Language-Driven Video Understanding

by

Luowei Zhou

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in The University of Michigan
2020

Doctoral Committee:

Professor Jason J. Corso, Chair
Professor Joyce Y. Chai
Assistant Professor David F. Fouhey
Professor Rada Mihalcea
Dr. Marcus Rohrbach, Facebook AI Research

Luowei Zhou

luozhou@umich.edu

ORCID iD: 0000-0003-1197-0101

To my family

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor Dr. Jason J. Corso for all his support and advices throughout my Ph.D. journey. Neither did I enter school as a Ph.D. student, nor was my initial research focus computer vision. I still remember the winter of 2016, it was he and his computer vision class that opened a new gate to my career. I had never seen anything as engaging as computer vision and it made me always eager to learn more. Dr. Corso generously offered me to work with him in the following summer and without notice, four years flew by just like day one. The journey was full of curiosity, joy, sometimes stress, and of course appreciation.

In addition, I would like to thank my thesis committee members Dr. Rada Mihalcea, Dr. Joyce Y. Chai, Dr. David Fouhey, and Dr. Marcus Rohrbach for their support and feedback during my thesis proposal and oral defense. My sincere thanks also go to all my summer intern mentors Dr. Caiming Xiong, Dr. Marcus Rohrbach, Dr. Hamid Palangi, Dr. Jianfeng Gao, and Dr. Lei Zhang. It is you who made my summer fruitful and smoothed my transition from school to industry working environment. I would like thank my collaborators Dr. Yingbo Zhou, Dr. Richard Socher, Nathan Louis, Dr. Yannis Kalantidis, Dr. Xinlei Chen, and Dr. Houdong Hu for their genuineness and open-mindedness. Special thanks to the entire COG Computer Vision group and my friends for the time we worked together and the amazing board game nights we spent together, the support and laughter.

Finally, I would like to thank my best friend Xilin and my family for their support, encouragement, and understanding. Thank you all for making this dissertation possible.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

Video understanding has advanced quite a long way in the past decade, accomplishing tasks including low-level segmentation and tracking that study objects as pixel-level segments or bounding boxes to more high-level activity recognition or classification tasks that classify a video scene to a categorical action label. Despite the progress that has been made, much of this work remains a proxy for an eventual task or application that requires a holistic view of the video, such as objects, actions, attributes, and other semantic components.

In this dissertation, we argue that language could deliver the required holistic representation. It plays a significant role in video understanding by allowing machines to communicate with humans and to understand our requests, as shown in tasks such as text-to-video search engine, voice-guided robot manipulation, to name a few. Our *language-driven video understanding* focuses on two specific problems: video description and visual grounding. What marks our viewpoint different from prior literature is twofold. First, we propose a bottom-up structured learning scheme by decomposing a long video into individual procedure steps and representing each step with a description. Second, we propose to have both explicit (*i.e.*, supervised) and implicit (*i.e.*, weakly-supervised and self-supervised) grounding between words and visual concepts which enables interpretable modeling of the two spaces.

We start by drawing attention to the shortage of large benchmarks on long video-language and propose the largest-of-its-kind YouCook2 dataset and ActivityNet-Entities dataset in Chap. II and III. The rest of the chapters circle around two main problems: video description and visual grounding. For video description, we first address the problem of decomposing a long video into compact and self-contained event segments in Chap. IV. Given

an event segment or short video clip in general, we propose a non-recurrent approach (*i.e.*, Transformer) for video description generation in Chap. V as opposed to prior RNN-based methods and demonstrate superior performance. Moving forward, we notice one potential issue in end-to-end video description generation, *i.e.*, lack of visual grounding ability and model interpretability that would allow humans to directly interact with machine vision models. To address this issue, we transition our focus from end-to-end, video-to-text systems to systems that could explicitly capture the grounding between the two modalities, with a novel grounded video description framework in Chap. VI. So far, all the methods are fully-supervised, *i.e.*, the model training signal comes directly from heavy & expensive human annotations. In the following chapter, we answer the question "Can we perform visual grounding without explicit supervision?" with a weakly-supervised framework where models learn grounding from (weak) description signal. Finally, in Chap. VIII, we conclude the technical work by exploring a self-supervised grounding approach—vision-language pre-training—that implicitly learns visual grounding from web multi-modal data. This mimics how humans obtain their commonsense from the environment through multi-modal interactions.

CHAPTER I

Introduction

1.1 Motivation

We are entering the era of video. Hundreds of hours of videos are uploaded to major video sharing platforms like YouTube every single minute, with billions of views.[1] Short video content has gained popularity among younger generations worldwidely.[2] Apart from the significant social impact the videos are creating, from a research perspective, videos could be regarded as a reflection of human knowledge (*e.g.*, conversation, storytelling, humor) and intelligence (*e.g.*, instructional video), which makes researchers wonder how the enormous video resources could open a new gate to general intelligence systems. We name the discipline of information mining or learning from videos as video understanding.

Video understanding has been long framed as a pure computer vision problem, accomplishing tasks including low-level segmentation [5, 6, 7] and tracking [8, 9, 10, 11] that study objects as pixel-level segments or bounding boxes to more high-level activity recognition or classification tasks [12, 13, 14, 15] that classify a video scene to a categorical action label. Despite the progress that has been made, much of this work remains a proxy for eventual tasks or applications that require a *holistic* view of video content, such as objects, actions, attributes, and other semantic components (see Fig. 1.1).

[1] `https://merchdope.com/youtube-stats/`

[2] `https://www.oberlo.com/blog/tiktok-statistics`

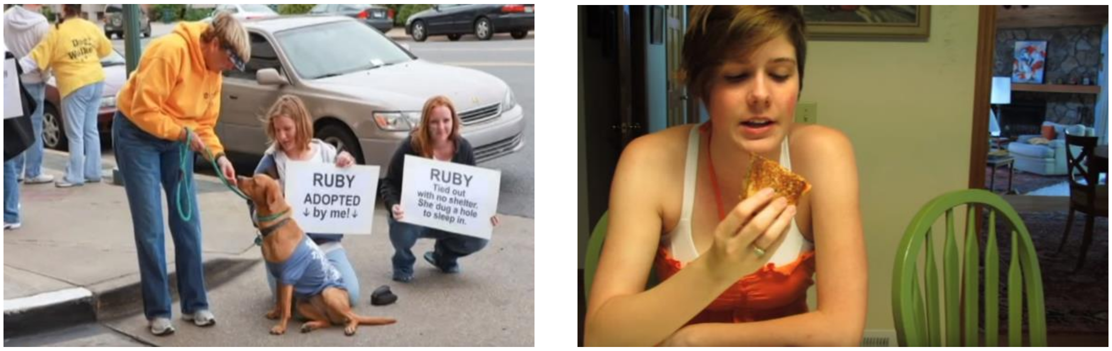


Figure 1.1: We need more than object labels and action labels to understand a scene. For instance, attributes help us disambiguate similar concepts and better ground the visual semantics. (left) It is the person in a **orange** hoodie who is *holding a dog*, not the two girls with the sign nor the people in the background. (right) The girl is *sitting* in the green chair **on the leftmost side**, not the green one on the right side nor the ones in the distance.

In this dissertation, we argue that language could deliver the required *holistic* representation. It plays a significant role in video understanding by allowing machines to communicate with humans and to understand our requests, as shown in tasks such as text-to-video search engine [16][3], voice-guided robot manipulation [17][4], to name a few. Our *language-driven video understanding* focuses on two specific problems: video description and visual grounding. Video description aims to describe the content of a video with a descriptive natural language, allowing the summarization of video content in a human-understandable fashion. Visual grounding aims to link semantics from language back into video, allowing the interpretability in the communication. Our research was made possible as multi-modal video-language data (e.g., video descriptions, subtitles) became prevalent and increasingly accessible recently. The scale of datasets grows substantially (see Fig. 1.2), and the importance of language in video understanding could no longer be overlooked. To what degree could language benefit video scene understanding and how well could language be grounded to visual semantics are among the prominent problems in the community.

Based on the video format, the current focus on video-language includes two major categories: open-domain trimmed short video clips and closed-domain uncurated long videos

[3] http://howto100m.inria.fr/youcook

[4] https://www.youtube.com/watch?v=4L6Q8sAjiCI

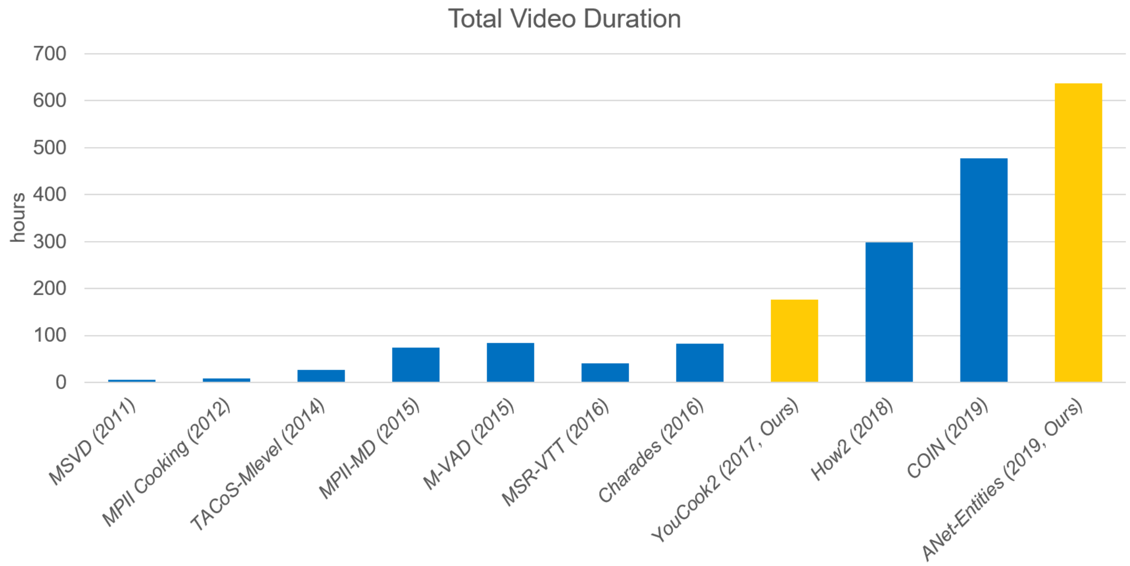


Figure 1.2: Total video duration of video-language datasets over the past decade. Note that our ANet-Entities dataset is based on the train & val splits of ActivityNet Captions [1].

(such as movies, sports, instructional videos). Video clips are usually a few second in length and contain limited and focused content (*e.g.*, a physical action). It is relatively well-studied due to the existence of major benchmarks such as MPII-MD [18], M-VAD [19], and MSR-VTT [20]. Long videos, however, are less-studied partially due to the lack of sufficient data resources. We aim to fill this gap from both the data perspective and the model perspective, and in particular, in the domain of instructional videos (*e.g.*, cooking and assembling). Instructional videos are engaging to the multi-modal learning community, partially because they simulate a teaching-learning environment where the objective is to fulfill a complex procedure through necessary steps. This unique problem setting enables researchers to develop perception-based systems which are capable of learning new tasks, potentially with less or no extra human intervention. Hence, in this work, we focus on this concrete and grounded task: learning from instructional videos.

To address the shortage of instructional video-language data, we propose the first large-scale benchmark named YouCook2 [21]. Since then, the field has accomplished a lot more as a whole [22, 23, 24]. We later propose two more benchmarks namely YouCook2-BoundingBox [25] and ActivityNet-Entities [26] to facilitate research on language-based

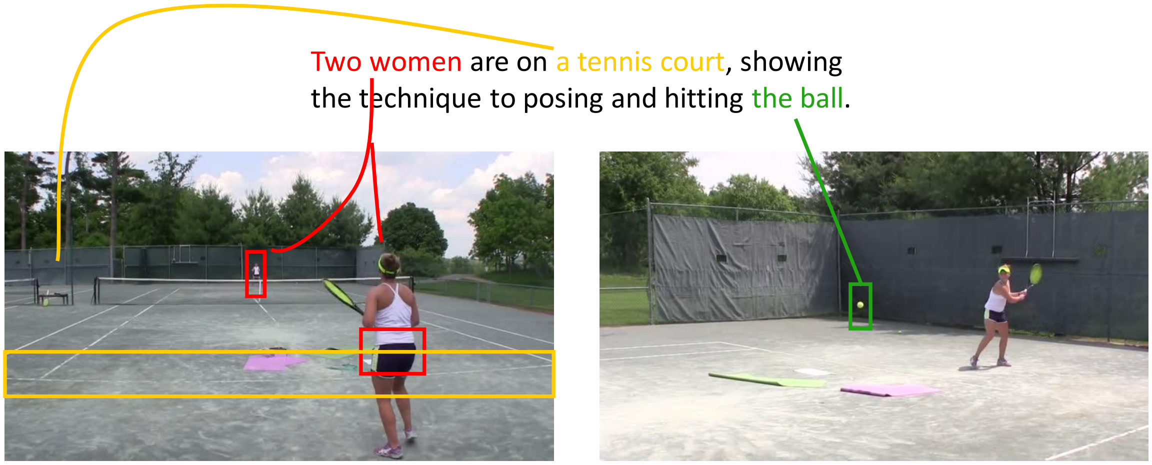

Two women are on a tennis court, showing
the technique to posing and hitting the ball.

Figure 1.3: An example on our grounding annotation. Object words (or noun phrases) are located in the video as spatial bounding boxes.

visual grounding (see Fig. 1.3). The availability of the new grounding annotation allows us to study how visual grounding improves the interpretability of a system as opposed to end-to-end video description models. However, creating dense video annotation is costly and time-consuming, which is arguably the major hurdle to the growth of dataset scale. For example, datasets with automatically-generated annotations could be two magnitudes larger than densely-annotated ones in terms of total video duration [24]. Hence, another aspect we want to highlight in this dissertation is making visual grounding less annotation-hungry. We propose a weakly-supervised grounding framework where models learn to localize objects from (weak) description signal. We also propose a self-supervised approach that implicitly learns grounding from loosely structured image-text pairs from the web, inspired by the recent success of language model pre-training (*e.g.*, BERT [27]). We demonstrate that the joint vision-language representation learned through implicit grounding could generalize well to unseen domains and lead to performance gains on downstream tasks.

