic values for each of these sets is shown in Figure 7.1. Looking at the set of all users in our data who have at least two known demographic attributes (2+Dem), we find that 83% of the time location is unknown. Age and religion are the next most frequently missing at 53% and 34% respectively. Gender is more likely to be known than the other attributes: only 10% of users in this subset have an unknown gender. In a manual evaluation of all our extracted attribute labels for the 100 users, we found accuracies of 94% for location and gender, 78% for religion, and 96% for age.

In order to verify the accuracy of our demographic attribute assignment, we manually annotated a sample of 100 users from the dataset. Our extraction of attributes with regular expressions and rules was meant to have high-precision. It is likely that more attributes marked 'unknown' by our extraction could be filled in upon manual inspection. We evaluate the retrieved attributes for these 100 users by viewing the set of all posts that matched our extraction rules and attempting to annotate age, religion, gender, and location. The

Set	Users	Posts
All Reddit	13,213,172	1,430,935,783
2+Dem		
Total	61,627	205,394,970
Training	34,110	50,000
Validation	9,190	10,000
Test	9,143	10,000
4Dem		
Total	354	3,433,062
Test	354	10,000

Table 7.2: Statistics for two Reddit sets: with at least two demographic attributes (2+Dem), or all four demographic attributes (4Dem). Training, development, test splits used in the language modeling experiments are also shown. First row shows overall number of posts and users from the entire set of Reddit posts.

Attribute	Values	Word	Neighbors
Gender	Male	blush	blushing, smile, chortle, swoon, snicker, wince, chuckle, blushes, smirk, guffaw
	Female		brow, eyeshadow, bronzer, nars, nyx, lipstick, mascara, primer, concealer, highlighter
Age	Younger	health	regen, mana, aid, permanent, condition, treatment, mental, preventative, benefits, medical
	Older		care, reform, healthcare, education, coverage, high-deductible, socialized, medical, insurance, condition
Religion	Christian	embodying	exalting, creaturely, extols, mysteriousness, idolization, magnanimity, asceticism, imbuing, unalterable, mortification
	Atheist		unionism, mercantilist, american, corporatocracy, un-free, proletarian, environmentalist, wage-slavery, communistic, free-marketeers
Location	USA	america	europe, country, canada, sweden, mexico, china, india, africa, usa, britain
	Canada		original, tv, worst, hot, space, actual, body, home, move, nation

Table 7.3: Examples of words with low overlap in nearest neighbors, showing how meaning can differ across the values of a demographic attribute.

annotation instructions were to identify the value of these four attributes based on the annotators interpretation of the text of the posts. Then, for cases where the extracted attribute is not ‘unknown’, we calculate the percentage of times that they are the same. We get 94% for location and gender, 78% for religion, and 96% for age. It should also be noted that despite the annotators best efforts, it is not possible to know the actual ground truth values.

7.4 Generating Compositional Demographic Word Embeddings

We propose two methods for learning compositional demographic embeddings. The first learns a generic embedding for each word and a vector representation of each demographic attribute (including ‘unknown’). This is memory efficient, as we need only 19 vectors to cover all of our attributes. In the second method, for each word we learn (a) a generic embedding and (b) a vector for each demographic attribute. This is more expressive, but requires twenty vectors for each word.

7.4.1 Demographic Attribute Vectors

In this approach we jointly learn a matrix for words and a separate vector for each demographic value. The word matrix $W \in \mathbb{R}^{|V| \times k}$ has a row for each word in the vocabulary and a k -dimensional vector for each embedding. The demographic values can be represented by another matrix $D \in \mathbb{R}^{|C| \times k}$, where C is the set of all demographic values (e.g., male, female, christian, USA). The hidden layer is calculated as $h = W_h^T (W_w + C_g + C_l + C_r + C_a)$ where w represents the one-hot encoding of an input word and g, l, r, a represent the demographic values of the speaker. This is a modified skip-gram architecture [130] with a hierarchical softmax, which sums five terms so that back-propagation updates the word representation as well as the demographic values.

We use posts from all users to train embeddings for words that occur at least five times across all users. This yields a vocabulary of 503k words. We learn 100-dimensional embeddings with an initial learning rate of 0.025 and a window size of five.

7.4.2 Demographic Word Matrices

When learning demographic matrices we separately run our skip-gram model for each of the demographic attributes (e.g., gender) and learn a generic word matrix $W_G \in \mathbb{R}^{|V| \times k}$ and a value specific word matrix for each value, v , of the given attribute, A , (e.g., male, female)

$W_v \in \mathbb{R}^{|V| \times k}, \forall v \in A_v$. This changes the hidden layer calculation to $h = W_h^T G_w + W_h^T W_{vw}$, with hidden layer weights W_h , and the model then learns a generic word representation, in matrix G , while learning the value specific impact on the meaning of that word.

Differences Across Demographic Embeddings. In order to understand what our embeddings capture, we examine words that have different representations across demographics. We can look at the nearest neighbors of a given query word across the embedding spaces for different demographics. We perform this analysis on both the demographic matrices and vectors, finding less variation in the neighbors when using demographic vectors, making them less interesting. We show examples of words with low overlap in nearest neighbors for demographic matrices in Table 7.3. These show the differences in word meaning across groups.

7.5 Language Modeling

We first examine the usefulness of our embeddings by showing that they can help us better model a user’s language. We consider two experiments. First, we focus on compositional demographic embeddings and sample 50k posts from our corpus for training the language model and 5k for each of validation and test. Next, we compare with a user-specific model on a sample of our data with text from just 100 users who each have a large amount of data available in our corpus, with an average of 3.2 million tokens per user.

In both experiments, we use the language model developed by Merity et al. (2018) [127, 128]. As discussed in Chapter 2, this model was recently state-of-the-art and has been the basis of many variations. We modify it to initialize the word embeddings with the ones we provide and to concatenate multiple embedding vectors as input to the recurrent layers. The rest of the architecture is unaltered. We tried adding rather than concatenating and found no improvement. We chose to concatenate the inputs with the intuition that the network would learn how to combine the information itself.

We explored various hyperparameter configurations on our validation set and found the best results using dropout with the same mask for generic and demographic-specific embeddings, untied weights, and fixed input embeddings. Untying and fixing input embeddings is supported by concurrent work [192]. Each model is trained for 50 epochs. We use the version from the epoch that had the best validation set perplexity, a standard metric in language modeling that measures the accuracy of the predicted probability distribution.

Model Type and Input Size	2+Dem	4Dem
Baseline, 100	123.8	124.6
Baseline, 500	125.1	126.1
Generic Words Only, 100	116.0	112.1
Generic Words Only, 500	115.8	112.6
Demographic Vectors, 200	116.7	113.0
Demographic Matrices		
+ Age Only, 200	109.4	110.3
+ Gender Only, 200	109.4	109.9
+ Location Only, 200	109.7	112.9
+ Religion Only, 200	110.9	112.0
+ All Demographics, 500	107.7	109.1

Table 7.4: Perplexity on the demographic data. Our demographic-based approach improves performance. The difference between the last row and generic words is significant ($p < 0.00001$ with a permutation test).

7.5.1 Demographic Perplexity Evaluation

Table 7.4 shows results for our demographic personalization methods, which are designed to handle new users for whom we have demographics but not much text data. The first method, demographic vectors, performs no better than generic embeddings. This is surprising since prior work has achieved success on a range of tasks with this kind of representation (see Chapter 2, Section 2.4.4). We suspect that for language modeling the variations are too fine-grained to be captured by a single vector. However, demographic matrices do improve significantly over generic embeddings. A model with all demographics improves the most, but we also see improvements when only one demographic value is known.

The LSTM hidden layer size is the same across models, but the change in the input size affects the total number of parameters. To control for this, we ran our baseline model and model initialized with generic words with a larger input size, matching the number of parameters in our best models. As shown in Table 7.4, this increase in parameters does not improve performance.

7.5.1.1 Ablation Experiments

Table 7.4 shows results when using no demographics (top 4 rows), one demographic at a time (rows 6-9) and all four demographics (row 10). Each attribute improves perplexity, with age and gender improving it more than location and religion.

Additionally, we perform a breakdown of the performance of our demographic matrices

Att.	Value	2+Dem		4Dem	
		0D	4D	0D	4D
Age	Young	107.1	103.6	110.6	108.0
	Old	115.1	111.1	114.0	112.0
	Unknown	112.3	108.6	-	-
Location	USA	108.5	105.7	108.1	105.1
	Canada	135.7	132.6	110.0	107.6
	Oceania	111.0	108.7	114.8	112.8
	Europe	130.0	128.2	133.0	130.1
	Asia	109.3	108.6	145.3	145.4
	UK	115.3	113.5	96.9	96.9
	Unknown	111.0	107.1	-	-
Religion	Christian	116.5	111.9	108.5	105.9
	Atheist	106.4	103.2	112.7	109.9
	Muslim	112.7	108.5	109.5	108.4
	Hindu	122.4	115.6	158.1	159.5
	Buddhist	114.1	111.7	116.4	114.1
	Unknown	122.3	109.1	-	-
Gender	Male	113.5	109.2	115.2	112.6
	Female	100.9	97.8	102.7	100.4
	Unknown	122.3	118.5	-	-

Table 7.5: Perplexity for language models with no demographics (0D) or with all four demographic matrices (4D) with results broken down by demographic values.

language model on each of the demographic groups. These results are shown in Table 7.5. We do see worse performance on some minorities as compared to other groups for the same model, although that is not always the case (gender, for instance, shows better perplexities for female than for male, and Muslim shows lower perplexity than Christianity, which has substantially more data). When we use the demographic word embeddings in our model, we are able to improve performance for all demographic groups, including minorities.

We also find that the performance on the ‘unknown’ group increases in all cases with our largest improvement on ‘unknown’ religion. The unknown is explicitly modeling people in our dataset who have either (1) stated this demographic information with a value that we model but not in a way that our regular expressions identify, (2) stated this demographic information with a value that we do not model, or (3) have not stated this demographic information. In the second case, the effect is that it is useful to know which demographic groups the speaker does not belong to. In the third case, it may be that not sharing this particular piece of information (while sharing other personal information) says more about what the speaker will tend to say.

7.5.2 Comparison with User Representations

For users with a lot of data, it is possible to train a user-specific model, with embeddings that capture their unique language use. We would expect this to be better than our demographic embeddings, but also only be feasible for users with a lot of data. This experiment compares our demographic approach with a user-specific approach.

We create a model for each user using the sample that has a large amount of data for 100 users (3.2 million tokens each on average) as done in concurrent work [189]. We tried two approaches, user vectors and user matrices, which are analogous to our demographic vectors and matrices. The difference is that rather than having a separate vector / matrix for each demographic we have a separate vector / matrix for each user. Our split sizes for language model experiments are the same as the demographic experiments.

Results. Table 7.6 shows results for generic embeddings, user vectors, user matrices, and demographic matrices. We find that user vectors, as have been used widely in previous work [95, 106], do not improve performance. Both our demographic and user matrices improve performance over generic embeddings with comparable performance. While we chose 100 users with a lot of data, they had less data than the amount used to train each demographic specific model. The relationship between the amount of data, its similarity to a user’s writing, and the effect on performance is an interesting open question.

Model	PPL
Generic Word Embeddings	63.94
User Vectors	68.98
User Matrices	61.69
Demographic Matrices	61.80

Table 7.6: Comparing our demographic-based approach with two user-specific approaches. Perplexities are generally lower than previous tables because the threshold for rare words being made UNK was higher.