1.2 Thesis Statement

In this dissertation, we argue that language plays a significant role in video understanding. Language allows machines to communicate with humans and to understand our requests, delivering a holistic view of video content through compact descriptions. We introduce our approach on video understanding from a unique language-driven perspective, with detailed analyses of video description generation (first half) and language-based visual grounding (second half), mostly in the context of instructional videos. What marks our viewpoint different from prior literature is twofold. First, we propose a bottom-up structured learning scheme by decomposing a long video into individual procedure steps and representing each step with a description. Second, we propose to have both explicit (*i.e.*, supervised) and implicit (*i.e.*, weakly-supervised and self-supervised) grounding between language semantics and video semantics, which enables us an interpretable modeling of the two spaces.

1.3 Contributions

1.3.1 Dataset and Benchmark for Video-Language Understanding

We collect and distribute a large-scale instructional video dataset YouCook2 for procedure learning and recipe generation, which is one of the major testbeds of our methods in this dissertation. YouCook2 contains 2000 videos from 89 recipes with a total length of 176 hours, which is largest-of-its-kind. The procedure steps for each video are annotated with temporal boundaries and described post-hoc by a viewer/annotator with imperative English sentences (see Fig. 1.4). The follow-up work on object grounding leads to YouCook2-BoundingBox, which further maps object words in YouCook2 descriptions to bounding boxes in the video. Later, as a larger-scale effort to bridge video description and visual grounding, we collect the ActivityNet-Entities dataset, which grounds video descriptions to bounding boxes on the level of noun phrases, with 158k boxes on 15k videos. Our dataset allows both, *teaching* models to explicitly rely on the corresponding evidence in the video frame when generating words and *evaluating* how well models are doing in grounding individual words or phrases they generated.

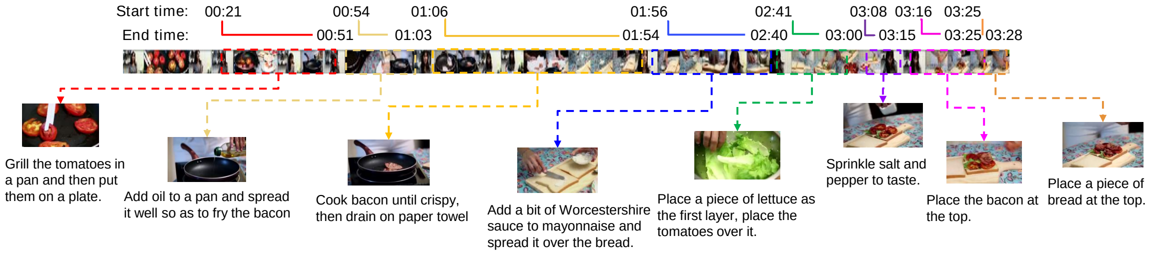

Start time:
00:21
00:54
01:06
01:56
02:41
03:08
03:16
03:25
End time:
00:51
01:03
01:54
02:40
03:00
03:15
03:25
03:28
Grill the tomatoes in a pan and then put them on a plate.
Add oil to a pan and spread it well so as to fry the bacon
Cook bacon until crispy, then drain on paper towel
Add a bit of Worcestershire sauce to mayonnaise and spread it over the bread.
Place a piece of lettuce as the first layer, place the tomatoes over it.
Sprinkle salt and pepper to taste.
Place the bacon at the top.
Place a piece of bread at the top.

Figure 1.4: An example from the YouCook2 dataset on making a BLT sandwich. Each procedure step has time boundaries annotated and is described by an English sentence. Video from YouTube with ID: `4eWzsx1vAi8`.

1.3.2 Instructional Video as Procedure Segments

We introduce and are the first to tackle the class-agnostic procedure segmentation problem in untrimmed videos. We define *procedure* as the sequence of necessary steps comprising a complex task (*e.g.*, *making sandwiches* [28], *changing tires* [29]), and define each individual step as a *procedure segment*, or simply *segment* for convenience. For example, there are 8 segments in the *making a BLT sandwich* video shown in Fig. 1.4. We represent these segments by their start and end temporal boundaries in a given video. Note that one procedure segment could contain multiple actions, but it should be conceptually compact, i.e., described with a single sentence. The number of procedure segments and their locations reflect human consensus on how the procedure is structured. To that end, we define the *Procedure Segmentation* problem as: automatically segment a video containing a procedure into class-agnostic procedure segments. We introduce a model that captures this human consensus from data and generate semantically meaningful segments.

1.3.3 Fine-Grained Video Description Generation

Video description aims to describe the content of a video with a descriptive natural language. It works as a bridge for human-machine interaction in the context of video. Since the capacity of a single sentence is rather limited, in the context of untrimmed videos where multiple events might occur, we first detect the event segments (*i.e.*, start and end timestamps) by using our techniques developed earlier in Sec. 1.3.2 and then describe each segment with a sentence. We call this Dense Video Description [1] and the two stages are named event proposal and description generation accordingly. In our work, we propose an end-to-end model for this task. A differentiable masking scheme is proposed to ensure the consistency between the proposal module and the description module during training. Also, we employ self-attention: a scheme that facilitates the learning of long-range visual dependencies existing in dense video description better than traditional RNN-based methods.

1.3.4 Visually-Grounded Learning from Videos

The major challenge of video description lies in the large variability both on the video and language side. Existing models, hence, typically shortcut the difficulty in recognition and generate plausible sentences that are based on priors but are not necessarily grounded in the video. Therefore, we propose to explicitly link the sentence to the evidence in the video by annotating each noun phrase in a sentence with a corresponding bounding box in one of the frames of a video (our ActivityNet-Entities dataset). To generate grounded captions, we propose a novel video description model which is able to exploit these bounding box annotations. More importantly, with the grounding annotation, we can now evaluate how grounded or faithful the learned model is to the video it describes. We demonstrate the effectiveness of our model on multiple benchmarks across both video and image domains and also showcase that our generated descriptions are more interpretable compared to baseline methods. We further show that even when grounding annotations are unavailable during training, we can still learn object grounding with (weak) language supervision through weighted ranking losses.

1.3.5 Grounding as Commonsense: Vision-Language Pre-training

Now, we take weakly-supervised grounding a step further. So far, available paired video-language data are rather limited as descriptions require manual annotation. What if we can generate unlimited paired data? Can we learn a “commonsense” grounding between the two? The commonsense here indicates a general relationship between words and visual concepts that could generalize to unseen domains and tasks. Contemporaneously with [30, 31, 32, 33, 34, 35, 36, 37], we propose Vision-Language Pre-training where a base model (*e.g.*, BERT [27]) is first pre-trained on a large amount of web image-text pairs using unsupervised learning objectives and then fine-tuned for various downstream tasks with minor architecture changes. The pre-training phase works as a warm-start for downstream tasks such that the model reaches a higher accuracy faster compared to the baseline (without pre-training) and achieves a better final performance. Our proposed method has two main advantages in comparison with the contemporaneous works. First, it unifies the language encoder and decoder and learns a more universal contextualized vision-language representation that can be more easily fine-tuned for vision-language generation and understanding tasks as different as image captioning and visual question answering (VQA). Second, the unified pre-training procedure leads to a single model architecture for two distinct vision-language prediction tasks, *i.e.*, bidirectional and seq2seq, alleviating the need for multiple pre-training models for different types of tasks without any significant performance loss in task-specific metrics.

1.4 Relevant Publications

1. Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2018

2. Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8739–8748, 2018

3. Luowei Zhou, Nathan Louis, and Jason J Corso. Weakly-supervised video object grounding from text by loss weighting and object interaction. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018

4. Luowei Zhou, Yannis Kalantidis, Xinlei Chen, Jason J Corso, and Marcus Rohrbach. Grounded video description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6578–6587, 2019

5. Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2020

CHAPTER II

Related Work

In this chapter, we summarize all the related work throughout the dissertation.

2.1 Learning from Instructional Videos

Procedure Learning from Subtitles. This line of work mainly emphasizes on video-subtitle alignment [39, 40] or discovery of common procedure steps from subtitles [29, 41]. They typically make a strong assumption on the availability of the subtitles, or the number of procedure steps for a certain procedure is fixed, or both. Such assumptions are limited: the text/speech instruction input is unavailable in some scenarios, e.g., streaming video from robot camera; the subtitles or action sequences automatically generated by machines, e.g., YouTube's Automatic Speech Recognition (ASR) system, are not fully reliable and require manual intervention [29]; and many procedures of a certain type, such as a specific recipe, will differ in the number of steps in different instances due to process variation.

Action Segmentation. Action segmentation or labeling [42, 43, 44, 45] approaches the problem in a more grounded setting, *i.e.*, by analysing atom actions (*e.g.*, grind coffee beans, pour coffee) based on the visual input alone. It addresses the problem of segmenting a long video into contiguous segments that correspond to a sequence of actions. Most recently, Huang *et al.* [44] propose to enforce action alignment through frame-wise visual similarities. Kuehne *et al.* [43] apply Hidden Markov Models (HMM) to learn the likelihood of image features given hidden action states. Both methods focus on the transitions between adjacent action states, leaving long-range dependencies not captured. Also, these methods generally assume contiguous action segments with limited or no background activities between segments. Yet, background activities are detrimental to the action localization accuracy [44].

Temporal Action Proposals. Another common approach to study action is through Temporal Action Proposals (TAP) or simply Action Proposals. It aims to temporally localize action-agnostic proposals in a long untrimmed video. Existing methods formulate TAP as a binary classification problem and differ in how the proposals are proposed and discriminated from the background. Shuo *et al.* [46] propose and classify proposal candidates directly over video frames in a sliding window fashion, which is computationally expensive. More recently, inspired by the anchoring mechanism from object detection [47], two types of methods have been proposed—explicit anchoring [48, 21] and implicit anchoring [49, 50]. In the former case, each anchor is an encoding of the visual features between the anchor temporal boundaries and is classfied as action or background. In implicit anchoring, recurrent networks encode the video sequence and, at each anchor center, multiple anchors with various sizes are proposed based on the same visual feature.

Summary. Our approach on instructinal video understanding is largely inspired by Action Proposals. We treat video as a set of well-structured and semantically-meaningful segments and make no assumption on subtitle availability nor recipe type (*e.g.*, number of recipe steps). We will demonstrate later in Chapter IV and V our improvements over existing methods.

2.2 Video and Language: a historical view

The history of bridging video and Language dates back to “pre-deep learning era”, roughly a decade ago. Inspired by natural language generation from images [51, 52], the initial efforts on video-language mainly focus on description generation [53, 54]. A few salient characteristics of this line of work include i) high-level concept detections from video, such as objects, ii) template-filling. The detection outcome is first converted into subject-verb-object triplets through ranking and then filled in pre-defined description template. Hence the generated descriptions are less human-like and can easily be distinguished. Follow-up works [55, 56] refine the detection outcome and the subject-verb-object ranking procedure by incorporating external language knowledge. Contemporaneous with these, Yu *et al.* [57] start to study video-language from a *grounding* perspective — learning word embedding from video clips paired with descriptive sentences. The visual concepts are represented by object tracks or their compounds and are grounded to the linguistic semantics, such as nouns, verbs, and adjectives. Later on, Yang *et al.* [58] consider visual grounding in a more situated setting called Semantic Role Labeling (SRL), where the arguments of verbs in a descriptive sentence (*i.e.*, agent and patient) are grounded to video object tracks and a joint-inference is performed between video and language. These two are among the first works dedicated to explicit language grounding in video.

After the deep learning revolution started in the early 2010s, numerous works have been poured into this field, and hence we will focus on the most relevant works to this dissertation in the following sections, including description generation and visual grounding. We have spread detailed discussions in the following sections and want to note one work particularly by Xu *et al.* [59], who propose a generic framework for learning joint video-language embedding and attack a series of video-language tasks, namely, video description generation, video-to-text retrieval, and text-to-video retrieval, all of which have been intensively studied until now. Another interesting perspective from this work is that it emphasizes the importance of learning linguistic concepts with corresponding visual concepts in mind, again, *visual grounding*. It points out that "commonly used word similarity captures more syntactic expression than visually grounded semantics, e.g., in WordNet, the Lesk similarity [60] between cat and kitten is 0.4 while the similarity between cat and dog is 1.04", which is a sound demonstration.

Summary. The core problem of video-language is essentially learning a joint embedding space between the two. The concept of visual grounding emerged recently, linking the two modalities in a fine-grained level and making the joint embedding space more human-interpretable. Overall, the core problem has not changed much over time while methodology migrates, especially with the recent boom of artificial neural networks.

2.3 Description Generation

Image & Video Description. Early work on automatic caption generation mainly includes template-based approaches [56, 52, 61], where predefined templates with slots are first generated and then filled in with detected visual evidences. Although these works tend to lead to well-grounded methods, they are restricted by their template-based nature. More recently, neural network-based methods have started to dominate major captioning benchmarks. They follow a similar encoder-decoder architecture where the encoder transforms each input frame or clip into a feature vector, aggregates features across video and then the decoder projects the aggregated feature into a sequence of words (see Fig. 2.1). The feature aggregation strategy varies from basic *static* approaches such as mean pooling and LSTM, to visual attention [62], in which the decoder *dynamically* focuses on different location on the "feature bank" as the caption generation proceeds. The prior visual attention-based work usually comes in the form of temporal attention [63] (or spatial-attention [64] in the image domain), semantic attention [65, 66, 67, 68] or both [69]. The recent unprecedented success in object detection [47, 70] has regained the community's interests on detecting fine-grained visual clues while incorporating them into end-to-end networks [71, 72, 73, 74]. This line of work is usually based on region attention [73]. Grounding happens during region attention and could be either implicit [73, 71] or explicit (*i.e.*, with grounding annotation for training) [74, 26]. Our work heavily relies on region attention and grounding and hence we will have a detailed review below.