7.6 Demographic Word Associations

Method	best		oo3		oo10	
	India	USA	India	USA	India	USA
C-SGM	0.09	0.03	0.14	0.07	0.19	0.10
Ours G	0.18	0.21	0.18	0.40	0.31	0.63
Ours G+D	0.17	0.19	0.16	0.39	0.32	0.64

Method	best		oo3		oo10	
	Male	Female	Male	Female	Male	Female
C-SGM	0.13	0.16	0.20	0.20	0.25	0.26
Ours G	0.17	0.17	0.22	0.26	0.35	0.42
Ours G+D	0.18	0.20	0.22	0.27	0.37	0.45

Method	best		oo3		oo10	
	Younger	Older	Younger	Older	Younger	Older
C-SGM	-	-	-	-	-	-
Ours G	0.18	0.18	0.19	0.26	0.31	0.42
Ours G+D	0.19	0.21	0.19	0.29	0.32	0.44

Table 7.7: Comparison of demographic-aware word association similarities for our embeddings using (G)eneric or (G)eneric+(D)emographic, and the best results of the two variants of the composite skip-gram model (C-SGM) from Garimella et al. (2017). We show improved results for USA, India, Male, and Female, and provide new results using age for Younger than 30 and Older.

Metric	Embeddings	Gender						Location							
		M	IN	M	IN	M	IN	M	US	M	US	M	US		
best	Generic	0.178	0.164	0.209	0.223	0.171	0.175	0.198	0.213	0.191	0.175	0.180	0.207	0.225	0.197
	Age	0.175	0.169	0.211	0.239	0.167	0.180	0.207	0.225	0.197	0.180	0.207	0.227	0.198 [†]	0.176
	Age + Gender	0.175	0.169	0.211	0.239	0.174	0.181	0.207	0.227	0.198 [†]	0.176	0.180	0.180	0.187	0.176
	Age + Gender + Location	0.163	0.161	0.187	0.198	0.158	0.176	0.180	0.187	0.176	0.158	0.180	0.180	0.187	0.176
oo3	Generic	0.116	0.105	0.205	0.271	0.118	0.126	0.216	0.365	0.190	0.118	0.126	0.216	0.365	0.190
	Age	0.119	0.111	0.207	0.284	0.121	0.137	0.221	0.378	0.197	0.121	0.137	0.221	0.378	0.197
	Age + Gender	0.120	0.111	0.207	0.284	0.123	0.136	0.235	0.378	0.199	0.123	0.136	0.235	0.378	0.199
	Age + Gender + Location	0.131	0.117	0.214	0.265	0.125	0.145	0.234	0.383	0.203 [†]	0.125	0.145	0.234	0.383	0.203 [†]
oo10	Generic	0.217	0.194	0.346	0.440	0.209	0.231	0.364	0.588	0.324	0.209	0.231	0.364	0.588	0.324
	Age	0.230	0.205	0.362	0.456	0.227	0.246	0.395	0.615	0.342	0.362	0.246	0.395	0.615	0.342
	Age + Gender	0.230	0.205	0.362	0.456	0.227	0.250	0.389	0.614	0.342	0.362	0.250	0.389	0.614	0.342
	Age + Gender + Location	0.227	0.214	0.339	0.432	0.224	0.249	0.373	0.581	0.329	0.224	0.249	0.373	0.581	0.329

Table 7.8: Results on the 8 disjoint word association subsets for each combination of attributes. Similarities concatenate three embeddings that are each either generic, or specific to that demographic attribute. Overall, using age and gender in combination gives the best performance, though using all three is better on oo3. † indicates statistically significant improvement (permutation test, $p < 0.001$) over the next best model on the marked metric.

As a second evaluation, we consider word associations, a core task in NLP that probes the relatedness or similarity between words. Data is collected for the task by presenting a stimulus word (e.g., *cat*) and asking people what other words come to mind (e.g., *dog* or *mouse*). Earlier systems relied on resources such as WordNet to solve the task, but most recent work has used word embeddings.

Data. For our evaluation, we use data from Garimella et al. (2017). They constructed a word association dataset and experimented with learning separate word embedding matrices for different demographic groups. To collect the data, they (1) asked crowd workers to write one word associated with a single word prompt and (2) asked the workers their gender, age, location, occupation, ethnicity, education, and income. Only gender and location information was released, but the authors provided age information upon request.

Evaluation. As in prior work, we consider evaluation metrics defined in terms of: f_w , the number of people who listed word w for a stimulus; f_{max} , the highest f_w across all words chosen for a stimulus; and t , the number of participants given a stimulus. $best$ is f_w divided by f_{max} , where w is the word in the embedding space closest to the stimulus word; ooN (out-of-N) is $\sum f_w/t$ for the N words in the embedding space closest to the stimulus word; both are averaged over all stimulus words.

We consider two experiments. One directly matches Garimella et al. (2017), testing each demographic group separately [50]. Since our interest is in compositionality, we also introduce a setting where the data is split into eight disjoint sets, one for each combination of the three attributes.

Models. Garimella et al. (2017) proposed two methods, which we merge by taking the best result from either one. We considered only our demographic matrix embeddings as they performed best on language modeling. For the experiment with separate demographics, we use the appropriate embeddings. For the experiment with combinations of demographics, we concatenate the embeddings. We also compare to concatenation of generic embeddings learned for each attribute (this performs better than any individual generic embedding).

Results. Table 7.7 shows results on the single-demographic experiment. We achieve higher performance, but that may come from the change in training dataset.^{8,9} Table 7.8 shows results on the multi-demographic setting. We include only the best pair (age and gender) due to space. We have seen in earlier experiments that location does not perform as well as the other attributes and found the same trend here. Overall, composing demographic-based representations helps, with a combination of all three attributes consistently performing well on the *oo3* metric, while having two helps on the *best* metric. Generic embeddings only score the highest on one subset: Male, India, Young.

7.7 Experiments Without Demographic Information

Our method for constructing personalized language models consists of a similarity metric and a method for leveraging similar user data to train a personalized language model. The similarity metric measures which anchor users are most similar to a new user. That is, given a set users (*anchors*), a new user (*n*), and a similarity function (*sim*), we compute $z = \text{sim}(n, \text{anchors}); z \subset \text{anchors}$ to get a set of similar users *z*. We explore three similarity metrics and two methods of applying them to the construction of personalized models.

7.7.1 Calculating User Similarity

We explore three methods for measuring the similarity between users. Two of them, authorship confusion and user embeddings, are derived from classifiers trained for other tasks, while the third, perplexity-based similarity, is obtained from the performance of language models on the new user. The user embedding method results in a vector space where we can use cosine similarity to measure the distance between individuals. The perplexity directly gives a distance between each pair and the authorship confusion vectors can be treated as a vector of continuous values where each value represents the similarity to an anchor user.

Authorship Attribution Confusion (AA). Similarity can be measured from the confusion matrix of an authorship attribution model. This model takes a post as input and encodes it with an LSTM [68]. The final state is passed to a feed-forward layer and then a softmax to get a distribution over authors. We denote this model *A*, and $A(U)$ as the class distribution output by the model for a given utterance set. For a new user, we take their set

⁸ Their models are trained on 67.6m tokens of blog data, while ours are trained on 1,400m tokens of Reddit data.

⁹ We see a larger gain for the US than the IN evaluation. This may be because in our data location is unknown for many users and India is underrepresented (so much so that we aggregate it into all of Asia).

of utterances, U_n and pass them to our model $A(U_n)$ which will give us a confusion vector of length K , one value for each author.

We train this model on the data from anchor users. Embeddings are initialized with 200d GloVe vectors pretrained on 6 billion tokens from randomly sampled Reddit posts [146]. Models are trained on an NVIDIA GeForce RTX-2080Ti GPU and take 2.5 hours for $K = 100$ anchors with test accuracy of 42.88%, and 4 hours for $K = 10,000$ anchors with test accuracy of 2.42%. These accuracies are reasonably high given the difficulty of the task (Note that when $K = 10,000$ the majority class is 0.01%). The classifier does not have to be high performing given our application to computing a user similarity metric.

We apply this model to each post in the training data from *new* users. The scores produced by the model for each new post indicate which of the anchor users has the most similar writing. The more frequently posts from a new user are predicted as coming from a specific anchor user, the more similar this anchor user is to the new user.

User Embeddings (UE). We first train a language model with a user embedding layer on the data from anchor users. The model is adapted from [128] with an added user embedding layer. This token embedding layer is initialized with our pretrained GloVe vectors and frozen during training. The output of the LSTM layer is concatenated to the user embedding at each time step based on the author of the token at that time step. Our optimizer starts with SGD and will switch to ASGD if there is no improvement in validation loss in the past 5 epochs [150]. We removed continuous cache pointers to speed up training [59]. The model is trained on an NVIDIA GeForce RTX-2080Ti GPU. For $K = 100$ anchors, it took 132 hours. The validation perplexity converges to 59.06 and test perplexity is 58.86. When training with $K = 10,000$, we reduced the hidden LSTM size to 500, which reduced training time to 112 hours. The validation perplexity converges to 88.71 and test perplexity is 88.54.

The embeddings of anchor users can be obtained from the user embedding layer in the trained model. To learn the embeddings of new users, we freeze all parameters of the trained model except the user embedding layer. We train the model on the data from each new user separately with the same training strategy. It takes 2 minutes to learn the embedding of each new user. The average test perplexity is 66.67 when $K = 100$ and 90.48 when $K = 10,000$. For each pair of new user and anchor user, we use the cosine similarity between two embeddings as the similarity.

Perplexity-Based (PPLB). Given N trained language models, one for each of N users, we can then use the perplexity of one language model on another user’s data as a measure

of distance. We could compare the word-level distributions, though this would be very computationally expensive. In our experiments, we use the probability of the correct words only, or the perplexity of each model on each new user’s data.

We take the large language model trained on all anchor users, as described in the user embedding section and fine-tune it for each anchor user. This step is relatively inexpensive and takes a few minutes per user for both of our anchor sets. We then measure the perplexity of each model on the data of each new user. For this matrix of $new \times anchor$ perplexities, we turn each row, representing a new user, into a similarity vector by computing $1 - \frac{c - \min(row)}{\max(row)}$ for each cell, c . This step is more expensive, taking close to 24 hours for $K = 100$ and intractable given our hardware constraints in the $K = 10,000$ setting.

7.8 Leveraging Similar Users

Our three similarity methods provide a way to identify anchor users with the most relevant data for a new user. In this section, we describe two methods to learn from that data to construct a personalized model.

7.8.1 Weighted Sample Fine-tuning

Users who speak in a similar style or about similar content may be harder to distinguish from each other and should then be more similar. For a given similarity metric, we compute similar users and use data from these users to fine-tune a language model before fine-tuning for the new user.

We compare to two baselines, (1) a model trained on all anchor users with no fine-tuning and (2) a model trained on all anchor users that is fine-tuned on the new user’s data, as is done in standard fine-tuning. Our method of weighted sample fine-tuning has two steps. The first step is to fine-tune the model trained on all anchor users on a new set of similar users, as determined by our chosen similarity metric. Then we fine-tune as in the standard case, by tuning on the new user’s data.

7.8.2 Interpolation Model

Our interpolation model is built from individual language models constructed for each anchor user. It takes the predictions of each anchor user model and weights their predictions by that anchor’s similarity to the new user. No model updates are done in this step, which makes it immediately applicable, without requiring further training, even if the aggregation of output from all anchor models is more resource intensive.

Method	#Sim. Users	Δ Perplexity			Δ Accuracy@1		
		UE	AA	PPLB	UE	AA	PPLB
Weighted Fine-tuning	5	0.276	1.728	-0.627	0.159	0.155	0.148
Interpolation	100	-2.055	-2.415	-1.992	0.249	0.277	0.223
Interpolation	50	-2.163	-2.415	-2.043	0.260	0.277	0.204
Interpolation	25	-2.242	-2.415	-2.022	0.248	0.277	0.232
Interpolation	10	-2.286	-2.435	-2.183	0.235	0.260	0.249

Table 7.9: Difference in perplexity for our interpolated model and weighted fine-tuning results on the **small anchor** set. The baseline metrics are subtracted from our model, meaning that more negative perplexity and more positive accuracy are better. The baseline perplexity average is 64.3 for a model that uses standard fine-tuning. Bold indicates best performance.

#Sim. Users	Δ Perplexity	Std.Dev.
Random 10	0.176	0.367
10	-0.354	0.659
20	-0.534	0.977
30	-0.673	1.080
40	-0.714	1.040
50	-0.803	1.127
100	-0.941	1.351
150	-0.986	1.560
200	-1.069	1.549

Table 7.10: Difference in perplexity for fine-tuning varying number of similar users on the **large anchor** set, first fine-tuning on similar users, and second on the new user’s data, as compared to a baseline that fine-tunes on new user data only with perplexity 89.7. Each similar user has 2k tokens and each new user has 2k.

We also want to incorporate the predictions of the model fine-tuned on the new user data with the predictions of models trained on similar anchor users. We define a set of similar anchor users, σ , each of which has a similarity to the new user, n . We vary s for each similarity function. The weight to give the new user fine-tuned model is η , and we interpolate as follows for a given resulting probability p_r , of a word, w :

$$p_r(w|\cdot) = \eta p_n(w|\cdot) + (1 - \eta) \sum_{i \in \sigma} s(\sigma_i, n) p_{\sigma_i}(w|\cdot)$$

7.9 Results

We divide our results into separate subsections for each of the anchor sets. On the small anchor set we were able to perform more exploration of the weighted fine-tuning method, as it does not scale as well to the large anchor set.

7.9.1 Small Anchor Set

In this section, we compare our weighted sample fine-tuning and interpolation approaches to the more standard fine-tuning, where a large pretrained model is fine-tuned only on the new user’s data. With no fine-tuning our language model achieves a perplexity of 67.6 and when fine-tuning on the new user only, this perplexity drops to 64.3. For weighted fine-tuning, we attempt to fine-tune the large pretrained model on 100 anchors using our two step method, first fine-tuning on a million tokens from most similar users, and then fine-tuning on new user data. Through tuning the number of similar users, we found 5 worked best. For the interpolation model, we found more similar users improved accuracy, though perplexity was slightly higher for ten similar users. Our interpolation model combines predictions from similar anchor user language models. We have a language model fine-tuned to each of our anchor users and for a given new user we predict words by weighting the predictions of the models representing the most similar users.

Results in Table 7.9 show that our weighted sample fine-tuning is not able to outperform the baseline for any of our three similarity metrics. Perplexity and accuracy results are reported averaged over the test set users. We also tried fine-tuning with random user’s data and found that this performance was better than no fine-tuning but worse than fine-tuning on new user data only, showing that there is no added benefit from simply continuing to fine-tune on all data.

For the interpolation model, we tune η (see Section 7.8.2) on a held-out set and use a value of 0.7. The results show that the authorship attribution similarity performs best on both metrics. We find that as the number of similar users increases it has little effect past around ten similar users, as the similarity weights decrease and have a smaller effect.

It appears that having similar user data does not help the weighted fine-tuning model. To further investigate this we looked at settings where the amount of training data is fixed, but the source is either random, or a sample of similar user’s data. For each new user, we build six datasets: a random dataset and five datasets consisting of data from top-k similar anchor users for this new user where k is in $\{10, 20, 30, 40, 50\}$. Each of these datasets has 2m tokens. The random dataset is comprised of 20k tokens from each anchor user. For the dataset built from the top-k similar users, we want the number of tokens selected from each

#Similar		2k per Anchor			
Users	Perplexity	Acc @1	Acc @3	Acc @5	Acc @10
10	-0.692	0.097	0.111	0.100	0.058
5	-0.615	0.091	0.103	0.090	0.049
4	-0.590	0.088	0.091	0.079	0.045
3	-0.553	0.084	0.087	0.072	0.039
2	-0.415	0.084	0.060	0.052	0.033
1	-0.006	0.047	0.016	0.002	-0.002

#Similar		6k per Anchor			
Users	Perplexity	Acc @1	Acc @3	Acc @5	Acc @10
10	-11.726	0.497	0.697	0.723	0.718
5	-11.463	0.491	0.656	0.694	0.705
4	-11.287	0.486	0.650	0.677	0.684
3	-11.001	0.457	0.622	0.657	0.654
2	-10.604	0.439	0.588	0.602	0.617
1	-8.866	0.282	0.423	0.485	0.516

Table 7.11: Comparison of our interpolated user embedding similarity model on the **large anchor** set to a standard fine-tuned baseline measured in perplexity and accuracy @N. We show results for 2k and 6k tokens per anchor user, showing improved performance when more data per anchor is available. Bold indicates best performance.

anchor user to be proportional to the similarity between the new user and each anchor user. To do this, we normalize the three similarities by subtracting the minimum and dividing by the maximum such that they are between zero and one.

For a given set of k users and similarity metric, we sort all anchor users in descending order by their similarity to the new user and choose the top k anchor users. For the rank 1 anchor user a_1 , we choose the following number of tokens from the training data, where $s(\cdot, \cdot)$ is the similarity between a pair of users:

$$n_{a_1} = 2000k * \frac{s(newuser, a_1)}{\sum_{i=1}^k s(newuser, a_i)}$$

If $n_{a_1} > 200k$, we choose $n_{a_1} = 200k$. For the rank x anchor user a_x , we choose

$$n_{a_x} = (2000k - \sum_{j=1}^{x-1} n_{a_j}) * \frac{s(newuser, a_x)}{\sum_{i=x}^k s(newuser, a_i)}$$

tokens from their training data. If $n_{a_x} > 200k$, we choose $n_{a_x} = 200k$. We repeat this

fuels, qaeda, zealand, inte, al., antonio, facto, neutrality, kong, differ, olds, custody, cruise, obligation, arts, beck, guise, scrolls, vegas, mph, dame, conclusions, laden, pedestal, throne, ck, charm, occasions, disorders, correctness, disposal, capita, hominem, floyd, thrones, sarcastic, ghz, explorer, comprehension, standpoint, ambulance, noting, diego, accusations, cares, forth, enforcement, amp, nukem, convicted

Table 7.12: Top 50 words for which our best model outperforms the baseline based on the frequency of word correctly predicted normalized by the word’s total frequency.

procedure until the rank k anchor user. The ratio of similarities in this equation enforces that the amount of data we select from each of the top- k similar users is proportional to their similarity.

We then train a separate model on each dataset. The architecture of the model is the same as what is described in Section 7.7.1 except that it does not have a user embedding layer. It takes about 2.5 hours to train a model on a dataset on an NVIDIA Tesla V100 GPU. We then fine-tune the trained models on the training data of the new user, which takes about one minute on average.

For a chosen similarity metric and number k , we average the test perplexity of the fine-tuned models for all new users and subtract from it the average test perplexity of the fine-tuned models trained on random datasets, whose average perplexity is 111.0. The results are shown in Figure 7.2 with shaded areas indicating standard deviation. In the figure, the lower a point is, the better the datasets built using the corresponding similarity metric and number k is for training a language model for new users, which we infer is because the weighted sample datasets are closer to the data from new users.

We see that in terms of similarity metrics, the user embedding is the best while perplexity-based is the worst. As k increases, the performance first increases then decreases. The best performance is achieved when using the similarities calculated with user embeddings and using top 20 or 30 similar anchor users. After that, including more users has little effect, as their similarity weights continue to decrease. The main takeaway from this experiment is that although similar user data helps more than random data, the benefit does not transfer to the larger fine-tuning scenario. This area may be worth further exploring for fine-tuning strategies or for training data selection in applications where new models must be trained.

7.9.2 Large Anchor Set

In a set of only one hundred anchor users, it may be the case that existing users are not similar enough to the new user to benefit from our approach. To test this idea we ran

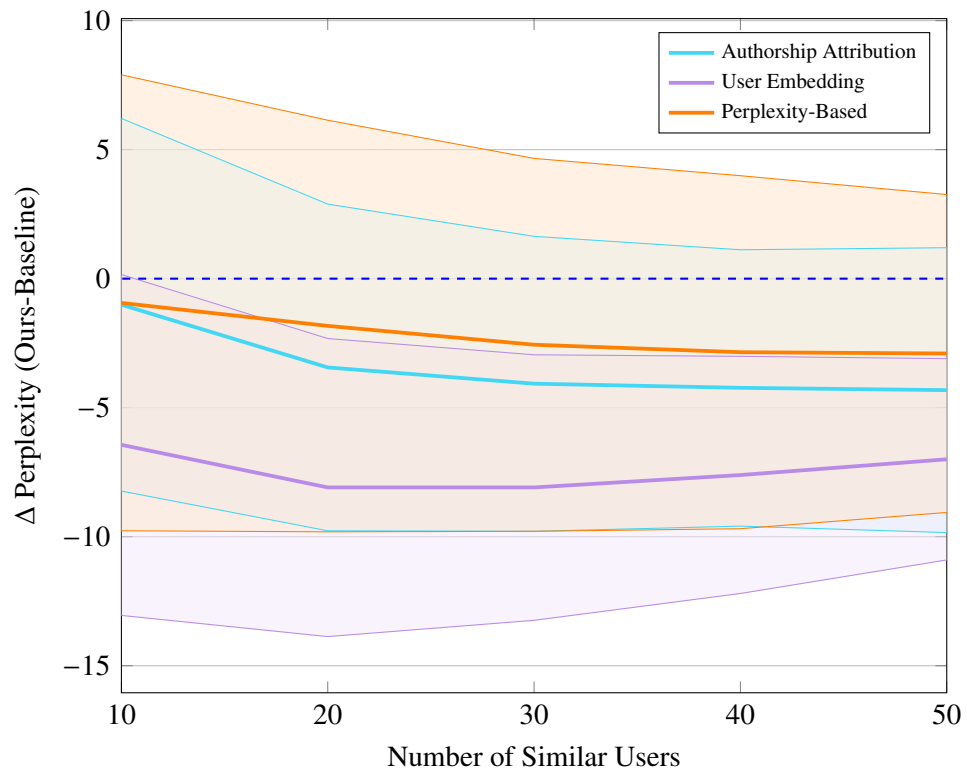


Figure 7.2: Change in perplexity for varying number of similar users considered in weighted fine-tuning for the three similarity metrics.

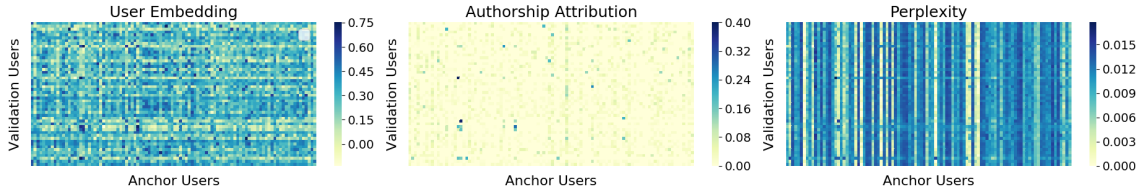


Figure 7.3: Heat maps showing normalized similarity for each metric on our 100 author anchor set.

additional experiments using the larger set of 10k anchor users and 100 new users.