Grounded Caption Generation. Grounding-based methods [71, 75, 73, 74, 56, 52] tackle

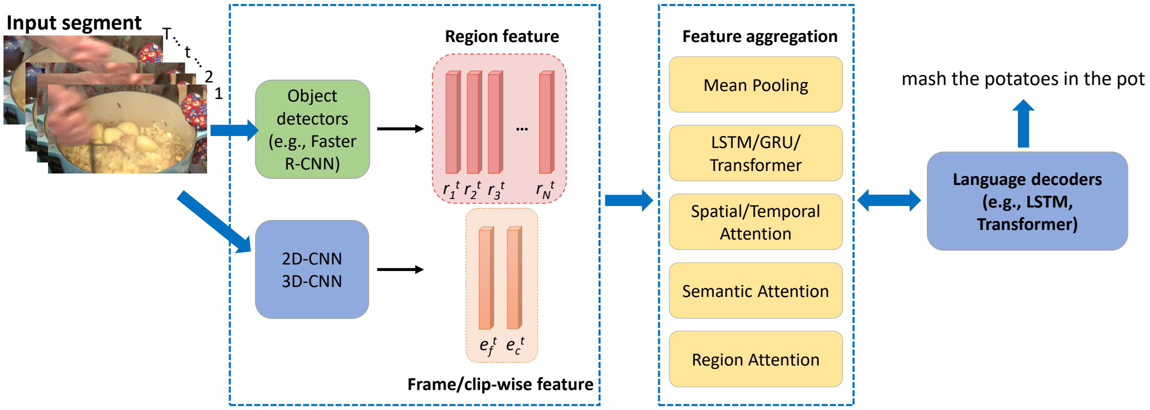

Input segment
T
t
2
1
Object detectors (e.g., Faster R-CNN)
Region feature
r_1^t r_2^t r_3^t ... r_N^t
2D-CNN 3D-CNN
e_f^t e_c^t
Frame/clip-wise feature
Feature aggregation
Mean Pooling
LSTM/GRU/ Transformer
Spatial/Temporal Attention
Semantic Attention
Region Attention
Language decoders (e.g., LSTM, Transformer)
mash the potatoes in the pot

Figure 2.1: Schematic illustration of neural network-based video description methods. Image description methods have a similar structure that takes in a single image as the input and uses feature from either global image feature, 2D-CNN activation map, or region proposals.

the captioning problem in two stages. They first use off-the-shelf or fine-tuned object detectors to propose object proposals/detections as for the visual recognition heavy-lifting. Then, in the second stage, they either attend to the object regions dynamically [71, 75, 73] or classify the regions into labels and fill into pre-defined/generated sentence templates [74, 56, 52]. However, directly generating proposals from off-the-shelf detectors causes the proposals to bias towards classes in the source dataset (*i.e.*, for object detection) *vs.* contents in the target dataset (*i.e.*, for description). One solution is to fine-tune the detector specifically for a dataset [74] but this requires exhaustive object annotations that are difficult to obtain, especially for videos. Instead of fine-tuning a general detector, we transfer the object classification knowledge from off-the-shelf object detectors to our model and then fine-tune this representation as part of our generation model with sparse box annotations. With a focus on co-reference resolution and identifying people, [72] proposes a framework that can refer to particular character instances and do visual co-reference resolution between video clips. However, their method is restricted to identifying human characters whereas we study more general the grounding of objects. Other works include capturing the relationships among object regions by using Graph Convolutional Networks (GCNs) [76] and incorporating language inductive bias [77].

Attention Supervision. As fine-grained grounding becomes a potential incentive for next-generation vision-language systems, to what degree it can benefit remains an open question. On one hand, for VQA [78, 79] the authors point out that the attention model does not attend to same regions as humans and adding attention supervision barely helps the performance. On the other hand, adding supervision to feature map attention [80, 81] was found to be beneficial. We noticed in our preliminary experiments that directly guiding the region attention with supervision [74] does not necessary lead to improvements in automatic sentence metrics. We hypothesize that this might be due to the lack of object context information and we thus introduce a self-attention [82] based context encoding in our attention model, which allows information passing across all regions in the sampled video frames.

Dense Caption Generation. Describing a image or video with a single sentence could be challenging. Johnson *et al.* [83] propose to enlarge descriptor capacity by first generating individual object regions and then describe each region separately. In the video domain, similarly, a problem called video paragraph captioning is proposed by Yu *et al.*[84] where sentences are generated for temporal event segments. However, the temporal locations of each event are provided beforehand. Das *et al.* [56] generate dense captions over the entire video using sparse object stitching, but their work relies on a top-down ontology for the actual description and is not data-driven like the recent captioning methods. The most similar work to ours is Krishna *et al.* [1] who introduce a dense video captioning model that learns to propose the event locations and caption each event with a sentence. However, they combine the proposal and the captioning modules through co-training and are not able to take advantage of language to benefit the event proposal [85]. To this end, we propose an end-to-end framework for doing dense video captioning that is able to produce proposal and description simultaneously. Also, our work directly incorporates the semantics from captions to the proposal module.

2.4 Language & Vision-Language Pre-training

Language Pre-training. Among numerous BERT variants in language pre-training, we review the two methods that are most relevant to our approach, namely Unified LM or UniLM [86] and Multi-Task DNN (MT-DNN) [87]. UniLM employs a shared Transformer network which is pre-trained on three language modeling objectives: unidirectional, bidirectional, and sequence-to-sequence. Each objective specifies different binary values in the self-attention mask to control what context is available to the language model. MT-DNN combines multi-task training and pre-training by attaching task-specific projection heads to the BERT network. Our work is inspired by these works and tailored for vision-language tasks in particular.

Vision-Language Pre-training. This has become a nascent research area in the vision-language community. Related works include ViLBERT [30] and LXMERT [88], both of which tackle understanding-based tasks only (e.g., VQA and Retrieval) and share the same two-stream BERT framework with a vision-language co-attention module to fuse the information from both modalities. ViLBERT is tested on a variety of downstream tasks including VQA, referring expression, and image-to-text retrieval. LXMERT only focuses on a particular problem space (*i.e.*, VQA and visual reasoning) and the generalization ability further compromises when the datasets from the downstream tasks are also exploited in the pre-training stage. The most similar work to ours is VideoBERT [36], which addresses generation-based tasks (*e.g.*, video captioning) and understanding-based tasks (*e.g.*, action classification). However, it separates the visual encoder and the language decoder and performs pre-training only on the encoder, leaving decoder uninitialized. In contrast, we propose a unified model for both encoding and decoding and fully leverage the benefit of pre-training.

2.5 Miscellaneous

Object Grounding with Weak Supervision. Supervised object grounding or referring expression grounding has been intensively studied in the image domain [89, 90, 91] and is gradually drawing attention in the video domain [92, 93]. These methods require dense bounding box annotations for training, which are expensive to obtain. Recently, an increasing amount of attention has shifted towards the weakly-supervised grounding problem [94, 95, 96, 97, 98], where only descriptive phrases, no explicit target grounding locations, are made accessible during training. Karpathy and Fei-Fei [97] propose to pair image regions to words in a sentence by computing a visual-semantic similarity score, finding the word that best describes the region. Rohrbach *et al.* [94] ground textual phrases in images by reconstructing the original phrase through visual attention. Yu and Siskind [99] ground objects from text in constrained videos. De-An *et al.* [98] extend [97] to the video domain and further improve the work by modeling the reference relationships among segments. In our work, we tackle the problem from a novel aspect as fully exploiting the visual-semantic relations within each segment, *i.e.*, frame-wise supervisions and object interactions.

VQA. VQA is another prevalent research area in vision and language. Given an image and a query natural language question about the image, the task is to answer the question by making a multi-choice selection or generating a natural language response. Since its initial proposal [100], there has been a significant amount of works proposing model architectures to fuse question and image representations [101, 73, 102], new datasets or models to reduce the dataset bias [103, 104, 105] and ground the answer in the question [106]. We present in our work a base model that tackles both description generation problem and VQA with minor architecture changes.

CHAPTER III

Dataset and Benchmark

3.1 YouCook2

Our YouCook2 dataset [21] contains 2000 videos that are nearly equal-distributed over 89 recipes. The recipes are from four major cuisine locales, e.g., Africa, Americas, Asia and Europe, and have a large variety of cooking styles, methods, ingredients and cookwares. The videos are collected from YouTube, where various challenges, e.g., fast camera motion, camera zooms, video defocus, and scene-type changes are present. Table 3.1 shows the comparison between YouCook2 and other commonly-used instructional video datasets, e.g., YouCook [56], MPII [107], 50Salads [108], Coffee [29], Breakfast [28] and Charades [109].

Most of the datasets mentioned above have temporally localized action annotations. Compared to action segments, our procedure segments can contain richer semantic information and better capture the human-involved processes in instructional videos. Due to the variety of instructional processes and how each process can be performed, a fixed set of actions fails to describe the details in the video process (e.g., attributes and fine-grained objects). For example, the attribute "crispy" in the recipe step "cook bacon until crispy then drain on paper towel" (see Fig. 1.4) cannot be described by any action nor activity labels.

Name	**Duration**	**UnCons.**	**Proc. Ann.**
YouCook	140 m	Yes	No
MPII	490 m	No	No
50Salads	320 m	No	No
Coffee	120 m	Yes	No
Breakfast	67 h	Yes	No
Charades	82h	Yes	No
YouCook2	**176h**	**Yes**	**Yes**

Table 3.1: Comparisons of instructional video datasets. UnCons. stands for Unconstrained Scene and Proc. Ann. is short for Procedure Annotation.

3.1.1 Annotations

Each video contains 3–16 procedure segments. The segments are temporally localized (timestamps) and described by English sentences in imperative form (e.g., *grill the tomatoes in a pan*). An example is shown in Fig. 1.4. The annotators have access to audio and subtitles but are required to organize and summarize the descriptions in their own way. As indicated in prior work [110], people generally agree with boundaries of salient events in video and hence we collect one annotation per video. To reflect the human consensus on how a procedure should be segmented, we annotate each video with two annotators, one for the major effort and the other one for verification. We also set up a series of restrictions on the annotation to enforce this consensus among different annotators. We have found that consensus is comparatively easy to achieve given the grounded nature of the instructional video domain.

3.1.2 Statistics and Splits

The average number of segments per video is 7.7 and the mean and standard deviation of the number of procedure segments per recipe are shown in Fig. 3.1. The distribution of video duration is shown in Fig. 3.2(a). The total video length is 175.6 hours with an average duration of 5.27 min per video. All the videos remain untrimmed and can be up to 10 min. The distribution of segment durations is shown in Fig. 3.2(b) with mean and

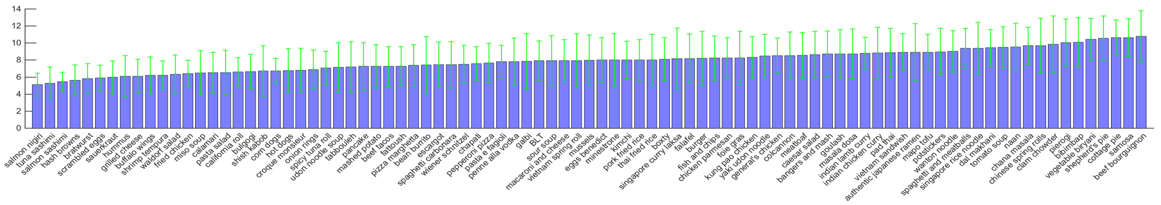
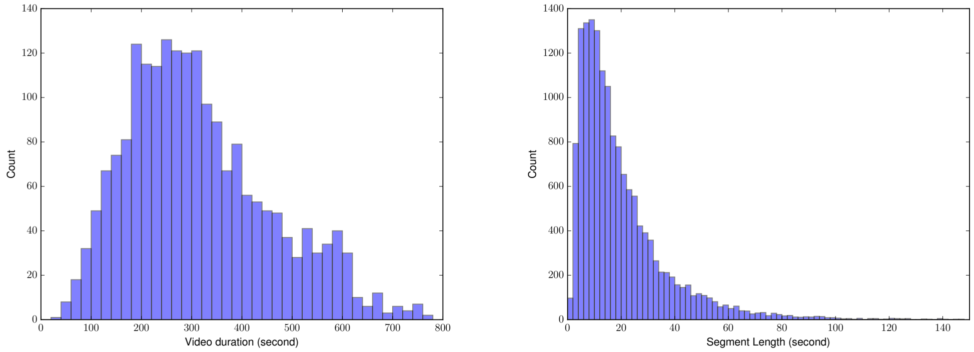


Figure 3.1: Mean and standard deviation of number of procedure segments for each recipe.



(a) Distribution of video duration.

(b) Distribution of segment duration.

Figure 3.2: YouCook2 dataset duration statistics.

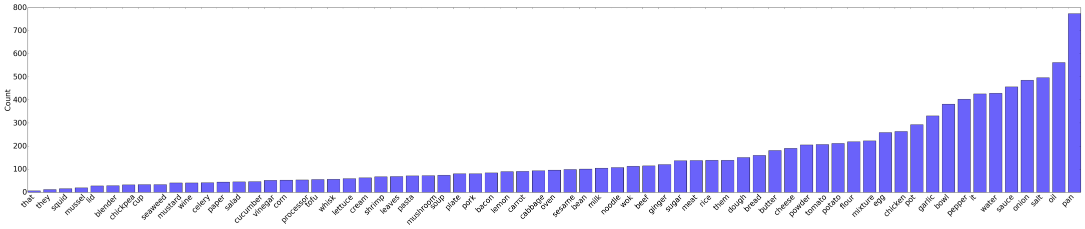
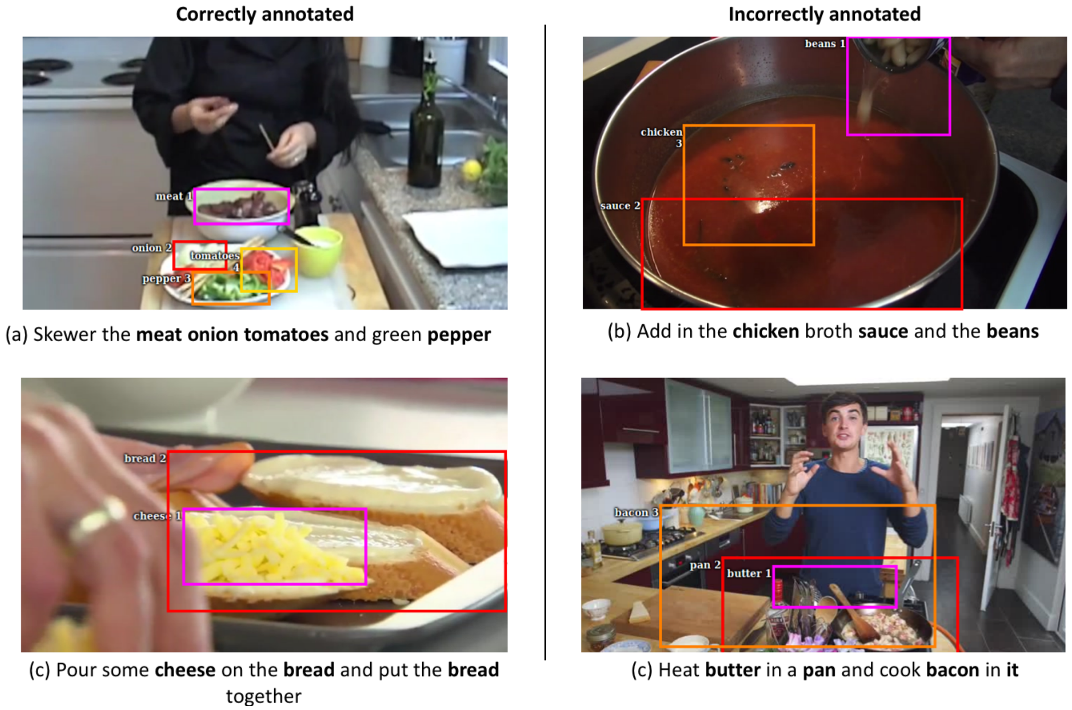


Figure 3.3: Frequency count of each class label (including referring expressions).

standard deviation of 19.6s and 18.2s, respectively. The longest segment lasts 264s and the shortest one lasts 1s. For the recipe descriptions, the total vocabulary is around 2600 words. We randomly split the dataset to 67%:23%:10% for training, validation and testing according to each recipe.