Taking our most promising user embedding similarity metric from the weighted sample fine-tuning, we tested this method’s performance varying the number of similar users. Our results in Table 7.10 show a reduction in perplexity of 0.94 at 100 similar users and over one point at 200 users. There is a logarithmic improvement with the number of similar users considered, as we would expect more dissimilar users to be less informative. The results in this table suggest that the anchor set must be diverse enough to contain similar users to new users, in order to benefit from this method.

We also try the interpolation model with a larger set of anchor users. Our base model is trained on 10k anchor users and 2k tokens from each anchor. We fine-tune this model to each similar anchor user for weighting predictions. On a held-out set we tune η and find that in this setting performance starts to drop after around 10 similar users. It is computationally expensive to run each of the 10k models on each new user. The perplexity similarity metric requires that all of these are run in order to determine similarity and thus is not scalable to the large anchor user setting. The user embedding metric scales better because similarity can be determined by tuning an existing language model on new user data. For ten similar users we require 1,000 times fewer computations than we would to weight all 10k users. We found that authorship attribution performed much worse in this setting, as the confusion matrix becomes very sparse.

The results for our best similarity metric, user embeddings, are shown in Table 7.11. On the left we see performance for our model on the larger set containing 2k tokens per anchor user. For this analysis of our best, scalable model, we include accuracy @N, a metric denoting the percentage of times the correct word was in the top-N most probable choices. This is comparable to Table 7.10, where we used the same amount of data for the weighted sample fine-tuning approach. On the right we see performance when the amount of data per anchor user is tripled. The baseline and fine-tuned models all benefit from this additional data, however we find that the difference in perplexity is much larger, as having additional data will allow the models to learn more accurate similarity metrics. We also

Metric 1	Metric 2	Pearson’s r	Spearman’s ρ
UE	AA	0.360	0.362
UE	PPL	0.280	0.316
PPL	AA	0.073	0.025

Table 7.13: Spearman and Pearson correlation coefficients for each pair of similarity metrics (User Embeddings (UE), Authorship Attribution (AA), and Perplexity (PPL)) computed for each of our 100 anchor users similarity to each new user.

find that when tuning η it tends toward 0.6 when there are 2k tokens per anchor user but 0.3 when there are 6k. As the amount of data from the anchor users increases, the optimal interpolation weights shift to weight the anchor user models more heavily than the model fine-tuned on the new user. How the tuning of η could be done on a per-user basis, rather than globally, is an interesting open question.

7.10 Analysis of Similarity and Personalized Words

7.10.1 Differences in Similarity Functions

We looked at the differences between our three similarity functions by plotting heat maps shown in Figure 7.3. We find that the three metrics seem to capture different information about the relationships between users. The user embedding metric leads to more evenly distributed similarities, while the other two metrics seem to have outlier anchor users that show stronger correlation with a subset of the new users. We compute the correlation coefficients for Spearman’s ρ and Pearson’s r in Table 7.13. Interestingly, the perplexity and authorship attribution metrics correlate much more strongly with the user embedding metric than with each other. It is possible that the user embedding metric performs best in our experiments because it contains more of the useful information from both of the other metrics.

7.10.2 Personalized Words

We take the highest performing model using user embedding similarity trained on our large anchor user set and compare it to our baseline model to look at which words are more accurately predicted. By taking the number of times each word is correctly predicted by the best model when the baseline was wrong and dividing by the total number of occurrences of that word in our language modeling data, we can find words that have the highest normalized

frequency of being improved by our model.

The top 50 words for which we see improvement are shown in Table 7.12. We see many proper nouns in this set that are made up of two tokens. Many names can start with “San” or “Las” and so we see “vegas”, “diego”, and “antonio”, in this list. Similarly, “new” precedes “zealand” and other location names. The top word is “fuels”, which occurs often in the data in conversation about “fossil fuels”, though there are also many others that mention other kinds of fuels, or use “fuels” as a verb, as in “it fuels outrage”. We also see that units such as “mph” or “ghz” are more accurately predicted. The units that one chooses may be more common depending on where one lives, or in the case of “ghz” it may depend more on the subject matter that a user is familiar with or tends to talk about. Other proper nouns such as “game of thrones”, or “hong kong” vs. “donkey kong”, contain common words, which individually may be hard to predict, but with knowledge of an individual’s preferences could be predicted more accurately.

7.11 Limitations and Ethical Considerations

This work uses demographic information to modify language representation. This type of work is encouraged by the numerous arguments outlined in [147], which demonstrate the need for demographic data disaggregation in order to make decisions and build technologies that are equitable for all. We view our work as an initial investigation of differences in language model performance across demographics and how technology can be improved for the identified groups. Our results in Tables 7.4 and 7.5 show that using demographic information can enable the development of language tools that improve performance for all groups compared to simply training on all data.

Although we show that some language production aspects are correlated with demographic information, we do not believe the way we speak is a direct and only consequence of one’s demographics, neither do we claim that this is the ideal information source for it or that this will necessarily hold for populations sampled significantly differently than in our study. As a consequence, it is possible that using demographics in embedding construction could accentuate bias, although this remains to be studied. Those that use our method should account for this possibility.

Our study uses four demographic variables and only covers a subset of the potential values of each demographic. For instance, we do not use the same granularity across locations, include all locations, religions, or gender identities. We simplify age into ranges. The groups ‘secular’, ‘agnostic’, and ‘atheist’ are grouped into one broader group. Our sample is further biased by the choice of platform as each platform contains text from different

populations. Users in our sample are predominately young, male, atheist, and live in the United States.

When using gender as a study variable, we followed the recommendations of Larson (2017) [102]. Our “gender” extraction method does not refer to biological sex. After running gender extraction patterns, users are assigned to either the ‘male’, ‘female’, or ‘unknown’ label, meaning that on the basis of these phrases one’s gender identity is assumed to be binary or to be a gender identity unknown to our model, which may include those who are transgender, non-binary, or those who do not wish to disclose their gender. However, we are aware that the use of regular expressions for the extraction of demographic attributes can lead to false positives and false negatives and that there exists a bias in using these strategies, as populations that do not wish to be identified are less likely to explicitly make such statements. For transparency, our released code includes the scripts used to assign demographic labels.

Above we discussed concerns for incorrect demographic assignment when developing models. There are also potential negative consequences when using these models in a deployed system. Our embeddings can only be used when the demographics of a user are known. This may be acceptable if the user voluntarily self-reports their demographics with the understanding that they will alter the predictions they receive. However, if demographics are automatically inferred there is a risk of misattribution, which depending on the application may have negative consequences.

A separate consideration is the environmental impact of this approach. Compared to the standard method, our approach does involve training more models, but the cost of inference is likely only marginally higher. We believe the additional cost in training is worth the benefits to individual users.

Finally, we acknowledge that components of our method could potentially be used for user profiling [160] and/or surveillance of target populations, thus exposing members of underrepresented groups to harms such as discrimination and coercion and threatening intellectual freedom [162]. Similarly, the language models could be used to generate text in the style of a target population or at least to estimate the label distribution of a given text, which would help obfuscate the identity of the author [154]. This obfuscation could help hide an author’s identity in order to avoid surveillance or could be used maliciously to infiltrate communities online. We advocate against the use of our methods for these or other ethically questionable applications.

7.12 Conclusions

In this chapter, we addressed the issue of language modeling in a low data setting where a new user may not have enough data to train a personalized language model. We first proposed a novel method of generating word representations by composing demographic-specific word vectors. Through experiments on two core language processing tasks, language modeling and word associations, we showed that demographic-aware word representations outperform generic embeddings. We also found that demographic matrices perform much better than demographic vectors. Through several ablation analyses we showed that word embeddings that leverage multiple demographic attributes give better performance than those using single attributes. To support future work that can help model individuals and demographics, our code is publicly available. Our data is not available due to licensing restrictions but can be redownloaded and processed with our scripts. We hope this will support work on solutions for NLP applications and resources that can better serve minorities and underrepresented groups.

When demographics are not available, we looked at how to find similar users based on small samples of their writing. We considered three similarity metrics and two methods of leveraging data from similar “anchor” users to improve the performance of language modeling over a standard fine-tuning baseline, and showed how our results vary with the amount of data available for anchor users. We found that the most easily scalable and highest performing method was to use user embedding similarity and to interpolate similar user fine-tuned models. Additionally, we provided an analysis of the kind of words that our personalized models are able to more accurately predict. The demographic embedding portion of this work was originally published in [188], while the work on weighted fine-tuning and interpolation was in submission at the time of writing.

CHAPTER 8

Conclusions

In this thesis, we developed methods for targeted sentiment analysis, prediction of conversational behavior, including what someone will say, when they will respond, and aspects of the relationship between interlocutors. We used new methods to construct word embeddings that are personalized to individuals, or to demographic groups, and showed how to use them to improve language modeling, word association, and authorship attribution tasks. Additionally, we explored the benefits of personal data over corpora constructed from larger sets of people and what types of words are more relevant for personalization. Now we will return to the research questions to discuss how each was addressed.

8.1 Revisiting the Research Questions

In the chapters of this thesis we explored several questions related to personalization of word representations and longitudinal dialog. We started by looking at detecting personal preferences through targeted sentiment analysis. We looked at how to detect the sentiment these speakers expressed and how to resolve which entities this sentiment was being expressed toward. We then moved to personal longitudinal corpora to examine behaviors in conversation including what people will say, when they will say it, and how the relationship between interlocutors affects our predictions. In the last two chapters of the thesis, we looked at constructing personalized representations of words, through personalized and demographic word embeddings, as well as personalized language models. We moved from building representations for individual speakers when a lot of data per individual is available, to when we do not have a lot of data per individual. In this case, we consider some known attributes by which we can categorize individuals; demographics, as well as methods for determining the similarity of users when a small amount of writing is available. We now return to the research questions outlined in the introduction to discuss how each has been addressed.

- 1. Can we predict an individuals behaviors, emotions, and relationships from their conversations?**

In Chapters 3-5, we looked at two types of data. We looked at statements made by students about their classes and instructors as well as personal data showing how an individual's conversational data allows us to better predict what people will say, when they will say it, and the relationship between speakers in these conversations. We used student statements to detect entities and the sentiment expressed toward them. The personal conversational data was used to predict the next common utterance in a conversation, the response time behavior, and to predict certain attributes of the interlocutors relationship. To this extent, we have answered this question, though future work may explore conversational behaviors further, in a generative setting and with more detailed response time prediction. The classes of relationships can be expanded in future work, and sentiments can be explored further in more conversational settings.

- 2. Can we better predict what people will say using personalized word representations from an individual or from a composition of demographic representations?**

In Chapter 6, we looked at personalized word embeddings for an individual, and in Chapter 7, we looked at demographic embeddings for four demographic attributes. We showed that demographic embeddings outperformed standard word embeddings. We used language models to predict what individuals will say, and studied how best to initialize such models with our personalized embeddings. We examined which words have the most different representations across individuals and demographics, and which types of words have the highest perplexity reduction when leveraging data from similar users. Future work could provide more insight into this question by looking for ways to more effectively leverage information from similar users and examining how to practically and efficiently apply these methods to transformer language models.