3.1.3 YouCook2-BoundingBox

We further collect YouCook2-BoundingBox (or YouCook2-BB), where we provide bounding box annotations for each segment-description pair in validation and testing sets of YouCook2. The target objects to annotate are the most frequently-appearing objects from



(a) Skewer the **meat onion tomatoes** and green **pepper**

(b) Add in the **chicken** broth **sauce** and the **beans**

(c) Pour some **cheese** on the **bread** and put the **bread** together

(c) Heat **butter** in a **pan** and cook **bacon** in **it**

Figure 3.4: Annotations completed by MTurk workers; The images on the left denote correct annotations and the right shows incorrect annotations. Each image is a frame from the video segment accompanied with its descriptive phrase. Better viewed in color.

YouCook2 recipe descriptions, *i.e.*, the top 63 recurring objects along with four referring expressions: *it, them, that, they* (see Fig. 3.3). Note that the training set is not annotated as we only need bounding boxes for evaluation purposes only.

For annotation, we sample each segment at 1 fps and tailor VATIC [111] for our task. We request Amazon Turk workers to draw bounding box around the objects in the video segment using the highlighted words in the sentence (from the 67 objects in our vocabulary). All annotations are further verified by the top 30 annotators. Some example annotations are in Fig. 3.4.

Dataset Statistics From the validation & testing segments annotated we have a total of 4,325 annotated segments with 2,962 validation and 1,363 testing segments, respectively. These segments were extracted from 647 videos that contain words from our vocabulary list.

Fig. 3.5 displays the number of target objects from the annotated YouCook2-BoundingBox

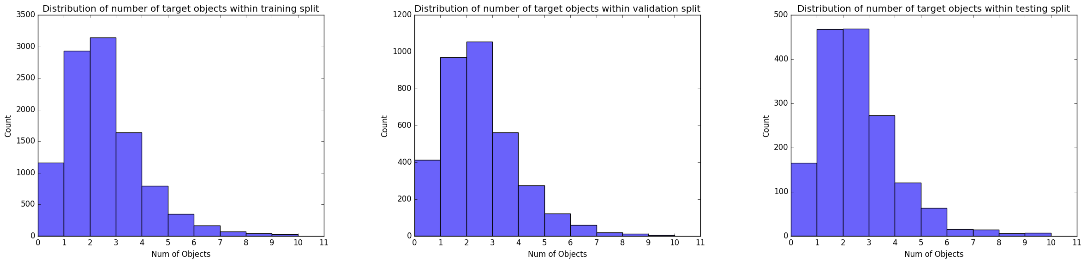


Figure 3.5: Distribution of number of target objects within each segment for train/val/test splits. Target objects belong in our vocabulary of 67 words.

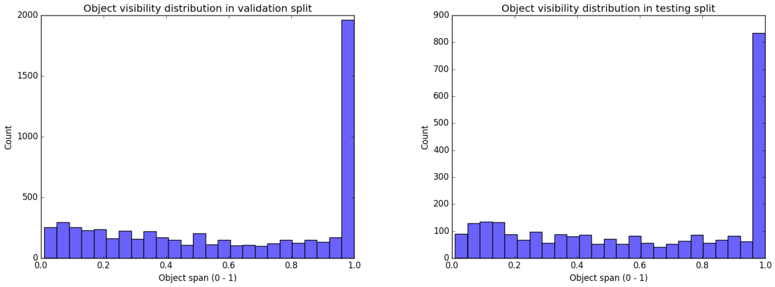


Figure 3.6: Span of object duration in each segment for annotated val/test splits.

segments. The mean target object per sentence is 2.05 with a standard deviation of 1.49. The target objects are words that belong in our vocabulary list of 67 objects.

When completing the annotations, the workers were given the option to mark an object as ”outside of view frame”, ”occluded”, or both. We define an object’s visibility as in view of the current frame with no occlusion. From our collected annotations, Fig. 3.6 shows each object’s visibility duration in the validation & testing split. In the validation split objects are visible 60.72% of the time, and 60.58% for testing. Note from Fig. 3.6 there is a spike in objects with 100% duration, this is attributed to the shorter segments from our collected data. It is perfectly reasonable to have a visible object for the entire duration of shorter segments, some as short as 2 seconds.

3.2 ActivityNet

ActivityNet dataset [4] is a large-scale benchmark with video-level human activity labels. ActivityNet Captions dataset [1] extend the all the 20k ActivityNet videos with dense language annotations, *i.e.*, both temporal event segments and natural language descriptions. We further extend ActivityNet Captions with object entity annotations, in the form of bounding boxes. We named our dataset ActivityNet-Entities.

3.2.1 ActivityNet-Entities Dataset

Our ActivityNet-Entities (ANet-Entities) dataset[1] provides more than 158k entity-level bounding box annotations on 15k videos, making it the largest annotated dataset of its kind to the best of our knowledge. Enriched with semantic information, ANet-Entities is designed for the grounded video description problem where explicit grounding is critical.

When it comes to videos, region-level annotations come with a number of unique challenges. A video contains more information than can fit in a single frame, and video descriptions reflect that. They may reference objects that appear in a disjoint set of frames, as well as multiple persons and motions. To be more precise and produce finer-grained annotations, we annotate *noun phrases* (NP) (defined below) rather than simple object labels, as from Sec. 3.1.3. Moreover, one would ideally have dense region annotations at every frame, but the annotation cost in this case would be prohibitive for even small datasets. Therefore in practice, video datasets are typically sparsely annotated at the region level [112]. Favouring scale over density, we choose to annotate segments as sparsely as possible and annotate every noun phrase only in one frame inside each segment.

Noun Phrases. Following [89], we define noun phrases as short, non-recursive phrases that refer to a specific region in the image, able to be enclosed within a bounding box. They can contain a single instance or a group of instances and may include adjectives, determiners, pronouns or prepositions. For granularity, we further encourage the annotators to split complex NPs into their simplest form (*e.g.*"the man in a white shirt with a heart" can be split into three NPs: "the man", "a white shirt", and "a heart").

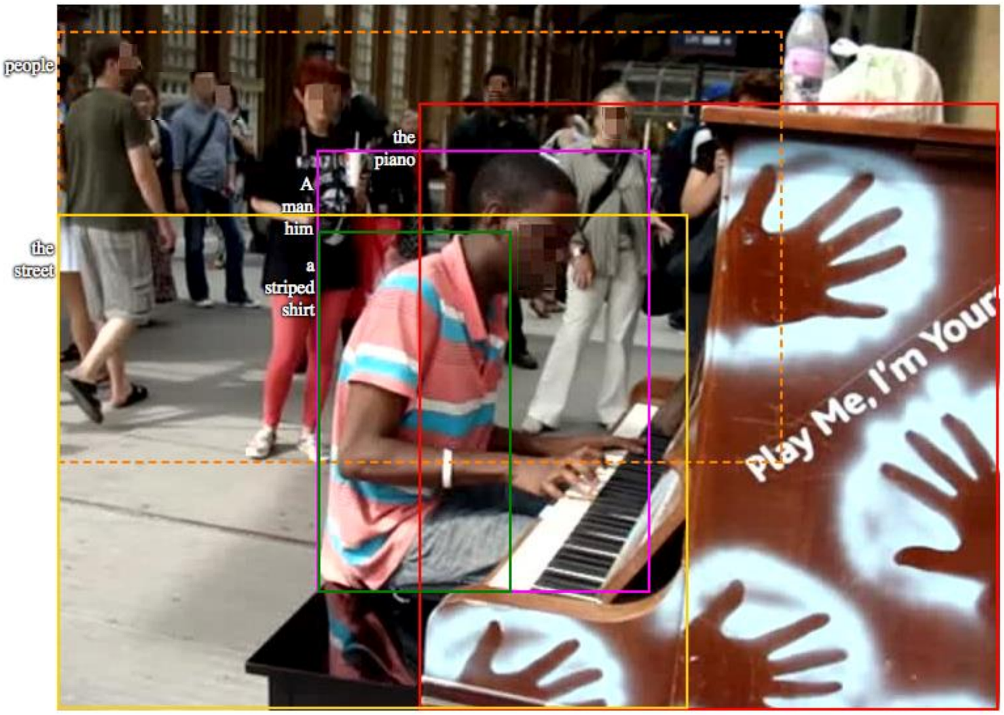
[1] ActivityNet-Entities is released at `https://github.com/facebookresearch/ActivityNet-Entities`.

Dataset	Domain	# Vid/Img	# Sent	# Obj	# BBoxes
Flickr30k Entities [89]	Image	32k	160k	480	276k
MPII-MD [72]	Video	≪1k	≪1k	4	2.6k
YouCook2 [25]	Video	2k	15k	67	135k
ActivityNet Humans [3]	Video	5.3k	30k	1	63k
ActivityNet-Entities (ours)	**Video**	**15k**	**52k**	**432**	**158k**
–train		10k	35k	432	105k
–val		2.5k	8.6k	427	26.5k
–test		2.5k	8.5k	421	26.1k

Table 3.2: Comparison of video description datasets with noun phrase or word-level grounding annotations. Our ActivityNet-Entities and ActivityNet Humans [3] dataset are both based on ActivityNet [4], but ActivityNet Humans provides boxes only for person on a small subset of videos. YouCook2 is restricted to cooking and only has box annotations for the val and the test splits.

3.2.2 Annotation Process

We uniformly sampled 10 frames from each video segment and presented them to the annotators together with the corresponding sentence. We asked the annotators to identify all concrete NPs from the sentence describing the video segment and then draw bounding boxes around them in *one* frame of the video where the target NPs can be clearly observed. Further instructions were provided including guidelines for resolving co-references within a sentence, *i.e.*, boxes may correspond to multiple NPs in the sentence (*e.g.*, a single box could refer to both “the man” and “him”) or when to use *multi-instance boxes* (*e.g.*, “crowd”, “a group of people” or “seven cats”). An annotated example is shown in Fig. 3.7. It is noteworthy that 10% of the final annotations refer to multi-instance boxes. We trained annotators, and deployed a rigid quality control by daily inspection and feedback. All annotations were verified in a second round. The full list of instructions provided to the annotators, validation process, as well as screen-shots of the annotation interface can be found in the Appendix B.



A man in a striped shirt is playing the piano on the street while people watch him.

Figure 3.7: An annotated example from our dataset. The dashed box ("people") indicates a group of objects.

3.2.3 Dataset Statistics and Analysis

As the test set annotations for the ActivityNet Captions dataset are not public, we only annotate the segments in the training (train) and validation (val) splits. This brings the total number of annotated videos in ActivityNet-Entities to 14,281. In terms of segments, we ended up with about 52k video segments with at least one NP annotation and 158k NP bounding boxes in total.

Respecting the original protocol, we keep as our training set the corresponding split from the ActivityNet Captions dataset. We further randomly & evenly split the original val set into our val set and our test set. We use all available bounding boxes for training our models, *i.e.*, including multi-instance boxes. Complete stats and comparisons with other related datasets can be found in Tab. 3.2.

From Noun Phrases to Objects Labels. Although we chose to annotate noun phrases, in this work, we model sentence generation as a word-level task. We follow the convention in [74] to determine the list of object classes and convert the NP label for box to a single-word object label. First, we select all nouns and pronouns from the NP annotations using the Stanford Parser [113]. The frequency of these words in the train and val splits are computed and a threshold determines whether each word is an object class. For ANet-Entities, we set the frequency threshold to be 50 which produces 432 object classes.

Stats on Object Annotations. The average number of annotated boxes per video segment is 2.56 and the standard deviation is 2.04. The average number of object labels per box is 1.17 and the standard deviation is 0.47. The top ten frequent objects are "man", "he", "people", "they", "she", "woman", "girl", "person", "it", and "boy", indicating that the dataset is human-centered. Note that these stats are on object boxes, *i.e.*, after pre-processing.

3.3 Miscellaneous

We list here other existing datasets used in our work, all based on images. Further reading on each dataset paper is encouraged for more information.

3.3.1 COCO Captions and Flickr30k Captions

COCO Captions [114] and Flickr30k Captions [115] are the two major benchmarks on image captioning. Both are collected by first gathering images from Flickr and then crowd-sourcing the image descriptoin. They each has 113.2k/5k/5k and 29.8k/1k/1k images for training/validation/testing respectively, following Karpathy's split.[2] Each image has five caption descriptions.

3.3.2 VQA 2.0

The VQA 2.0 dataset [104] has become the major benchmark for Visual Question Answering (VQA). The task is to given an image, answer a question related to the image in

[2] `cs.stanford.edu/people/karpathy/deepimagesent/captiondatasets.zip`

an open-ended format. We split the dataset with the official partition, i.e., 443.8k questions from 82.8k images for training, 214.4k questions from 40.5k images for validation and report results on the Test-Dev set and the Test-Standard set through the official evaluation server.

3.3.3 Flickr30k-Entities

Flickr30k-Entities [89] augments the original Flickr30k dataset with 276k entity-level bounding box annotations. The annotation format is similar to our ANet-Entities dataset, but on images only.

CHAPTER IV

Video Structure Learning through Event Proposal

4.1 Introduction

Action understanding remains an intensely studied problem-space, *e.g.*, action recognition [116, 117], action detection [118, 119, 46] and action labeling [43, 44, 45]. These works all emphasize instantaneous or short term actions, which clearly play a role in understanding short or structured videos [120]. However, for long, unconstrained videos, such as user-uploaded instructional videos of complex tasks—*preparing coffee* [28], *changing tires* [29]—learning the steps of accomplishing these tasks and their dependencies is essential, especially for agents' automatic acquisition of language or manipulation skills from video [57, 121, 122].

As defined in Sec. 1.3.2, *procedure* is the sequence of necessary steps comprising such a complex task, and each individual step is a *procedure segment*, or simply *segment* for convenience, inspired by [41, 29]. The problem of *Procedure Segmentation* is to automatically segment a video containing a procedure into category-independent procedure segments. Although this is a new problem, there are two related, existing problems: event proposal and procedure learning. The event proposal problem [1] is to localize category-independent temporal events from unconstrained videos. Both event proposals and procedure segments can contain multiple actions. However, the event proposal problem emphasizes the recall quality given a large amount of proposals, rather than the identification of a procedure (sequence of segments) from limited but necessary proposals. Events might overlap and are loosely-coupled but procedure segments barely overlap, are closely-coupled and usually have long-term dependencies.

The existing work in procedure learning is less-supervised than that of event proposals (no labels are given for the segments). It emphasizes video-subtitle alignment [39, 40] and discovery of common procedure steps of a specific process [29, 41]. However, the methods proposed in these works make restrictive assumptions: they typically assume either language is concurrently available, *e.g.*, from subtitles, or the number of procedure steps for a certain procedure is fixed, or both. Such assumptions are limited: extra textual input is unavailable in some scenarios; the subtitles or action sequences automatically generated by machines, *e.g.*, YouTube's ASR system, are inaccurate and require manual intervention; and many procedures of a certain type, such a specific recipe, will vary the number of steps in different instances (process variation).

Unfortunately, work in neither of these two problems sheds sufficient light on understanding procedure segmentation, as posed above. In this work, we directly focus on procedure segmentation. We propose a new dataset (see Sec. 3.1) of sufficient size and complexity to facilitate investigating procedure segmentation, and we present an automatic procedure segmentation method, called *Procedure Segmentation Networks*.

Procedure Segmentation Networks or *ProcNets* make neither of the assumptions made by existing procedure learning methods: we do not rely on available subtitles and we do not rely on knowledge of the number of segments in the procedure. *ProcNets* segments a long, unconstrained video into a sequence of category-independent procedure segments. ProcNets have three pieces: 1) context-aware frame-wise feature encoding; 2) procedure segment proposal for localizing segment candidates as start and end timestamps; 3) sequential prediction for learning the temporal structure among the candidates and generating the final proposals through a Recurrent Neural Network (RNN). The intuition is: when humans are segmenting a procedure, they first browse the video to have a general idea where

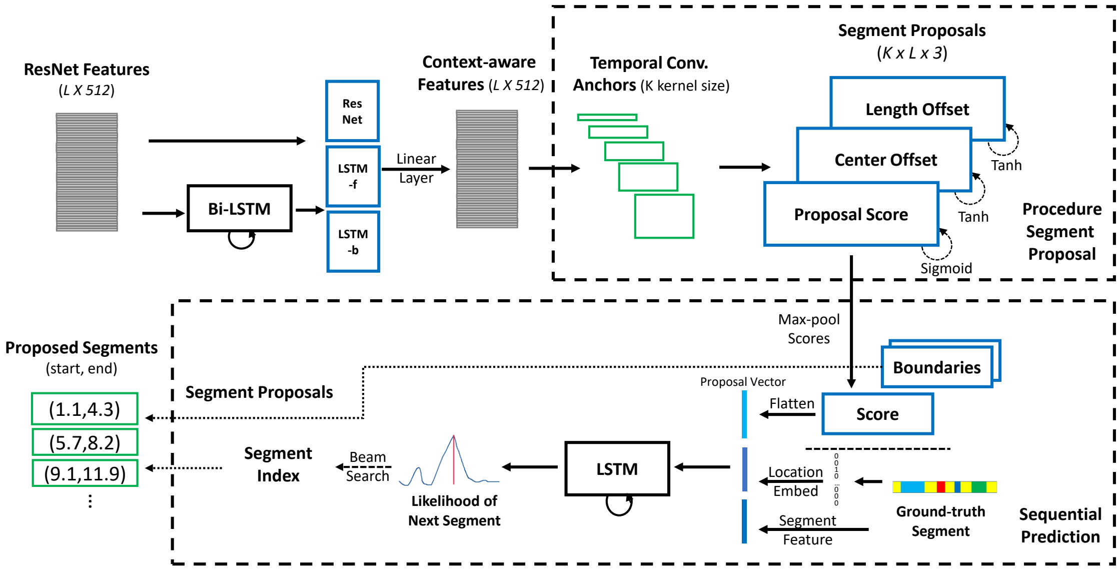


Figure 4.1: Schematic illustration of the ProcNets. The input are the frame-wise ResNet features (by row) for a video. The output are the proposed procedure segments. First, the bi-directional LSTM embeds the ResNet features into context-aware features. Then, the procedure segment proposal module generates segment candidates. Finally, the sequential prediction module selects the final proposals for output. During training, the ground-truth segments are embedded to composite the sequential prediction input, which are replaced with beam-searched segment in testing (as shown in the dashed arrows).

are the salient segments, which is done by our proposal module. Then they finalize the segment boundaries based on the dependencies among the candidates, i.e., which happens after which, achieved by our sequential prediction module.

For evaluation, we compare variants of our model with competitive baselines on standard metrics and the proposed methods demonstrate top performance against baselines. Furthermore, our detailed study suggests that ProcNets learn the structure of procedures as expected.

4.2 Procedure Segmentation Networks

We propose Procedure Segmentation Networks (ProcNets) for segmenting an untrimmed and unconstrained video into a sequence of procedure segments. We accomplish this by three core modules: 1) context-aware video encoding; 2) segment proposal module that localizes a handful of proposal candidates; 3) sequential prediction that predicts final segments based on segment-level dependencies among candidates. At training, ProcNets are given ground-truth procedure segment boundaries for each video; no recipe categories or segment descriptions are given. At testing, for any given unseen video, ProcNets propose and localize procedure segments in the video based on their visual appearance and temporal relations. The overall network structure is shown in Fig. 4.1 and next, we explain each component.

4.2.1 Context-Aware Video Encoding

Define a video as $\mathbf{x} = \{x_1, x_2, \ldots, x_L\}$, where L denotes the number of sampled frames and x_i is the frame-wise CNN feature vector with fixed encoding size. In this work $L = 500$ and encoding size is 512. We use ResNet [123] as the appearance feature extractor for its state-of-the-art performance in image classification. We then forward the ResNet features through a bi-directional long short-term memory (Bi-LSTM) [124] as context encoding. The outputs (forward and backward) are concatenated with the ResNet feature at each frame and the feature dimension is reduced to the same as ResNet feature for a fair comparison. We call these frame-wise *context-aware features*, denoted as $b_i = \text{Bi-LSTM}(\mathbf{x})$. Empirically, Bi-LSTM encoder outperforms context-free ResNet feature and LSTM-encoded feature by a relative 9% on our evaluation metric.

4.2.2 Procedure Segment Proposal

Inspired by the anchor-offset mechanism for spatial object proposal, such as in Faster R-CNN [47], we design a set of K explicit anchors for segment proposal. Each anchor has the length: l_k $(k = 1, 2, .., K)$ and their centers cover all the frames.

Each anchor-based proposal is represented by a proposal score and two offsets (center and length), from the output of a temporal convolution applied on the context-aware feature. The score indicates the likelihood for an anchor to be a procedure segment and the offsets are used to adjust the proposed segment boundaries. By zero-padding the video encoding at the boundaries (depending on anchor sizes), we obtain score and offset matrices of size $K \times L$ (see upper right of Fig. 4.1) respectively, and hence the output of proposal module is $K \times L \times 3$. Sigmoid function and Tanh functions are applied for proposal score and offsets, respectively.

We formulate the proposal generation as a classification problem and proposal offset as a regression problem. The segment proposals are classified as procedure segment or non-procedure segment with binary cross-entropy loss applied. During training, the segment proposals having at least 0.8 IoU (Intersection over Union) with any ground-truth segments are regarded as positive samples and these having IoU less than 0.2 with all the ground-truth are treated as negative samples. We randomly pick U samples from positive and negative separately for training. Then for the positive samples, we regress the proposed length and center offsets to the ground-truth ones from a relative scale. Given a ground-truth segment with center c_g and length l_g, the target offsets (θ_c, θ_l) w.r.t. anchor (center c_a and length l_a) are given by:

$$\theta_c = \frac{c_g - c_a}{l_a} \qquad \theta_l = \log \frac{l_g}{l_a} \quad . \tag{4.1}$$

Smooth l_1-loss [47] is applied in a standard way. For inference, the proposed offsets adjust the anchor location towards the final prediction location.

4.2.3 Sequential Prediction

Contrary to spatial objects, video procedure segments, by their nature, have strong temporal dependencies and yet ambiguous temporal boundaries. Therefore, we treat them differently. Recently, modeling frame-level temporal dependency in video has been explored [125]. However, memorizing dependencies over enormous frames is still challenging for recurrent models to date [118]. In contrast, we propose to learn segment-level

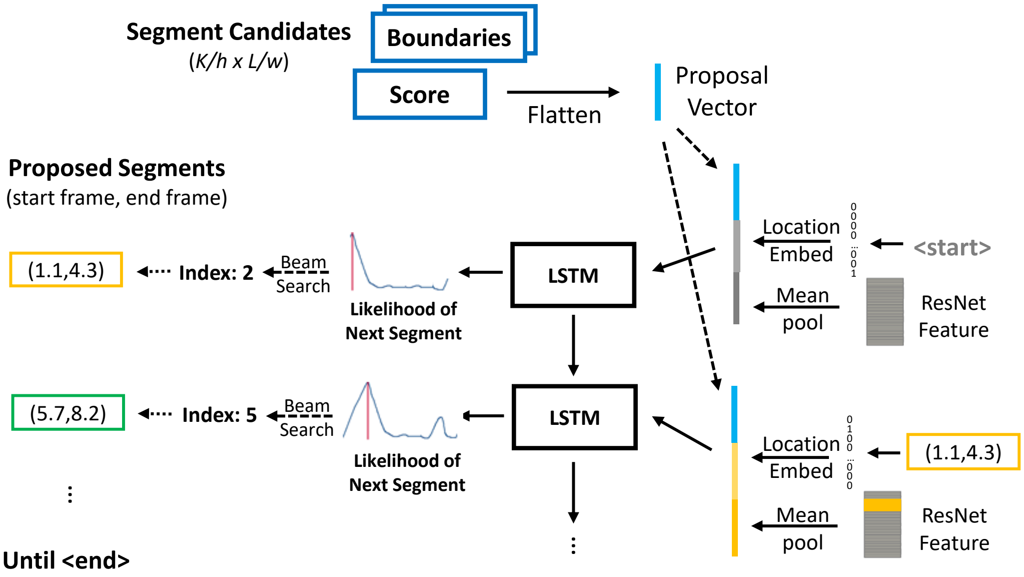


Figure 4.2: An example on sequential prediction during inference with unrolled LSTM. The <start> token is feed into model at time 0. The previously generated segment is feed into model at time 1. Best viewed in color.

dependency because the number of proposal segments could be smaller so learning dependencies over segments are more tractable. By leveraging the segment-level dependency, we predict the sequence of procedure segments while dynamically determine the number of segments to propose.

We use long short-term memory (LSTM) for sequential prediction due to its state-of-the-art performance in sequence modeling [20, 125]. The input of LSTM is constructed from three parts: 1) Proposal Vector $\mathbf{S}$: max-pooled proposal scores from the proposal module, fixed over time; 2) Location Embedding B_t: a set of vectors that discretely encode the locations of ground-truth or previously generated segments; 3) Segment Content C_t: the visual features of the ground-truth or previously generated segments. The tuple $(\mathbf{S}, B_t, C_t),\ t = 1, 2, ..., N$, is concatenated as the input to LSTM at each time step t. Intuitively, when we learn to choose a few winners from a pool of candidates, we need to know who and how good they are (Proposal Vector), what they look like (Segment Content) and the target candidates (location embedding). We will detail each component after introducing the overall model first.

The softmax output of LSTM is the likelihood of each proposal being the next segment prediction. Therefore, the likelihood for the entire procedure segment sequence $\epsilon_1, ..., \epsilon_S$ of a video can be formulated as:

$$\begin{aligned} &\log p(\epsilon_1, ..., \epsilon_S|\mathbf{S}) \\ &= \sum_{t=1}^{N} \log p(\epsilon_t|\mathbf{S}, B_{t-1}, C_{t-1}, \epsilon_0, ..., \epsilon_{t-1}) \ , \end{aligned} \tag{4.2}$$

where ϵ_0 is the special <start> segment token, B_0 is the embedding for the <start> token, C_0 is the meal-pooled video feature over all frames, B_{t-1} and C_{t-1} are determined by ϵ_{t-1}. The objective is to maximize the segment sequence likelihood for all training videos. We apply cross-entropy loss to the likelihood output P_t at time step t given the ground-truth segment index. During inference, we sample a sequence of segment indexes with greedy search while beam search does not improve further [126, 116], i.e., greedily picking the index with the maximal likelihood as the next proposed segment. The algorithm terminates when the special <end> token is picked. An example is shown in Fig. 4.2. Next, we describe the three input vectors in details.

Proposal Vector. As shown at the middle right of Fig. 4.1, we apply max-pooling to proposal score to filter out proposals with low proposal scores. The max-pooling kernel size is $h \times w$ and so as its stride, i.e., no overlapping. Empirically, $h = 8$ and w at 4 or 5 yields the best results. Given the filtered proposals (score and offsets), we flatten the proposal scores into a vector S by columns as Proposal Vector, which encodes the location and confidence information of all likely segment candidates in a video.