- 3. How does the amount of data from a new user, what we know about them, and ability to measure the similarity of users affect how well we can predict what a new user will say?**

In Chapter 7 of this thesis, we addressed how the data we have about a new user affects our ability to predict their language. If we have demographic attributes, we

can often benefit from demographic embeddings. Otherwise, using small amounts of writing from the user, we can find similar users from which to build language models, and interpolate them to more accurately predict the new users language. We found that our similarity metrics capture information and that our user embedding similarity metric allowed us to perform best. Future work could explore in more detail the time compute trade-off of the options discussed here, for optimizing personalization given a scenario of available information and compute resources.

8.2 Future Work

Personalized models discussed in this thesis have raised additional questions and leave room for several directions of future work. Chapter specific limitations and ethical considerations exist in Sections 6.8 and 7.11.

In Chapter 3, we discussed targeted sentiment analysis in order to develop a method to extract individual user preferences from their utterances. However, this has yet to be tested on natural conversational data. The extent to which the proposed techniques extend to other domains is currently unknown.

Chapters 4 and 5 use personal longitudinal data while Chapters 6 and 7 use public Reddit data. There are important differences in personal language when pragmatic factors vary. For instance, Reddit users develop an identity on the platform where some users may also know the true identity of an individual. The user may also make “throwaway” accounts that provide anonymity and have been shown to exhibit language patterns that differ significantly from the language use of other Reddit users [30, 103]. The differences between personalized language use and modeling between public platforms with varying degrees of anonymity and private discourse remain to be studied.

The experiments in Chapters 4 and 5 could be further expanded upon using techniques from later chapters. For instance, personalized or demographic embeddings, or embeddings based on the speaker relationship annotations could improve performance. Also, the language modeling task in Chapter 4 could be broadened to all exchanged language, which may be more insightful in terms of resulting perplexities and further analysis. Some features we used were derived from lexicons such as LIWC, and while combining these with other features appeared to improve model performance, the analyses using LIWC are limited. Using word counts can give an inaccurate picture, and recent studies have shown, for instance, that emotion word counts are not predictive of affect [98].

Work in Chapter 6 discusses authorship attribution with a result using the perplexities of independently trained language models. Building upon this work, one could construct

a model that jointly considers personalized embeddings in the author classification task in order to improve accuracy. The relative difficulty of this task and the usefulness of the proposed model in the context of state-of-the-art authorship attribution remains to be seen. Some authorship attribution corpora could be used for this task, though they contain significantly less text per author than the Reddit data in our experiments. This also raises the question of how much data is needed for our personalized models to work well, or what is the trade-off between data volume and the performance on language modeling as well as downstream tasks. This is important to consider, as it is often the case that users have little data to leverage for personalization and may lead to further interesting questions about how efficiently model users or create user/personalized embeddings while minimizing the required data volume. Meta-learning is an area of research that focuses on this issue and could have important uses in personalization, as evidenced by early work bridging these fields [114].

In Chapter 7, we build demographic embeddings for users who self-reported demographics and leveraged similar user data when they did not. Similar user data could possibly be used as an alternative source from which to build personalized embeddings. Similarly, the techniques for leveraging similar user data could be applied to users on the basis of their self-reported demographics. Other work has tried variants of interpolated models and priming [93, 104, 171]. Priming language models entails showing it text from the new user and taking that resulting hidden state of the model as it's initialization. It would also be interesting to explore how this method compares to our other methods and how the parameters of priming affect performance (e.g. amount and type of priming data).

A body of recent work has examined how to find informative training examples at the sentence or token level for fine-tuning pretrained language models [5, 63]. The authors find that their methods can speed up training and results in higher performing language models. With such implications, a future direction to pursue could be to take a model like ELMo or BERT and fine-tune it to model an individual's language, and in the process elucidate which training examples are most informative (i.e. provide the largest benefit to personalized models). An analysis of the most useful points (tokens or sentences) could be insightful for learning what language is important to personalize. Alternatively, user representations could be provided to the fine-tuning process in order to improve the selection of informative examples, thus leading to improved fine-tuned, personalized models and consequently, improved personalized embeddings.

Another idea that came up during our work was to build off of a kNN-LM, which finds nearest neighbor words to a given context and weights the standard decoding based on these neighbors [91]. Instead of finding neighbors just based on previous words in the

context, we would also weight or embed user information in this space. We had considered clustering similar data into sets to be used in personalization methods and the kNN-LM is a way of treating each individual data point (i.e. post, sentence, or utterance) as a separate set from which distance can be calculated to what a new user has said. These would be interesting to compare to other ways of grouping, including posts grouped by time-window or subreddit.

Finally, models that take situational context and personal identity into account together should help improve personalized models in many contexts. A problem may arise when a user often talks about one topic (e.g. makeup, beer, diet) but only to a specific audience. In other contexts (e.g. a meeting at work), suggesting such words, although the model has correctly learned their importance for a given user, may be inappropriate. A model that incorporates audience or other pragmatic features may help the model avoid this issue. To some extent the style of a user's language may provide evidence that correlates with these pragmatic features and it would be interesting to devise an experiment that provides insight into the possible gains when explicitly incorporating the pragmatic features in the model.

8.3 Final Remarks

Throughout this thesis we have examined how to use personal, longitudinal, and conversational data from Reddit, text messaging platforms, and newly collected data in order to study the behaviors of individuals and to be able to better predict what people think and how they will act. We look at how to improve natural language technologies for individuals, which we believe is integral to the future directions of the NLP community. Much of work up to this point has focused on large models trained on huge amounts of text written by many people. As our field progresses it will be important to continue to develop techniques for personalization that can be used to develop models for individuals rather than one-size-fits-all. There are many directions in which to take this work and we hope that our methods will help to serve as foundation for further discoveries in the areas of personalization and conversational natural language technologies.

BIBLIOGRAPHY

- [1] AGARWAL, A., AND BHATTACHARYYA, P. Sentiment analysis: A new approach for effective use of linguistic knowledge and exploiting similarities in a set of documents to be classified. In *Proceedings of the International Conference on Natural Language Processing (ICON)* (2005).
- [2] AL ZAMAL, F., LIU, W., AND RUTHS, D. Homophily and latent attribute inference: Inferring latent attributes of twitter users from neighbors. In *Sixth International AAAI Conference on Weblogs and Social Media* (2012).
- [3] AMER, N. O., MULHEM, P., AND GÉRY, M. Toward word embedding for personalized information retrieval. In *Neu-IR: The SIGIR 2016 Workshop on Neural Information Retrieval* (2016).
- [4] ANNETT, M., AND KONDRAK, G. A comparison of sentiment analysis techniques: Polarizing movie blogs. In *Conference of the Canadian Society for Computational Studies of Intelligence* (2008), Springer, pp. 25–35.
- [5] ANTONELLO, R., TUREK, J., AND HUTH, A. Selecting informative contexts improves language model finetuning. *arXiv preprint arXiv:2005.00175* (2020).
- [6] ARGAMON, S., KOPPEL, M., FINE, J., AND SHIMONI, A. R. Gender, genre, and writing style in formal written texts. *Text-Interdisciplinary Journal for the Study of Discourse* 23, 3 (2003), 321–346.
- [7] ARGAMON, S., KOPPEL, M., PENNEBAKER, J. W., AND SCHLER, J. Automatically profiling the author of an anonymous text. *Communications of the ACM* 52, 2 (2009), 119–123.
- [8] ARGAMON, S., ŠARIĆ, M., AND STEIN, S. S. Style mining of electronic messages for multiple authorship discrimination: First results. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (2003), ACM, pp. 475–480.
- [9] ARORA, S., MAY, A., ZHANG, J., AND RÉ, C. Contextual embeddings: When are they worth it? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 2650–2663.

- [10] ARTETXE, M., LABAKA, G., AGIRRE, E., AND CHO, K. Unsupervised neural machine translation. In *International Conference on Learning Representations (ICLR)* (2018).
- [11] BAMMAN, D., DYER, C., AND SMITH, N. A. Distributed representations of geographically situated language. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Baltimore, Maryland, June 2014), Association for Computational Linguistics, pp. 828–834.
- [12] BENEVENUTO, F., RODRIGUES, T., CHA, M., AND ALMEIDA, V. Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement* (New York, NY, USA, 2009), IMC '09, ACM, pp. 49–62.
- [13] BERGSMAN, S., DREDZE, M., VAN DURME, B., WILSON, T., AND YAROWSKY, D. Broadly improving user classification via communication-based name and location clustering on twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2013), pp. 1010–1019.
- [14] BLEI, D. M., NG, A. Y., AND JORDAN, M. I. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.
- [15] BLONDEL, V. D., GUILLAUME, J.-L., LAMBIOTTE, R., AND LEFEBVRE, E. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.
- [16] BOJANOWSKI, P., CELEBI, O., MIKOLOV, T., GRAVE, E., AND JOULIN, A. Updating pre-trained word vectors and text classifiers using monolingual alignment. *arXiv preprint arXiv:1910.06241* (2019).
- [17] BOLUKBASI, T., CHANG, K.-W., ZOU, J. Y., SALIGRAMA, V., AND KALAI, A. T. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems* (2016), pp. 4349–4357.
- [18] BOSTROM, K., AND DURRETT, G. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (Online, Nov. 2020), Association for Computational Linguistics, pp. 4617–4624.
- [19] BRENNAN, M., AFROZ, S., AND GREENSTADT, R. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)* 15, 3 (2012), 12.
- [20] CABRAL, L., AND HORTACSU, A. The dynamics of seller reputation: Evidence from ebay. *The Journal of Industrial Economics* 58, 1 (2010), 54–78.

- [21] CHAMLERTWAT, W., BHATTARAKOSOL, P., RUNGKASIRI, T., AND HARUECHAIYASAK, C. Discovering consumer insight from twitter via sentiment analysis. *J. UCS* 18, 8 (2012), 973–992.
- [22] CHEN, T., AND KAN, M.-Y. Creating a live, public short message service corpus: the nus sms corpus. *Language Resources and Evaluation* 47, 2 (2013), 299–335.
- [23] CORREA, T., HINSLEY, A. W., AND DE ZUNIGA, H. G. Who interacts on the web?: The intersection of users’ personality and social media use. *Computers in human behavior* 26, 2 (2010), 247–253.
- [24] COTTERELL, R., MIELKE, S. J., EISNER, J., AND ROARK, B. Are all languages equally hard to language-model? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)* (2018), pp. 536–541.
- [25] DAI, Z., YANG, Z., YANG, Y., CARBONELL, J., LE, Q., AND SALAKHUTDINOV, R. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 2978–2988.
- [26] DANESCU-NICULESCU-MIZIL, C., GAMON, M., AND DUMAIS, S. Mark my words!: Linguistic style accommodation in social media. In *Proceedings of the 20th international conference on World wide web* (2011), ACM, pp. 745–754.
- [27] DANESCU-NICULESCU-MIZIL, C., AND LEE, L. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics* (Stroudsburg, PA, USA, 2011), CMCL ’11, Association for Computational Linguistics, pp. 76–87.
- [28] DANESCU-NICULESCU-MIZIL, C., WEST, R., JURAFSKY, D., LESKOVEC, J., AND POTTS, C. No country for old members: User lifecycle and linguistic change in online communities. In *Proceedings of the 22nd International Conference on World Wide Web* (2013), ACM, pp. 307–318.
- [29] DAS, A. S., DATAR, M., GARG, A., AND RAJARAM, S. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web* (2007), ACM, pp. 271–280.
- [30] DE CHOUDHURY, M., AND DE, S. Mental health discourse on reddit: Self-disclosure, social support, and anonymity. In *Proceedings of the International AAAI Conference on Web and Social Media* (2014), vol. 8.
- [31] DEERWESTER, S., DUMAIS, S. T., FURNAS, G. W., LANDAUER, T. K., AND HARSHMAN, R. Indexing by latent semantic analysis. *Journal of the American society for information science* 41, 6 (1990), 391–407.

- [32] DEMETER, D., KIMMEL, G., AND DOWNEY, D. Stolen probability: A structural weakness of neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (July 2020), pp. 2191–2197.
- [33] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. 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)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 4171–4186.
- [34] DING, S. H., FUNG, B. C., IQBAL, F., AND CHEUNG, W. K. Learning stylometric representations for authorship analysis. *IEEE transactions on cybernetics* 49, 1 (2017), 107–121.
- [35] DODGE, J., GURURANGAN, S., CARD, D., SCHWARTZ, R., AND SMITH, N. A. Show your work: Improved reporting of experimental results. 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)* (Nov. 2019), pp. 2185–2194.
- [36] DODGE, J., ILHARCO, G., SCHWARTZ, R., FARHADI, A., HAJISHIRZI, H., AND SMITH, N. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305* (2020).
- [37] DU, W., LIN, Z., SHEN, Y., O’DONNELL, T. J., BENGIO, Y., AND ZHANG, Y. Exploiting syntactic structure for better language modeling: A syntactic distance approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (July 2020), pp. 6611–6628.
- [38] DUGGAN, M., AND BRENNER, J. *The Demographics of Social Media Users, 2012*, vol. 14. Pew Research Center’s Internet & American Life Project Washington, DC, 2013.
- [39] EBRAHIMI, J., AND DOU, D. Personalized semantic word vectors. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 1925–1928.
- [40] EIRINAKI, M., AND VAZIRGIANNIS, M. Web mining for web personalization. *ACM Transactions on Internet Technology (TOIT)* 3, 1 (2003), 1–27.
- [41] ELLIS, J., GETMAN, J., AND STRASSEL, S. Overview of linguistic resource for the tac kbp 2014 evaluations: Planning, execution, and results. In *Proc. Text Analysis Conference (TAC2014)* (2014).
- [42] FINKEL, J. R., GRENAGER, T., AND MANNING, C. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (2005), Association for Computational Linguistics, pp. 363–370.

- [43] FLEK, L. Returning the N to NLP: Towards contextually personalized classification models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 7828–7838.
- [44] FORSYTH, E., LIN, J., AND MARTELL, C. Lexical and discourse analysis of online chat dialog. In *Semantic Computing, 2007. ICSC 2007. International Conference on* (2007), IEEE, pp. 19–26.
- [45] GAL, Y., AND GHARAMANI, Z. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems* (2016), pp. 1019–1027.
- [46] GAMON, M. Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In *Proceedings of the 20th International Conference on Computational Linguistics* (2004), Association for Computational Linguistics, p. 611.
- [47] GAO, J., HE, D., TAN, X., QIN, T., WANG, L., AND LIU, T.-Y. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations (ICLR)* (2019).
- [48] GAO, X., LEE, S., ZHANG, Y., BROCKETT, C., GALLEY, M., GAO, J., AND DOLAN, B. Jointly optimizing diversity and relevance in neural response 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)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 1229–1238.
- [49] GARERA, N., AND YAROWSKY, D. Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2* (2009), Association for Computational Linguistics, pp. 710–718.
- [50] GARIMELLA, A., BANEJA, C., AND MIHALCEA, R. Demographic-aware word associations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (Copenhagen, Denmark, 2017), Association for Computational Linguistics, pp. 2285–2295.
- [51] GE, Z., SUN, Y., AND SMITH, M. J. T. Authorship Attribution Using a Neural Network Language Model. In *Thirtieth AAAI Conference on Artificial Intelligence* (2016).
- [52] GJURKOVIĆ, M., AND ŠNAJDER, J. Reddit: A gold mine for personality prediction. In *Proceedings of the Second Workshop on Computational Modeling of People’s Opinions, Personality, and Emotions in Social Media* (New Orleans, Louisiana, USA, June 2018), Association for Computational Linguistics, pp. 87–97.

- [53] GLAVAŠ, G., AND VULIĆ, I. Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Melbourne, Australia, July 2018), Association for Computational Linguistics, pp. 34–45.
- [54] GODBOLE, N., SRINIVASAIAH, M., AND SKIENA, S. Large-scale sentiment analysis for news and blogs. *ICWSM* 7, 21 (2007), 219–222.
- [55] GODFREY, J. J., HOLLIMAN, E. C., AND MCDANIEL, J. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on* (1992), vol. 1, IEEE, pp. 517–520.
- [56] GONG, C., HE, D., TAN, X., QIN, T., WANG, L., AND LIU, T.-Y. Frage: Frequency-agnostic word representation. In *Advances in neural information processing systems* (2018), pp. 1334–1345.
- [57] GONZALES, A. L., HANCOCK, J. T., AND PENNEBAKER, J. W. Language style matching as a predictor of social dynamics in small groups. *Communication Research* 37, 1 (2010), 3–19.
- [58] GRAFF, D. North american news text corpus LDC95T21. Web Download. Philadelphia: Linguistic Data Consortium, 1995.
- [59] GRAVE, E., JOULIN, A., AND USUNIER, N. Improving neural language models with a continuous cache. In *International Conference on Learning Representations* (2016).
- [60] GROSZ, B., SCOTT, D., KAMP, H., COHEN, P., AND GIACHIN, E. Discourse and dialogue. *Survey of the State of the Art in Human Language Technology, Center for Spoken Language Understanding* (1995), 227–254.
- [61] GROVER, A., AND LESKOVEC, J. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining* (2016), ACM, pp. 855–864.
- [62] GU, Y., ZHANG, Z., WANG, X., LIU, Z., AND SUN, M. Train no evil: Selective masking for task-guided pre-training. *ArXiv abs/2004.09733* (2020).
- [63] GU, Y., ZHANG, Z., WANG, X., LIU, Z., AND SUN, M. Train no evil: Selective masking for task-guided pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 6966–6974.
- [64] GURURANGAN, S., MARASOVIĆ, A., SWAYAMDIPTA, S., LO, K., BELTAGY, I., DOWNEY, D., AND SMITH, N. A. Don’t stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (July 2020), pp. 8342–8360.

- [65] HAMDAN, H., BELLOT, P., AND BECHET, F. Lsislif: Crf and logistic regression for opinion target extraction and sentiment polarity analysis. *SemEval-2015* (2015), 753–758.
- [66] HEAFIELD, K., POUZYREVSKY, I., CLARK, J. H., AND KOEHN, P. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (2013), pp. 690–696.
- [67] HIRST, G., AND FEIGUINA, O. Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing* 22, 4 (2007), 405–417.
- [68] HOCHREITER, S., AND SCHMIDHUBER, J. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
- [69] HOFMANN, V., PIERREHUMBERT, J. B., AND SCHÜTZE, H. Dynamic contextualized word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020).
- [70] HOLM, S. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* (1979), 65–70.
- [71] HOLMER, T. Discourse structure analysis of chat communication. *Language@Internet* 5, 10 (2008).
- [72] HOVY, D. Demographic factors improve classification performance. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Beijing, China, July 2015), Association for Computational Linguistics, pp. 752–762.
- [73] HOVY, D., AND SØGAARD, A. Tagging performance correlates with author age. 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)* (Beijing, China, July 2015), Association for Computational Linguistics, pp. 483–488.
- [74] HOVY, D., AND SPRUIT, S. L. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Berlin, Germany, Aug. 2016), Association for Computational Linguistics, pp. 591–598.
- [75] HU, M., AND LIU, B. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (2004), ACM, pp. 168–177.
- [76] HU, M., PENG, Y., HUANG, Z., LI, D., AND LV, Y. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the*

57th Annual Meeting of the Association for Computational Linguistics (Florence, Italy, July 2019), Association for Computational Linguistics, pp. 537–546.

- [77] HUANG, Y.-Y., YAN, R., KUO, T.-T., AND LIN, S.-D. Enriching cold start personalized language model using social network information. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 21, Number 1, June 2016* (June 2016).
- [78] HUTTO, C. J., YARDI, S., AND GILBERT, E. A longitudinal study of follow predictors on twitter. In *Proceedings of the SIGCHI conference on human factors in computing systems* (2013), ACM, pp. 821–830.
- [79] INAN, H., KHOSRAVI, K., AND SOCHER, R. Tying word vectors and word classifiers: A loss framework for language modeling. In *International Conference on Learning Representations (ICLR)* (2017).
- [80] IVANOVIC, E. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop* (Stroudsburg, PA, USA, 2005), ACLstudent '05, Association for Computational Linguistics, pp. 79–84.
- [81] JAECH, A., AND OSTENDORF, M. Personalized language model for query auto-completion. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Melbourne, Australia, July 2018), Association for Computational Linguistics, pp. 700–705.
- [82] JAFARIKINABAD, F., AND HUA, K. A. Style-aware neural model with application in authorship attribution. *arXiv preprint arXiv:1909.06194* (2019).
- [83] JAVA, A., SONG, X., FININ, T., AND TSENG, B. Why we twitter: An analysis of a microblogging community. In *Advances in web mining and web usage analysis* (2009), Springer, pp. 118–138.
- [84] JING, H., KAMBHATLA, N., AND ROUKOS, S. Extracting social networks and biographical facts from conversational speech transcripts. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* (2007), pp. 1040–1047.
- [85] JOSHI, C. K., MI, F., AND FALTINGS, B. Personalization in goal-oriented dialog. *Advances in Neural Information Processing Systems 2017 Conversational AI Workshop* (2017).
- [86] JOULIN, A., BOJANOWSKI, P., MIKOLOV, T., JÉGOU, H., AND GRAVE, E. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 2979–2984.
- [87] JUOLA, P. Authorship attribution. *Foundations and Trends in Information Retrieval* 1, 3 (2008), 233–334.

- [88] KANNAN, A., KURACH, K., RAVI, S., KAUFMAN, T., MIKLOS, B., CORRADO, G., TOMKINS, A., LUKACS, L., GANEA, M., YOUNG, P., AND RAMAVAJJALA, V. Smart reply: Automated response suggestion for email. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (2016)*. (2016).
- [89] KAWAKAMI, K., DYER, C., AND BLUNSOM, P. Learning to create and reuse words in open-vocabulary neural language modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2017)*, pp. 1492–1502.
- [90] KESTEMONT, M., STAMATATOS, E., MANJAVACAS, E., DAELEMANS, W., POTTHAST, M., AND STEIN, B. Overview of the Cross-domain Authorship Attribution Task at PAN 2019. In *CLEF 2019 Labs and Workshops, Notebook Papers (2019)*.
- [91] KHANDELWAL, U., LEVY, O., JURAFSKY, D., ZETTLEMOYER, L., AND LEWIS, M. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations (ICLR) (2019)*.
- [92] KIELA, D., HILL, F., AND CLARK, S. Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (Lisbon, Portugal, Sept. 2015)*, Association for Computational Linguistics, pp. 2044–2048.
- [93] KING, M., AND COOK, P. Evaluating Approaches to Personalizing Language Models. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020) (Marseille, France, 2020)*, European Language Resources Association (ELRA).
- [94] KOCMI, T., AND BOJAR, O. An exploration of word embedding initialization in deep-learning tasks. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017) (2017)*, pp. 56–64.
- [95] KOLCHINSKI, Y. A., AND POTTS, C. Representing social media users for sarcasm detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (Brussels, Belgium, Oct.-Nov. 2018)*, Association for Computational Linguistics, pp. 1115–1121.
- [96] KOPPEL, M., SCHLER, J., AND ARGAMON, S. Computational methods in authorship attribution. *Journal of the Association for Information Science and Technology* 60, 1 (2009), 9–26.
- [97] KOULOUMPIS, E., WILSON, T., AND MOORE, J. D. Twitter sentiment analysis: The good the bad and the omg! *ICWSM 11 (2011)*, 538–541.
- [98] KROSS, E., VERDUYN, P., BOYER, M., DRAKE, B., GAINSBURG, I., VICKERS, B., YBARRA, O., AND JONIDES, J. Does counting emotion words on online social networks provide a window into people’s subjective experience of emotion? a case study on facebook. *Emotion* 19, 1 (2019), 97.