Location Embedding. During training, each ground-truth segment is represented by a one-hot vector where the index of one matches to the nearest proposal candidate as illustrated in Fig. 4.3. This discrete representation of location is easier to learn than continuous location values. Through a trainable embedding matrix (similar to word embedding in language modeling), this one-hot vector maps to a vector, which we call Location Embedding

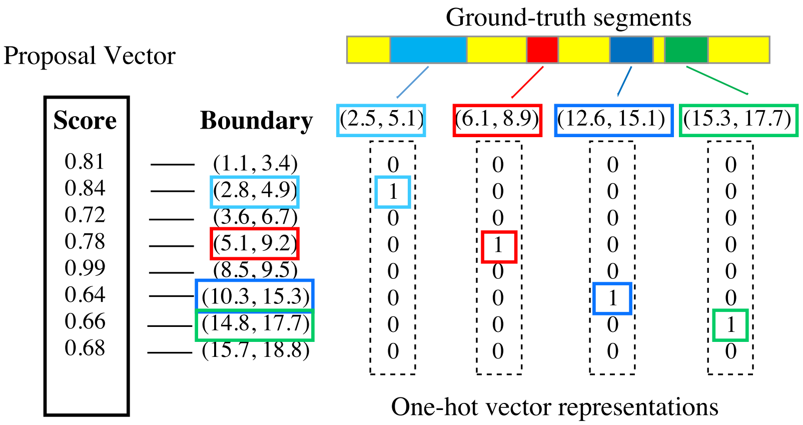


Figure 4.3: An example on converting ground-truth segments into one-hot vector representations from Proposal Vector.

vector and depicts the location information of a segment. This Location Embedding vector has the same size as the Proposal Vector. During testing, we greedily sample the softmax output of LSTM at previous time step to form location embedding for the current time step. Location Embedding represents the previous selected candidate, i.e., who we have and who we need next.

Segment Content. We then encode the visual content for the candidate represented in the one-hot vector. We mean-pool the video ResNet feature bounded by the start and end timestamps of the candidate. Its dimension is reduced to the same as Proposal Vector by a fully-connected layer. Segment Content indicates what the candidate looks like.

Relations to Other Models. To the best of our knowledge, we are the first to apply segment-level sequential modeling on category-independent procedure segments. The proposed model builds the video temporal structure without the need of knowing the hidden states such as in HMM. Note that there are other design choices.

Non-Maximal Suppression (NMS). In terms of proposal selection, a commonly adopted method in object detection [47] or action detection [46] is NMS. This approach fails to capture the temporal structure or segment dependencies of instructional videos. We consider it as a baseline in our experiment along with our sequential prediction model.

Other Time Sequence Models. Other methods for proposing segments have rigid model configurations, such as an HMM or pre-defined "grammar" for the whole video, which is infeasible for general video structure inference.

4.2.4 Loss Function

The loss function for procedure segmentation network consists of three parts, the binary cross-entropy loss for procedureness classification, the smooth $l1$-loss [47] for offset regression and the cross-entropy loss for sequential prediction. The formulations are as follows:

$$\begin{aligned} L &= L_{cla} + \alpha_r L_{reg} + \alpha_s L_{seq} \\ L_{cla} &= -\frac{1}{U_p + U_n}\left(\sum_{i=1}^{U_p} \log(S_i^{(pos)}) + \sum_{i=1}^{U_n} \log(1 - S_i^{(neg)})\right) \\ L_{reg} &= \frac{1}{U_p}\sum_{i=1}^{U_p} ||B_i - B_i^{(gt)}||_{smooth-l1} \\ L_{seq} &= -\frac{1}{N}\sum_{t=1}^{N} \log(P_t^T \mathbb{1}_t^{(gt)}) \end{aligned} \tag{4.3}$$

where U_p and U_n are the number of positive and negative samples, respectively, $S_i^{(pos)}$ and $S_i^{(neg)}$represents their scores, $B_i^{(gt)}$ is the ground-truth boundary corresponding to positive sample i, P_t is the softmax output of LSTM at time t and $\mathbb{1}_t^{(gt)}$ is one-hot vector of ground-truth segment index. Discount factors α_r and α_s are applied to balance the contributions of the regression loss and sequential prediction loss, respectively. Empirically, equally weighting each part, i.e. $\alpha_r = \alpha_s = 1$, yields good results.

4.3 Experiments and Results

In this section, we benchmark our new dataset YouCook2 [21] on procedure segmentation with competitive baselines and our proposed methods under standard metrics. We also show ablation studies, qualitative results and analysis on the procedure structure learned by our approach.

Baselines. We compare our methods against state-of-the-art methods in video summarization and action proposal due to lack of direct baselines in our new problem. These methods include: 1) Video Summarization LSTM (vsLSTM) [125], 2) Segment CNN for proposals (SCNN-prop) [46]. The major difference between ProcNets and vsLSTM is, our model learns the segment-level temporal dependency while vsLSTM learns the frame-level temporal dependency. SCNN-prop is the proposal module of action detector SCNN, which achieves state-of-the-art performance in action proposal.[1] In addition, we also evaluate a uniform segment baseline (denoted as Uniform). Two variants of ProcNets are evaluated, one with all the modules (ProcNets-LSTM) and one that replaces sequential prediction with NMS (ProcNets-NMS). Finally, note that we compare with no action segmentation methods since these approaches require an action pool and directly model the finite action states (*e.g.*, with HMM) which requires the "grammar" of the video procedure; both of these needs violate the core assumptions in this work.

Metrics. For procedure segmentation, we adopt two standard metrics for evaluating segment proposals: Jaccard [45] and mean Intersection over Union (mIoU). In Jaccard measure, the maximal intersection over prediction between all the final proposals and each ground-truth segment is computed and averaged. The individual Jaccard for each video is then averaged as the overall Jaccard. mIoU replaces the intersection over prediction in Jaccard with intersection over union (IoU). Hence, mIoU penalizes all the misalignment of segments while Jaccard only penalizes the partition of proposal beyond the ground truth. All the methods except for ProcNets-LSTM output 7 segments per video, determined by the average number of segments in the training set. Note that the average number of proposals from ProcNets-LSTM is also around 7, makes that a fair comparison. Inspired by the average recall metric in action proposal [49], we also report the proposal averaged

[1] New results comparing DAPs and SCNN-prop: `https://github.com/escorciav/daps/wiki`

recall, precision and F1 score but with limited segments (10 per video), as motivated in Introduction section.

Data Preprocessing. To preserve the overall information in the videos, we uniformly down-sample 500 frames for each video in YouCook2. The average sample rate is 1.58 fps. To further enlarge the training samples, we temporally augment the data, i.e., sample each video 10 times with temporal shifts. Then, we extract the frame-wise ResNet-34 feature [123],[2] pretrained on both ImageNet [127] and MSCOCO caption [128, 68]. Hence, each video is represented as a sequence of image spatial features. Local motion features are not used in our study; they may further improve performance.

Implementation and Training Details. The sizes of the temporal conv. kernels (also anchor length) are from 3 to 123 with an interval of 8, which covers 95% of the segment durations in training set. The 16 explicit anchors centered at each frame, i.e., stride for temporal conv. is 1. We randomly select $U = 100$ samples from all the positive and negative samples respectively and feed in negative samples if positive ones are less than U. Our implementation is in Torch. All the LSTMs have one layer and 512 hidden units. For hyper-parameters, the learning rate is 4×10^{-5}. We use the Adam optimizer [129] for updating weights with $\alpha = 0.8$ and $\beta = 0.999$. Note that we don't fine-tune the layers of the CNN which heavily slows down the training process.

4.3.1 Procedure Segmentation Results

We report the procedure segmentation results on both validation and testing sets in Tab. 4.1. The proposed ProcNets-LSTM model outperforms all other methods by a huge margin in both Jaccard and mIoU. SCNN-prop [46] suffers in our sequential segmentation task result from the lack of sequential modeling. vsLSTM [125] models frame-level temporal dependency and shows superior results than SCNN-prop. However, our model learns segment-level temporal dependency and yields better segmentation results, which shows

[2] Torch implementation of ResNet by Facebook: `https://github.com/facebook/fb.resnet.torch`

	validation		**test**	
Method (%)	**Jaccard**	**mIoU**	**Jaccard**	**mIoU**
Uniform	41.5	**36.0**	40.1	**35.1**
vsLSTM	47.2	33.9	45.2	32.2
SCNN-prop	46.3	28.0	45.6	26.7
ProcNets-NMS (ours)	**49.8**	35.2	**47.6**	33.9
ProcNets-LSTM (ours)	**51.5**	**37.5**	**50.6**	**37.0**

Table 4.1: Results on temporal segmentation. Top two scores are highlighted. See text for details.

	Jaccard	**mIoU**
Full model	50.6	37.0
-Proposal Vec	47.6	36.1
-Location Emb	46.2	35.1
-Segment Feat	49.0	36.4

Table 4.2: Ablation study on LSTM input. We remove either Proposal Vector (as *-Proposal Vec*), Location Embedding (as *-Location Emb*) or Segment Content (as *-Segment Feat*).

its effectiveness. Uniform baseline shows competitive results and the possible reason is, in instruction videos, generally procedures span the whole video which favors segments that can cover the majority of video. For rest of the experiments, all the results are on testing set.

Ablation study on sequential prediction. The input of the sequence modeling LSTM is the concatenation of three parts: Proposal Vector, Location Embedding and Segment Content. We remove either one of them as the ablation study. Results are shown in Tab. 4.2. Unsurprisingly, the proposal scores (Proposal Vector) play a significant role in determining the final proposals. When this information is unavailable, the overall performance drops by 6% on Jaccard relatively. The Location Embedding encodes the location information for ground-truth segments and is the most important component for procedure structure learning. Jaccard and mIoU scores drop by 8.7% and 5.1% relatively when location embedding is not available. The segment visual feature has less impact on the sequence prediction, which implies the visual information represented in the video appearance feature is noisy

Method (%)	Recall	Precision	F1
vsLSTM	22.1	**24.1**	23.0
SCNN-prop	**28.2**	23.2	**25.4**
ProcNets-NMS	**37.1**	**30.4**	**33.4**

Table 4.3: Results on segment localization accuracy. Top two scores are highlighted.

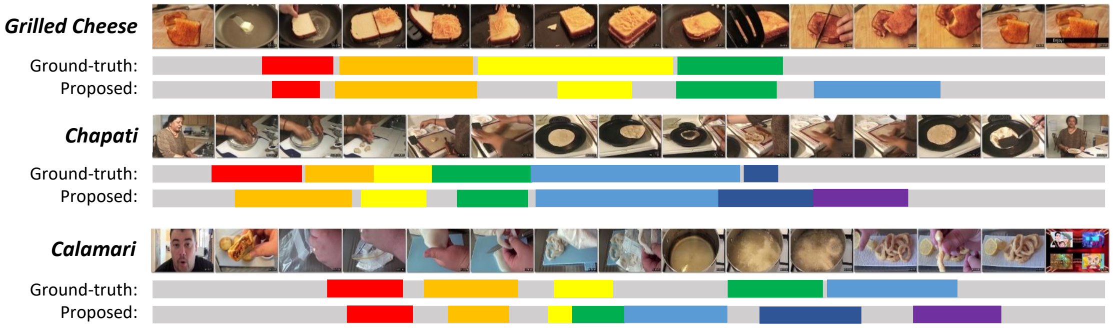


Figure 4.4: Qualitative results from test set. YouTube IDs: `BlTCkNkfmRY`, `jD4o_Lmy6bU` and `jrwHN188H2I`.

and less informative.

Proposal localization accuracy. We study the proposal localization problem when each model proposes 10 segments per video. Note that the metrics used here are not suitable for ProcNets-LSTM as they impose a fixed number of segments, where ProcNets-LSTM learns that automatically; nonetheless, we evaluate ProcNets-NMS for the quality of procedure segment proposal. The average recall, precision and F1 are shown in Tab. 4.3. The IoU threshold for true positive is 0.5. SCNN-prop shows competitive localization results as expected. vsLSTM yields inferior localization accuracy even though it performs better than SCNN-prop on segmentation. Our proposed model has more than 9% and 7% higher recall and precision than the baselines.

Qualitative results. We demonstrate qualitative results with videos from YouCook2 test set (see Fig. 4.4). The model can accurately localize some of the segments and predict their lengths. Moreover, the number of segments proposed is adapted to individual videos and the model learns to propose fewer segments at the beginning and the end of the video, where usually no cooking processes happen. In the example of *making Grilled Cheese*,

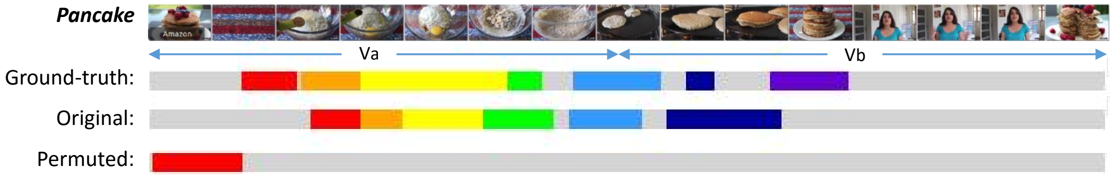


Figure 4.5: An example output of ProcNets on the original and the permutated video. YouTube ID: `ejq2ZsHgwFk`.

ProcNets propose the fifth segment to cover the process of cutting bread slices into two pieces. This trivial segment is not annotated but is still semantically meaningful.

Analysis on temporal structure learning. We conduct additional experiments to evaluate the temporal structure learning capability of ProcNets. For a given testing video, denote the first half as V_a and the second half as V_b. We inverse the order of V_aV_b to V_bV_a to construct the permutated video. We evaluate our model on both original test set and the permutated test set. The performance of pre-trained ProcNets decreases by over a half in the permutated set and 10%-20% of the videos only have segments predicted at the beginning of V_b (see Fig. 4.5). We believe reasons are two. First, the model captures the ending content in V_b and terminates the segment generation within V_b. Second, the temporal structure of V_a has no dependencies on V_b and hence is ignored by the model.

4.4 Discussion

We introduce a new problem called procedure segmentation to study human consensus on how a procedure is structured from unconstrained videos. Our proposed ProcNets take frame-wise video features as the input and predict procedure segments exist in the video. We evaluate the model against competitive baselines on the newly collected dataset YouCook2 with standard metrics and show significant improvements. Besides, ProcNets are capable of inferring the video structure by video content and modeling the temporal dependencies among procedure segments. There are potential extensions of our work for weakly-supervised learning and multi-modal learning. The first one is weakly supervised segmentation, which is to first align the weak subtitle signal with the video and then train our model with the aligned annotation. The other one is dense video description, which is covering in the following chapter.