- [99] KUMMERFELD, J. K., GOURAVAJHALA, S. R., PEPPER, J. J., ATHREYA, V., GUNASEKARA, C., GANHOTRA, J., PATEL, S. S., POLYMENAKOS, L., AND LASECKI, W. S. A large-scale corpus for conversation disentanglement. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019), pp. 3846–3856.
- [100] KURITA, K., VYAS, N., PAREEK, A., BLACK, A. W., AND TSVETKOV, Y. Quantifying social biases in contextual word representations. In *Proceedings of GeBNLP 2019: 1st ACL Workshop on Gender Bias for Natural Language Processing* (2019).
- [101] LAKKARAJU, H., SOCHER, R., AND MANNING, C. Aspect specific sentiment analysis using hierarchical deep learning. *Advances in Neural Information Processing Systems* (2014).
- [102] LARSON, B. Gender as a variable in natural-language processing: Ethical considerations. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing* (Valencia, Spain, Apr. 2017), Association for Computational Linguistics, pp. 1–11.
- [103] LEAVITT, A. ” this is a throwaway account” temporary technical identities and perceptions of anonymity in a massive online community. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing* (2015), pp. 317–327.
- [104] LEE, N., BANG, Y., MADOTTO, A., AND FUNG, P. Misinformation has high perplexity. *arXiv preprint arXiv:2006.04666* (2020).
- [105] LESKOVEC, J., AND HORVITZ, E. Planetary-scale views on a large instant-messaging network. In *Proceedings of the 17th International Conference on World Wide Web* (New York, NY, USA, 2008), WWW ’08, ACM, pp. 915–924.
- [106] LI, J., GALLEY, M., BROCKETT, C., SPITHOURAKIS, G., GAO, J., AND DOLAN, B. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Berlin, Germany, Aug. 2016), Association for Computational Linguistics, pp. 994–1003.
- [107] LI, X., BING, L., LI, P., AND LAM, W. A unified model for opinion target extraction and target sentiment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 6714–6721.
- [108] LI, Y., HU, C., ZHANG, Y., XU, N., JIANG, Y., XIAO, T., ZHU, J., LIU, T., AND LI, C. Learning architectures from an extended search space for language modeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 6629–6639.

- [109] LI, Y., ZHANG, Y., DOYU LI, X. T., WANG, J., ZUO, N., WANG, Y., XU, W., CHEN, G., AND GUO, J. Pris at knowledge base population 2013. In *Proc. TAC 2013 Workshop* (2013).
- [110] LIN, Z.-W., SUNG, T.-W., LEE, H.-Y., AND LEE, L.-S. Personalized word representations carrying personalized semantics learned from social network posts. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2017), IEEE, pp. 533–540.
- [111] LOWE, R., POW, N., SERBAN, I., AND PINEAU, J. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909* (2015).
- [112] LUO, L., HUANG, W., ZENG, Q., NIE, Z., AND SUN, X. Learning personalized end-to-end goal-oriented dialog. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 6794–6801.
- [113] LYNN, V., SON, Y., KULKARNI, V., BALASUBRAMANIAN, N., AND SCHWARTZ, H. A. Human centered NLP with user-factor adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (Copenhagen, Denmark, Sept. 2017), Association for Computational Linguistics, pp. 1146–1155.
- [114] MADOTTO, A., LIN, Z., WU, C.-S., AND FUNG, P. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019), pp. 5454–5459.
- [115] MAJUMDER, N., PORIA, S., GELBUKH, A., AND CAMBRIA, E. Deep learning-based document modeling for personality detection from text. *IEEE Intelligent Systems* 32, 2 (2017), 74–79.
- [116] MAJUMDER, N., PORIA, S., HAZARIKA, D., MIHALCEA, R., GELBUKH, A., AND CAMBRIA, E. Dialogue RNN: An attentive rnn for emotion detection in conversations. *Proceedings of the 33rd Conference on Artificial Intelligence (AAAI)* (2019).
- [117] MANNING, C. D., SURDEANU, M., BAUER, J., FINKEL, J., BETHARD, S. J., AND MCCLOSKEY, D. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations* (2014), pp. 55–60.
- [118] MANZINI, T., YAO CHONG, L., BLACK, A. W., AND TSVETKOV, Y. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. 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)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 615–621.

- [119] MARCUS, M. P., SANTORINI, B., AND MARCINKIEWICZ, M. A. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19, 2 (1993), 313–330.
- [120] MAY, C., WANG, A., BORDIA, S., BOWMAN, S. R., AND RUDINGER, R. On measuring social biases in sentence encoders. 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)* (Minneapolis, Minnesota, June 2019), Association for Computational Linguistics, pp. 622–628.
- [121] MAZARÉ, P.-E., HUMEAU, S., RAISON, M., AND BORDES, A. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 2775–2779.
- [122] MCCRAE, R. R., AND COSTA JR, P. T. Personality trait structure as a human universal. *American psychologist* 52, 5 (1997), 509.
- [123] MELIS, G., DYER, C., AND BLUNSOM, P. On the state of the art of evaluation in neural language models. In *International Conference on Learning Representations (ICLR)* (2018).
- [124] MELIS, G., KOČISKÝ, T., AND BLUNSOM, P. Mogrifier LSTM. In *International Conference on Learning Representations* (2020).
- [125] MERITY, S. Single headed attention rnn: Stop thinking with your head. *arXiv preprint arXiv:1911.11423* (2019).
- [126] MERITY, S. Single headed attention rnn: Stop thinking with your head. *ArXiv abs/1911.11423* (2019).
- [127] MERITY, S., KESKAR, N. S., AND SOCHER, R. An analysis of neural language modeling at multiple scales. *arXiv preprint arXiv:1803.08240* (2018).
- [128] MERITY, S., KESKAR, N. S., AND SOCHER, R. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations (ICLR)* (2018).
- [129] MERITY, S., XIONG, C., BRADBURY, J., AND SOCHER, R. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)* (2017).
- [130] MIKOLOV, T., CHEN, K., CORRADO, G., AND DEAN, J. Efficient estimation of word representations in vector space. *International Conference on Learning Representations (ICLR)* (2013).
- [131] MIKOLOV, T., SUTSKEVER, I., CHEN, K., CORRADO, G. S., AND DEAN, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (2013), pp. 3111–3119.

- [132] MIKOLOV, T., AND ZWEIG, G. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT) (2012)*, IEEE, pp. 234–239.
- [133] MITCHELL, M., AGUILAR, J., WILSON, T., AND VAN DURME, B. Open domain targeted sentiment. In *Conference on Empirical Methods in Natural Language Processing (2013)*, pp. 1643–1654.
- [134] MOSTELLER, F., AND WALLACE, D. L. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed federalist papers. *Journal of the American Statistical Association* 58, 302 (1963), 275–309.
- [135] NEISHI, M., SAKUMA, J., TOHDA, S., ISHIWATARI, S., YOSHINAGA, N., AND TOYODA, M. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017) (2017)*, pp. 99–109.
- [136] NEUMANN, M., KING, D., BELTAGY, I., AND AMMAR, W. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task (Aug. 2019)*, pp. 319–327.
- [137] NIGAM, K., AND HURST, M. Towards a robust metric of opinion. In *AAAI spring symposium on exploring attitude and affect in text (2004)*, pp. 598–603.
- [138] NOVIKOVA, J., DUŠEK, O., CERCAS CURRY, A., AND RIESER, V. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (Copenhagen, Denmark, Sept. 2017)*, Association for Computational Linguistics, pp. 2241–2252.
- [139] PAK, A., AND PAROUBEK, P. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc (2010)*, vol. 10, pp. 1320–1326.
- [140] PANG, B., AND LEE, L. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval* 2, 1-2 (2008), 1–135.
- [141] PAPPAS, N., MICULICICH, L., AND HENDERSON, J. Beyond weight tying: Learning joint input-output embeddings for neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers (2018)*, pp. 73–83.
- [142] PAPPAS, N., AND MULCAIRE, PHOEBE SMITH, N. A. Grounded compositional outputs for adaptive language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (2020)*.
- [143] PAVALANATHAN, U., AND EISENSTEIN, J. Confounds and consequences in geo-tagged twitter data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (Lisbon, Portugal, Sept. 2015)*, Association for Computational Linguistics, pp. 2138–2148.

- [144] PENNEBAKER, J. W., FRANCIS, M. E., AND BOOTH, R. J. Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates* 71, 2001 (2001).
- [145] PENNEBAKER, J. W., AND STONE, L. D. Words of wisdom: Language use over the life span. *Journal of personality and social psychology* 85, 2 (2003), 291.
- [146] PENNINGTON, J., SOCHER, R., AND MANNING, C. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Doha, Qatar, Oct. 2014), Association for Computational Linguistics, pp. 1532–1543.
- [147] PEREZ, C. C. *Invisible women: Exposing data bias in a world designed for men*. Random House, 2019.
- [148] PETERS, M., NEUMANN, M., IYYER, M., GARDNER, M., CLARK, C., LEE, K., AND ZETTMEOYER, L. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (New Orleans, Louisiana, June 2018), Association for Computational Linguistics, pp. 2227–2237.
- [149] PETERS, M. E., RUDER, S., AND SMITH, N. A. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)* (2019), pp. 7–14.
- [150] POLYAK, B. T., AND JUDITSKY, A. B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization* 30, 4 (1992), 838–855.
- [151] PONTIKI, M., GALANIS, D., PAPAGEOGIOU, H., MANANDHAR, S., AND ANDROUTSOPOULOS, I. Semeval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, Denver, Colorado (2015).
- [152] PONTIKI, M., PAPAGEORGIOU, H., GALANIS, D., ANDROUTSOPOULOS, I., PAVLOPOULOS, J., AND MANANDHAR, S. Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)* (2014), pp. 27–35.
- [153] POTTHAST, M., ROSSO, P., STAMATATOS, E., AND STEIN, B. A decade of shared tasks in digital text forensics at pan. In *European Conference on Information Retrieval* (2019), Springer, pp. 291–300.
- [154] POTTHAST, M., SCHREMMER, F., HAGEN, M., AND STEIN, B. Overview of the author obfuscation task at PAN 2018: A new approach to measuring safety. In *Proceedings of the Conference and Labs of the Evaluation Forum (CLEF)* (2018).
- [155] PRESS, O., AND WOLF, L. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (2017), pp. 157–163.

- [156] PULMAN, S., AND MIHALCEA, R. Linguistic ethnography: Identifying dominant word classes in text. 595–602.
- [157] QI, P., ZHANG, Y., ZHANG, Y., BOLTON, J., AND MANNING, C. D. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (July 2020), pp. 101–108.
- [158] QI, Y., SACHAN, D., FELIX, M., PADMANABHAN, S., AND NEUBIG, G. When and why are pre-trained word embeddings useful 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 2 (Short Papers)* (2018), pp. 529–535.
- [159] RADFORD, A., WU, J., CHILD, R., LUAN, D., AMODEI, D., AND SUTSKEVER, I. Language models are unsupervised multitask learners. *OpenAI Blog* 1, 8 (2019).
- [160] RANGEL, F., ROSSO, P., KOPPEL, M., STAMATATOS, E., AND INCHEs, G. Overview of the author profiling task at pan 2013. In *CLEF Conference on Multilingual and Multimodal Information Access Evaluation* (2013), CELCT, pp. 352–365.
- [161] RAO, D., YAROWSKY, D., SHREEVATS, A., AND GUPTA, M. Classifying latent user attributes in twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents* (2010), ACM, pp. 37–44.
- [162] RICHARDS, N. M. The dangers of surveillance. *Harv. L. Rev.* 126 (2013).
- [163] RILOFF, E., AND WIEBE, J. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing* (2003), Association for Computational Linguistics, pp. 105–112.
- [164] SAUPER, C., AND BARZILAY, R. Automatic aggregation by joint modeling of aspects and values. *Journal of Artificial Intelligence Research* (2013).
- [165] SCHICK, T., AND SCHÜTZE, H. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (July 2020), pp. 3996–4007.
- [166] SCHUSTER, M., AND PALIWAL, K. K. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45, 11 (1997), 2673–2681.
- [167] SCHWARTZ, H. A., EICHSTAEDT, J. C., KERN, M. L., DZIURZYNSKI, L., RAMONES, S. M., AGRAWAL, M., SHAH, A., KOSINSKI, M., STILLWELL, D., SELIGMAN, M. E., ET AL. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one* 8, 9 (2013), e73791.

- [168] SENNRICH, R., HADDOW, B., AND BIRCH, A. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Berlin, Germany, Aug. 2016), Association for Computational Linguistics, pp. 1715–1725.
- [169] SEROUSSI, Y., ZUKERMAN, I., AND BOHNERT, F. Authorship attribution with topic models. *Computational Linguistics* 40, 2 (2014), 269–310.
- [170] SHAO, L., MANTRAVADI, S., MANZINI, T., BUENDIA, A., KNOERTZER, M., SRINIVASAN, S., AND QUIRK, C. Examination and extension of strategies for improving personalized language modeling via interpolation. *arXiv preprint arXiv:2006.05469* (2020).
- [171] SHAO, L., MANTRAVADI, S., MANZINI, T., BUENDIA, A., KNOERTZER, M., SRINIVASAN, S., AND QUIRK, C. Examination and extension of strategies for improving personalized language modeling via interpolation. In *Proceedings of the First Workshop on Natural Language Interfaces* (Online, July 2020), Association for Computational Linguistics, pp. 20–26.
- [172] SHAREGHI, E., GERZ, D., VULIĆ, I., AND KORHONEN, A. Show some love to your n-grams: A bit of progress and stronger n-gram language modeling baselines. 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)* (2019), pp. 4113–4118.
- [173] SHRESTHA, P., SIERRA, S., GONZÁLEZ, F., MONTES, M., ROSSO, P., AND SOLORIO, T. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (Valencia, Spain, Apr. 2017), Association for Computational Linguistics, pp. 669–674.
- [174] SOCHER, R., PERELYGIN, A., WU, J. Y., CHUANG, J., MANNING, C. D., NG, A. Y., AND POTTS, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)* (2013), vol. 1631, p. 1642.
- [175] SORDONI, A., GALLEY, M., AULI, M., BROCKETT, C., JI, Y., MITCHELL, M., NIE, J.-Y., GAO, J., AND DOLAN, B. A neural network approach to context-sensitive generation of conversational responses. In *North American Chapter of the Association for Computational Linguistics (NAACL)* (2015).
- [176] STAMATATOS, E. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology* 60, 3 (2009), 538–556.
- [177] SUN, K., YU, D., CHEN, J., YU, D., CHOI, Y., AND CARDIE, C. DREAM: A challenge dataset and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics* (2019).

- [178] SURDEANU, M. Overview of the TAC 2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *Proceedings of the Sixth Text Analysis Conference (TAC 2013)* (2013).
- [179] SURDEANU, M., AND JI, H. Overview of the english slot filling track at the TAC 2014 knowledge base population evaluation. In *Proc. Text Analysis Conference (TAC2014)* (2014).
- [180] TAUSCZIK, Y. R., AND PENNEBAKER, J. W. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of language and social psychology* 29, 1 (2010), 24–54.
- [181] THET, T. T., NA, J.-C., AND KHOO, C. S. Aspect-based sentiment analysis of movie reviews on discussion boards. *Journal of information science* (2010).
- [182] TOUTANOVA, K., KLEIN, D., MANNING, C. D., AND SINGER, Y. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 conference of the North American chapter of the association for computational linguistics on human language technology (NAACL-HLT)* (2003), Association for computational Linguistics, pp. 173–180.
- [183] TURNEY, P. D. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics* (2002), Association for Computational Linguistics, pp. 417–424.
- [184] TUULOS, V. H., AND TIRRI, H. Combining topic models and social networks for chat data mining. In *Proceedings of the 2004 IEEE/WIC/ACM international Conference on Web intelligence* (2004), IEEE Computer Society, pp. 206–213.
- [185] WANG, L. L., LO, K., CHANDRASEKHAR, Y., REAS, R., YANG, J., EIDE, D., FUNK, K., KINNEY, R. M., LIU, Z., MERRILL, W., MOONEY, P., MURDICK, D. A., RISHI, D., SHEEHAN, J., SHEN, Z., STILSON, B., WADE, A. D., WANG, K., WILHELM, C., XIE, B., RAYMOND, D. M., WELD, D. S., ETZIONI, O., AND KOHLMEIER, S. Cord-19: The covid-19 open research dataset. In *ACL NLP-COVID Workshop* (2020).
- [186] WANG, Z., HALE, S., ADELANI, D. I., GRABOWICZ, P., HARTMAN, T., FLÖCK, F., AND JURGENS, D. Demographic inference and representative population estimates from multilingual social media data. In *The World Wide Web Conference* (2019), pp. 2056–2067.
- [187] WEBSON, A., CHEN, Z., EICKHOFF, C., AND PAVLICK, E. Are “undocumented workers” the same as “illegal aliens”? Disentangling denotation and connotation in vector spaces. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 4090–4105.

- [188] WELCH, C., KUMMERFELD, J. K., PÉREZ-ROSAS, V., AND MIHALCEA, R. Compositional demographic word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (November 2020).
- [189] WELCH, C., KUMMERFELD, J. K., PÉREZ-ROSAS, V., AND MIHALCEA, R. Exploring the value of personalized word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics* (December 2020).
- [190] WELCH, C., AND MIHALCEA, R. Targeted sentiment analysis: Identifying student sentiment toward courses and instructors. In *Proceedings of IVA 2016, the 16th International Conference on Intelligent Virtual Agents* (2016).
- [191] WELCH, C., AND MIHALCEA, R. Targeted sentiment to understand student comments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (Osaka, Japan, Dec. 2016), The COLING 2016 Organizing Committee, pp. 2471–2481.
- [192] WELCH, C., MIHALCEA, R., AND KUMMERFELD, J. K. Improving low compute language modeling with in-domain embedding initialisation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (November 2020).
- [193] WELCH, C., PÉREZ-ROSAS, V., KUMMERFELD, J. K., AND MIHALCEA, R. Learning from personal longitudinal dialog data. *IEEE Intelligent Systems* 34, 4 (2019), 16–23.
- [194] WELCH, C., PÉREZ-ROSAS, V., KUMMERFELD, J. K., AND MIHALCEA, R. Look who’s talking: Inferring speaker attributes from personal longitudinal dialog. In *Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)* (La Rochelle, France, 2019).
- [195] WEN, T.-H., HEIDEL, A., LEE, H.-Y., TSAO, Y., AND LEE, L.-S. Recurrent neural network based language model personalization by social network crowdsourcing. In *Interspeech* (2013), pp. 2703–2707.
- [196] WILSON, T., WIEBE, J., AND HOFFMANN, P. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing* (2005), Association for Computational Linguistics, pp. 347–354.
- [197] WU, X., LIN, W., WANG, Z., AND RASTORGUEVA, E. Author2Vec: A Framework for Generating User Embedding. *arXiv e-prints* (Mar. 2020), arXiv:2003.11627.
- [198] YATES, A., COHAN, A., AND GOHARIAN, N. Depression and self-harm risk assessment in online forums. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (Copenhagen, Denmark, Sept. 2017), Association for Computational Linguistics, pp. 2968–2978.

- [199] YI, J., NASUKAWA, T., BUNESCU, R., AND NIBLACK, W. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on* (2003), IEEE, pp. 427–434.
- [200] YU, H., AND HATZIVASSILOGLOU, V. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 conference on Empirical methods in natural language processing* (2003), Association for Computational Linguistics, pp. 129–136.
- [201] ZELLERS, R., HOLTZMAN, A., RASHKIN, H., BISK, Y., FARHADI, A., ROESNER, F., AND CHOI, Y. Defending against neural fake news. In *Advances in Neural Information Processing Systems* (2019), pp. 9054–9065.
- [202] ZENG, Z., YIN, Y., SONG, Y., AND ZHANG, M. Socialized word embeddings. In *International Joint Conferences on Artificial Intelligence (IJCAI)* (2017), pp. 3915–3921.
- [203] ZHANG, L., AND LIU, B. Aspect and entity extraction for opinion mining. In *Data mining and knowledge discovery for big data*. Springer, 2014, pp. 1–40.
- [204] ZHANG, M., ZHANG, Y., AND VO, D.-T. Neural networks for open domain targeted sentiment. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)* (2015).
- [205] ZHANG, M., ZHANG, Y., AND VO, D.-T. Gated neural networks for targeted sentiment analysis. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA. Association for the Advancement of Artificial Intelligence* (2016).
- [206] ZHANG, R., HU, Z., GUO, H., AND MAO, Y. Syntax encoding with application in authorship attribution. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 2742–2753.
- [207] ZHANG, S., DINAN, E., URBANEK, J., SZLAM, A., KIELA, D., AND WESTON, J. 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)* (Melbourne, Australia, July 2018), Association for Computational Linguistics, pp. 2204–2213.
- [208] ZHANG, Y., GALLEY, M., GAO, J., GAN, Z., LI, X., BROCKETT, C., AND DOLAN, B. Generating informative and diverse conversational responses via adversarial information maximization. In *Advances in Neural Information Processing Systems* (2018), pp. 1810–1820.
- [209] ZHANG, Z., AND LAN, M. Ecnu: Extracting effective features from multiple sequential sentences for target-dependent sentiment analysis in reviews. *SemEval-2015* (2015), 736.

- [210] ZHOU, K., YANG, S.-H., AND ZHA, H. Functional matrix factorizations for cold-start recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval* (2011), pp. 315–324.