CHAPTER V

Dense Video Description

5.1 Introduction

Video content consumes high cognitive bandwidth, and thus is slow for humans to digest. Although the visual signal itself can sometimes disambiguate certain semantics, one way to make video content more easily and rapidly understood by humans is to compress it in a way that retains the semantics. This is particularly important given the massive amount of video being produced everyday. Video summarization [125] is one way of doing this, but it loses the language components of the video, which are particularly important in instructional videos. Dense Video Description [1]—describing events in the video with descriptive natural language—is another way of achieving this compression while retaining the language components.

A Dense Video Description model contains two parts: event detection and event description. Existing methods tackle these two sub-problems using event proposal and captioning modules, and exploit two ways to combine them for Dense Video Description. One way is to train the two modules independently and generate descriptions for the best event proposals with the best captioning model [130]. The other way is to alternate training [1] between the two modules, *i.e.*, alternate between i) training the proposal module only and ii) training the captioning module on the positive event proposals while fine-tuning the proposal module. However, in either case, the language information cannot have direct

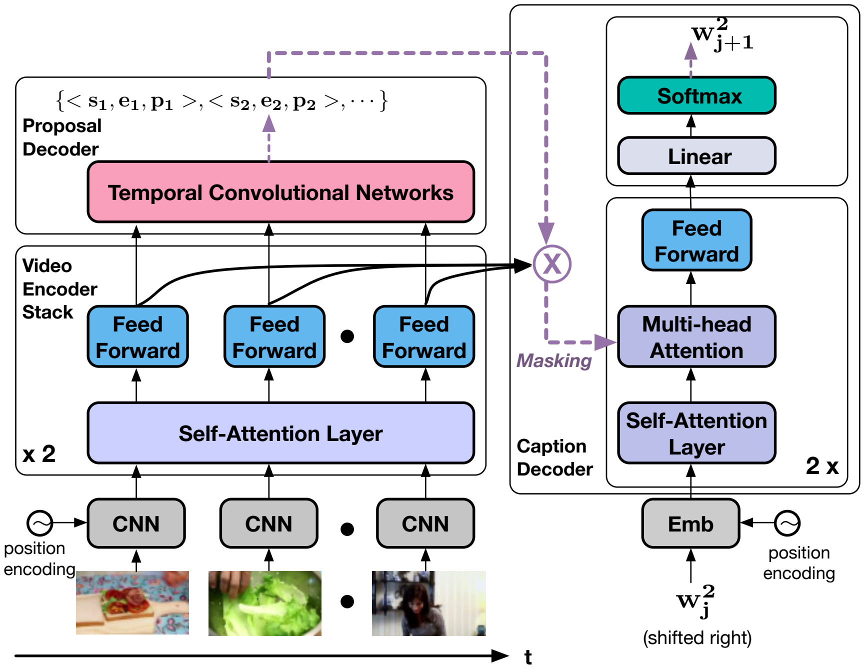


Figure 5.1: Dense Video Description is to localize (temporal) events from a video, which are then described with natural language sentences. We leverage temporal convolutional networks and self-attention mechanisms for precise event proposal generation and captioning.

impacts on the event proposal.

Intuitively, the video event segments and language are closely related and the language information should be able to help localize events in the video. To this end, we propose an encoder-decoder based end-to-end model for doing Dense Video Description (see Fig. 5.1). The encoder encodes the video frames (features) into the proper representation. The proposal decoder then decodes this representation with different anchors to form event proposals, *i.e.*, start and end time of the event, and a confidence score. The captioning decoder then decodes the proposal specific representation using a masking network, which converts the event proposal into a differentiable mask. This continuous mask enables both the proposal and captioning decoder to be trained consistently, *i.e.*, the proposal module now learns to adjust its prediction based on the quality of the generated caption. In other words, the language information from caption now is able to guide the visual model to generate more plausible proposals. In contrast to the existing methods where the proposal module solves a class-agnostic binary classification problem regardless the details in the video content, our model enforces the consistency between the content in the proposed video segment and the semantic information in the language description.

Another challenge for Dense Video Description, and more broadly for sequence modeling tasks, is the need to learn a representation that is capable of capturing long term dependencies. Recurrent Neural Networks (RNN) are possible solutions to this problem, however, learning such representation is still difficult [131]. *Self-attention* [132, 133, 82] allows for an attention mechanism within a module and is a potential way to learn this long-range dependence. In self-attention the higher layer in the same module is able to attend to all states below it. This made the length of the paths of states from the higher layer to all states in the lower layer to be one, and thus facilitates more effective learning. The shorter path length facilitates learning these dependencies because larger gradients can now pass to all states. Transformer [82] implements a fast self-attention mechanism and has demonstrated its effectiveness in machine translation. Unlike traditional sequential models, transformer does not require unrolling across time, and therefore trains and tests much faster as compared to RNN based models. We employ transformer in both the encoder and decoder of our model.

5.2 End-to-End Dense Video Description

Our end-to-end model is composed of three parts: a video encoder, a proposal decoder, and a captioning decoder that contains a mask prediction network to generate text description from a given proposal. The video encoder is composed of multiple self-attention layers. The proposal decoder takes the visual features from the encoder and outputs event proposals. The mask prediction network takes the proposal output and generates a differentiable mask for a certain event proposal. To make the decoder caption the current proposal, we then apply this mask by element-wise multiplication between it, the input visual embedding and all outputs from proposal encoder. In the following sections, we illustrate each component of our model in detail.

5.2.1 Preliminary

Our model relies heavily on the Transformer networks [82]. We strongly recommend you to read Appendix A for a detailed walk-through and examples if you are new to this topic.

5.2.2 Video Encoder

Each frame x_t of the video $X = \{x_1, \ldots, x_T\}$ is first encoded to a continuous representation $F^0 = \{f_1^0, \ldots, f_T^0\}$. It is then fed forward to L encoding layers, where each layer learns a representation $F^{l+1} = V(F^l)$ by taking input from previous layer l,

$$\text{V}(F^l) = \Psi(\text{PF}(\Gamma(F^l)), \Gamma(F^l)) \tag{5.1}$$

$$\Gamma(F^l) = \begin{pmatrix} \Psi(\text{MA}(f_1^l, F^l, F^l), f_1^l)^\top \\ \cdots \\ \Psi(\text{MA}(f_T^l, F^l, F^l), f_T^l)^\top \end{pmatrix}^\top \tag{5.2}$$

$$\Psi(\alpha, \beta) = \text{LayerNorm}(\alpha + \beta) \tag{5.3}$$

$$\text{PF}(\gamma) = M_2^l \max(0, M_1^l \gamma + b_1^l) + b_2^l \tag{5.4}$$

where $\Psi(\cdot)$ represents the function that performs layer normalization on the residual output, PF$(\cdot)$ denotes the 2-layered feed-forward neural network with ReLU nonlinearity for the first layer, MA$(\cdot)$ indicates the multi-head attention as detailed in Appendix A, M_1^l, M_2^l are the weights for the feed-forward layers, and b_1^l, b_2^l are the biases. Notice the self-attention used in Eq. 5.2. At each time step t, f_t^l is given as the query to the attention layer and the output is the weight sum of f_t^l, $t = 1, 2, ..., T$, which encodes not only the information regarding the current time step, but also all other time steps. Therefore, each time step of the output from the self-attention is able to encode all context information. In addition, it is easy to see that the length of the path between time steps is only one. In contrast to recurrent models, this makes the gradient update independent with respect to their position in time, and thus makes learning potential dependencies amongst distant frames easier.

5.2.3 Proposal Decoder

The event proposal decoder is based on our ProcNets from Chap. IV, for its state-of-the-art performance on long dense event proposals. We adopt the same anchor-offset mechanism as in ProcNets and design a set of N explicit anchors for event proposals. Each anchor-based proposal is represented by an event proposal score $P_e \in [0, 1]$ and two offsets: center θ_c and length θ_l. The associated anchor has length l_a and center c_a. The proposal boundaries (S_p, E_p) are determined by the anchor locations and offsets:

$$\begin{aligned} c_p = c_a + \theta_c l_a \quad & l_p = l_a \exp\{\theta_l\}, \\ S_p = c_p - l_p/2 \quad & E_p = c_p + l_p/2. \end{aligned} \tag{5.5}$$

These proposal outputs are obtained from temporal convolution (*i.e.*, 1-D convolutions) applied on the last layer output of the visual encoder. The score indicates the likelihood for a proposal to be an event. The offsets are used to adjust the proposed segment boundaries from the associated anchor locations. We made following changes to ProcNets:

- The sequential prediction module in ProcNets is removed, as the event segments in a video are not closely coupled and the number of events is small in general.
- Use input from a multi-head self-attention layer (see Appendix A) instead of a bidirectional LSTM (Bi-LSTM) layer [124].
- Use multi-layer temporal convolutions to generate the proposal score and offsets. The temporal convolutional network contain three 1-D conv. layers, with batch normalization [134]. We use ReLU activation for hidden layers.

- In our model, the conv. stride depends on kernel size ($\lceil\frac{kernel\ size}{s}\rceil$) versus always 1 in ProcNets.[1]

We encode the video context by a self-attention layer as it has potential to learn better context representation. Changing stride size based on kernel size reduces the number of longer proposals so that the training samples is more balanced, because a larger kernel size makes it easier to get good overlap with ground truth. It also speeds up training as the number of long proposals is reduced.

5.2.4 Captioning Decoder

Masked Transformer. The captioning decoder takes input from both the visual encoder and the proposal decoder. Given a proposal tuple (P_e, S_p, E_p) and visual representations $\{F^1, \dots, F^L\}$, the L-layered captioning decoder generates the t-th word by doing the following

$$Y_{\leq t}^{l+1} = \text{C}(Y_{\leq t}^l) = \Psi(\text{PF}(\Phi(Y_{\leq t}^l)), \Phi(Y_{\leq t}^l)) \tag{5.6}$$

$$\Phi(Y_{\leq t}^l) = \begin{pmatrix} \Psi(\text{MA}(\Omega(Y_{\leq t}^l)_1, \hat{F}^l, \hat{F}^l), \Omega(Y_{\leq t}^l)_1) \\ \cdots \\ \Psi(\text{MA}(\Omega(Y_{\leq t}^l)_t, \hat{F}^l, \hat{F}^l), \Omega(Y_{\leq t}^l)_t) \end{pmatrix} \tag{5.7}$$

$$\Omega(Y_{\leq t}^l) = \begin{pmatrix} \Psi(\text{MA}(y_1^l, Y^l, Y^l), y_1^l)^\top \\ \cdots \\ \Psi(\text{MA}(y_t^l, Y^l, Y^l), y_t^l)^\top \end{pmatrix} \tag{5.8}$$

$$\hat{F}^l = f_M(S_p, E_p) \odot F^l \tag{5.9}$$

$$p(w_{t+1}|X, Y_{\leq t}^L) = \text{softmax}(W^V y_{t+1}^L) \tag{5.10}$$

where y_i^0 represents word vector, $Y_{\leq t}^l = \{y_1^l, \dots, y_t^l\}$, w_{t+1} denotes the probability of each word in the vocabulary for time $t+1$, $W^V \in \mathbb{R}^{\nu \times d}$ denotes the word embedding matrix with

[1] s is a scalar that affects the convolution stride for different kernel size

vocabulary size ν, and $\odot$ indicates elementwise multiplication. $\mathrm{C}(\cdot)$ denotes the decoder representation, *i.e.*, the output from feed-forward layer in Fig. 5.1. $\Phi(\cdot)$ denotes the cross module attention that use the current decoder states to attend to encoder states (*i.e.*, multi-head attention in Fig. 5.1). $\Omega(\cdot)$ represents the self-attention in decoder. Notice that the subscript $\leq t$ restricts the attention only on the already generated words. $f_M : \mathbb{R}^2 \mapsto [0,1]^T$ is a masking function that output values (near) zero when outside the predicted starting and ending locations, and (near) one otherwise. With this function, the receptive region of the model is restricted to the current segment so that the visual representation focuses on describing the current event. Note that during decoding, the encoder performs the forward propagation again so that the representation of each encoder layer contains only the information for the current proposal (see Eq. 5.9). This is different from simply multiplying the mask with the existing representation from the encoder during proposal prediction, since the representation of the latter still contains information that is outside the proposal region. The representation from the L-th layer of captioning decoder is then used for predicting the next word for the current proposal using a linear layer with softmax activation (see Eq. 5.10).

Differentiable Proposal Mask. We cannot choose any arbitrary function for f_M as a discrete one would prevent us from doing end-to-end training. We therefore propose to use a fully differentiable function to obtain the mask for visual events. This function f_M maps the predicted proposal location to a differentiable mask $M \in \mathbb{R}^T$ for each time step $i \in \{1, \ldots, T\}$.

$$\begin{aligned} & f_M(S_p, E_p, S_a, E_a, i) = \sigma(g(\\ & [\rho(S_p, :), \rho(E_p, :), \rho(S_a, :), \rho(E_a, :), \mathrm{Bin}(S_a, E_a, :)])) \end{aligned} \tag{5.11}$$

$$\rho(pos, i) = \begin{cases} \sin(pos/10000^{i/d}) & i \textit{ is even} \\ \cos(pos/10000^{(i-1)/d}) & \textit{otherwise} \end{cases} \tag{5.12}$$

$$\text{Bin}(S_a, E_a, i) = \begin{cases} 1 & \textit{if}\, i \in [S_a, E_a] \\ 0 & otherwise \end{cases} \tag{5.13}$$

where S_a and E_a are the start and end position of anchor, $[\cdot]$ denotes concatenation, $g(\cdot)$ is a continuous function, and $\sigma(\cdot)$ is the logistic sigmoid function. We choose to use a multilayer perceptron to parameterize g. In other words, we have a feed-forward neural network that takes the positional encoding from the anchor and predicted boundary positions and the corresponding binary mask to predict the continuous mask. We use the same positional encoding strategy as in [82].

Directly learning the mask would be difficult and unnecessary, since we would already have a reasonable boundary prediction from the proposal module. Therefore, we use a gated formulation that lets the model choose between the learned continuous mask and the discrete mask obtained from the proposal module. More precisely, the gated masking function f_{GM} is

$$\begin{aligned} &f_{GM}(S_p, E_p, S_a, E_a, i) = \\ &P_e \text{Bin}(S_p, E_p, i) + (1 - P_e) f_M(S_p, E_p, S_a, E_a, i) \end{aligned} \tag{5.14}$$

Since the proposal score $P_e \in [0, 1]$, it now acts as a gating mechanism. This can also be viewed as a modulation between the continuous and proposal masks, the continuous mask is used as a supplement for the proposal mask in case the confidence is low from the proposal module.

5.2.5 Model Learning

Our model is fully differentiable and can be trained consistently from end-to-end.[2] The event proposal anchors are sampled as follows. Anchors that have overlap greater than 70% with any ground-truth segments are regarded as positive samples and ones that have less

[2] Source code is made available at `https://github.com/LuoweiZhou/densecap`

than 30% overlap with all ground-truth segments are negative. The proposal boundaries for positive samples are regressed to the ground-truth boundaries (offsets). We randomly sample $U = 10$ anchors from positive and negative anchor pools that correspond to one ground-truth segment for each mini-batch.

The loss for training our model has four parts: the regression loss $\mathcal{L}_r$ for event boundary prediction, the binary cross entropy mask prediction loss $\mathcal{L}_m$, the event classification loss $\mathcal{L}_e$ (*i.e.*, prediction P_e), and the captioning model loss $\mathcal{L}_c$. The final loss $\mathcal{L}$ is a combination of these four losses,

$$\mathcal{L}_r = \text{Smooth}_{\ell 1}(\hat{\theta}_c, \theta_c) + \text{Smooth}_{\ell 1}(\hat{\theta}_l, \theta_l) \tag{5.15}$$

$$\mathcal{L}_m^i = \text{BCE}(Bin(S_p, E_p, i), f_M(S_p, E_p, S_a, E_a, i)) \tag{5.16}$$

$$\mathcal{L}_e = \text{BCE}(\hat{P}_e, P_e) \tag{5.17}$$

$$\mathcal{L}_c^t = \text{CE}(\hat{w}_t, p(w_t | X, Y^L_{\leq t-1})) \tag{5.18}$$

$$\mathcal{L} = \lambda_1 \mathcal{L}_r + \lambda_2 \sum_i \mathcal{L}_m^i + \lambda_3 \mathcal{L}_e + \lambda_4 \sum_t \mathcal{L}_c^t \tag{5.19}$$

where $\text{Smooth}_{\ell 1}$ is the smooth ℓ_1 loss defined in [135], BCE denotes binary cross entropy, CE represents cross entropy loss, $\hat{\theta}_c$ and $\hat{\theta}_l$ represent the ground-truth center and length offset with respect to the current anchor, $\hat{P}_e$ is the ground-truth label for the proposed event, $\hat{w}_t$ denotes the ground-truth word at time step t, and $\lambda_{1...4} \in \mathbb{R}^+$ are the coefficients that balance the contribution from each loss.

Simple Single Stage Models. The key for our proposed model to work is not the single stage learning of a compositional loss, but the ability to keep the consistency between the proposal and captioning. For example, we could make a single-stage trainable model by simply sticking them together with multi-task learning. More precisely, we can have the same model but choose a non-differentiable masking function f_M in Eq. 5.9. The same training procedure can be applied for this model (see the following section). Since the masking function would then be non-differentiable, error from the captioning model cannot be back propagated to modify the proposal predictions. However, the captioning decoder is still able to influence the visual representation that is learned from the visual encoder. This may be undesirable, as the updates the visual representation may lead to worse performance for the proposal decoder. As a baseline, we also test this single-stage model in our experiments.

5.3 Implementation Details

For the proposal decoder, the temporal convolutional networks take the last encoding output from video encoder as the input. The sizes of the temporal convolution kernels vary from 1 to 251 and we set the stride factor s to 50. For our Transformer model, we set the model dimension $d = 1024$ (same as the Bi-LSTM hidden size) and set the hidden size of feed-forward layer to 2048. We set number of heads (H) to 8. In addition to the residual dropout and attention dropout layers in Transformer, we add a 1-D dropout layer at the visual input embedding to avoid overfitting. We use recurrent dropout proposed in [136] for this 1-D dropout. Due to space limits, more details are included in the supplementary material.

5.4 Experiments

5.4.1 Datasets

ActivityNet Captions and YouCook2 are the two largest datasets with temporal event segments annotated and described by natural language sentences, detailed in Chap. III. ActivityNet Captions contains 20k videos, and on average each video has 3.65 events annotated. YouCook2 has 2k videos and the average number of segments per video is 7.70. The train/val/test splits for ActivityNet Captions are 0.5:0.25:0.25 while for YouCook2 are 0.66:0.23:0.1. We report our results from both datasets on the validation sets. For ActivityNet Captions, we also show the testing results on the evaluation server while the testing set for YouCook2 is not available.

Data Preprocessing. We down-sample the video every 0.5s and extract the 1-D appearance and optical flow features per frame, as suggested by Xiong et al. [137]. For appearance features, we take the output of the "Flatten-673" layer in ResNet-200 [123]; for optical flow features, we extract the optical flow from 5 contiguous frames [138], encode with BN-Inception [134] and take output of the "global-pool" layer. Both networks are pre-trained on the ActivityNet dataset [4] for the action recognition task. We then concatenate the two feature vector and further encode with a linear layer. We set the window size T to 480. The input is zero padded in case the number of sampled frames is smaller than the size of the window. Otherwise, the video is truncated to fit the window. Note that we do not fine-tune the visual features for efficiency considerations, however, allowing fine-tuning may lead to better performance.

5.4.2 Baseline and Metrics

Baselines. Most of the existing methods can only caption an entire video or specified video clip. For example, LSTM-YT [139], S2YT [140], TempoAttn [63], H-RNN [84] and DEM [1]. The most relevant baseline is TempoAttn, where the model temporally attends on visual sequence inputs as the input of LSTM language encoder. For a fair comparison, we made the following changes to the original TempoAttn. First, all the methods take the same visual feature input. Second, we add a Bi-LSTM context encoder to TempoAttn while our method use self-attention context encoder. Third, we apply temporal attention on Bi-LSTM output for all the language decoder layers in TempoAttn since our decoder has attention each layer. We name this baseline Bi-LSTM+TempoAttn. Since zero inputs deteriorates Bi-LSTM encoding, we only apply the masking on the output of the LSTM encoder when it is passed to the decoder. We also compare with a simple single-stage Masked Transformer baseline as mentioned in section 5.2.5, where the model employs a discrete binary mask.

Method	B@3	B@4	M
Bi-LSTM +TempoAttn	2.43	1.01	7.49
Masked Transformer	4.47	2.14	9.43
End-to-end Masked Transformer	**4.76**	**2.23**	**9.56**

Table 5.1: Captioning results from ActivityNet Caption Dataset with learned event proposals. All results are on the validation set and all our models are based on 2-layer Transformer. We report BLEU (B) and METEOR (M). All results are on the validation set. Top scores are highlighted. The improvements of our methods over the baseline method are statistically significant (p-value$\ll$0.05).

For event proposals, we compare our self-attention transformer-based model with ProcNets and our own baseline with Bi-LSTM. For captioning-only models, we use the same baseline as the full Dense Video Description but instead, replace the learned proposals with ground-truth proposals. Results for other dense captioning methods (*e.g.*the best published method DEM [1]) are not available on the validation set nor is the source code released. So, we compare our methods against those methods that participated in CVPR 2017 ActivityNet Video Dense-captioning Challenge [130] for test set performance on ActivityNet.

Evaluation Metrics. For ground-truth segment captioning, we measure the captioning performance with most commonly-used evaluation metrics: BLEU$\{3,4\}$ and METEOR. For dense captioning, the evaluate metric takes both proposal accuracy and captioning accuracy into account. Given a tIoU threshold, if the proposal has an overlapping larger than the threshold with any ground-truth segments, the metric score is computed for the generated sentence and the corresponding ground-truth sentence. Otherwise, the metric score is set to 0. The scores are then averaged across all the proposals and finally averaged across all the tIoU thresholds–0.3, 0.5, 0.7, 0.9 in this case.

[3] This work is unpublished by Nov. 2017. It employs external data for model training and the final prediction is obtained from an ensemble of models.

Method	METEOR
DEM [1]	4.82
Wang et al.	9.12
Jin et al.	9.62
Guo et al.	9.87
Yao et al.[3](Ensemble)	12.84
Our Method	**10.12**

Table 5.2: Dense Video Description challenge leader board results. For results from the same team, we keep the highest one.

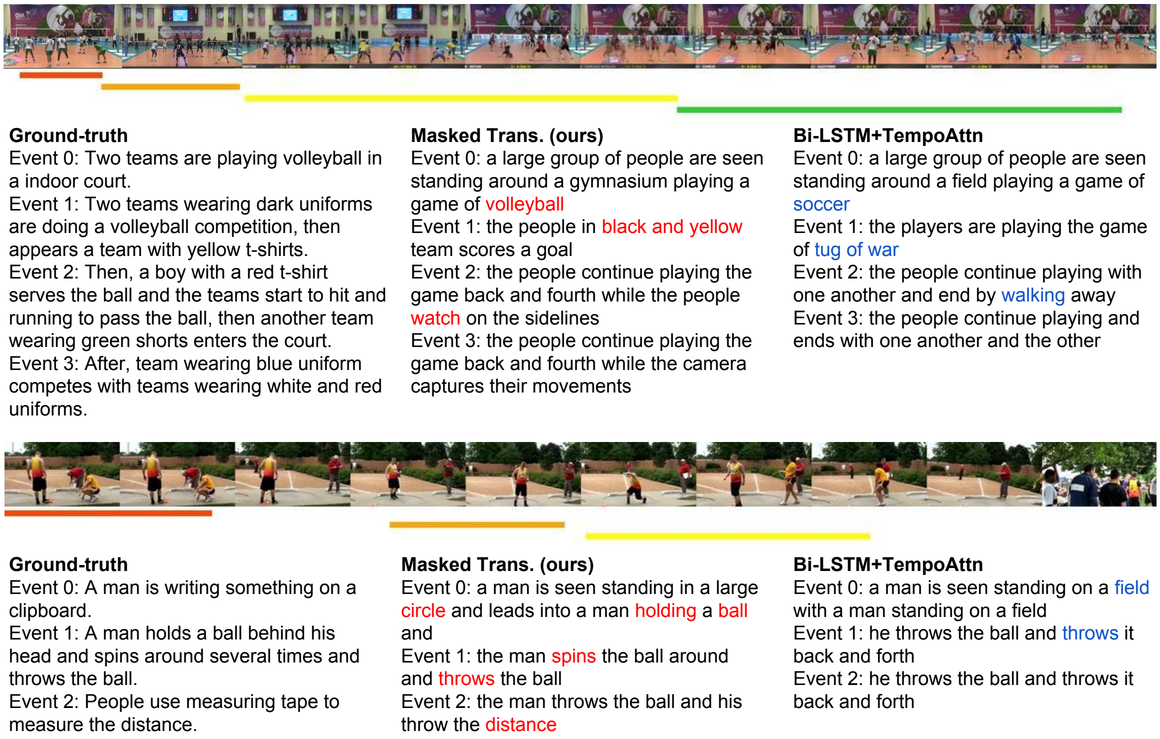
Method	**GT Proposals**		**Learned Proposals**	
	B@4	M	B@4	M
Bi-LSTM +TempoAttn	0.87	8.15	0.08	4.62
Our Method	**1.42**	**11.20**	**0.30**	**6.58**

Table 5.3: Recipe generation benchmark on YouCook2 validation set. GT proposals indicate the ground-truth segments are given during inference. The improvement of our method over the baseline method is statistically significant (p-value$\ll$0.05).

5.4.3 Comparison with State-of-the-Art Methods

We compare our proposed method with baselines on the ActivityNet Caption dataset. The validation and testing set results are shown in Tab. 5.1 and 5.2, respectively. All our models outperform the LSTM-based models by a large margin, which may be attributed to their better ability of modeling long-range dependencies.

We also test the performance of our model on the YouCook2 dataset, and the result is shown in Tab. 5.3. Here, we see similar trend on performance. Our transformer based model outperforms the LSTM baseline by a significant amount. However, the results on learned proposals are much worse as compared to the ActivityNet dataset. This is possibly because of small objects, such as utensils and ingredients, are hard to detect using global visual features but are crucial for describing a recipe. Hence, one future extension for our work is to incorporate object detectors/trackers [141, 57] into the current captioning system.



Ground-truth
Event 0: Two teams are playing volleyball in a indoor court.
Event 1: Two teams wearing dark uniforms are doing a volleyball competition, then appears a team with yellow t-shirts.
Event 2: Then, a boy with a red t-shirt serves the ball and the teams start to hit and running to pass the ball, then another team wearing green shorts enters the court.
Event 3: After, team wearing blue uniform competes with teams wearing white and red uniforms.

Masked Trans. (ours)
Event 0: a large group of people are seen standing around a gymnasium playing a game of volleyball
Event 1: the people in black and yellow team scores a goal
Event 2: the people continue playing the game back and fourth while the people watch on the sidelines
Event 3: the people continue playing the game back and fourth while the camera captures their movements

Bi-LSTM+TempoAttn
Event 0: a large group of people are seen standing around a field playing a game of soccer
Event 1: the players are playing the game of tug of war
Event 2: the people continue playing with one another and end by walking away
Event 3: the people continue playing and ends with one another and the other

Ground-truth
Event 0: A man is writing something on a clipboard.
Event 1: A man holds a ball behind his head and spins around several times and throws the ball.
Event 2: People use measuring tape to measure the distance.

Masked Trans. (ours)
Event 0: a man is seen standing in a large circle and leads into a man holding a ball and
Event 1: the man spins the ball around and throws the ball
Event 2: the man throws the ball and his throw the distance

Bi-LSTM+TempoAttn
Event 0: a man is seen standing on a field with a man standing on a field
Event 1: he throws the ball and throws it back and forth
Event 2: he throws the ball and throws it back and forth

Figure 5.2: Qualitative results on ActivityNet Captions. The color bars represent different events. Colored text highlight relevant content to the event. Our model generates more relevant attributes as compared to the baseline.

We show qualitative results in Fig. 5.2 where the proposed method generates captions with more relevant semantic information. More visualizations are in the supplementary.

5.4.4 Model Analysis

In this section we perform experiments to analyze the effectiveness of our model on different sub-tasks of Dense Video Description.

Video Event Proposal. We first evaluate the effect of self-attention on event proposal, and the results are shown in Tab. 5.4. We use standard average recall (AR) metric [49, 130] given 100 proposals. Bi-LSTM indicates our improved ProcNets-prop model by using temporal convolutional and large kernel strides. We use our full model here, where the context encoder is replaced by our video encoder. We have noticed that the anchor sizes have a large impact on the results. So, for fair comparison, we maintain the same anchor sizes across

Method	Average Recall (%)
ProcNets-prop [21]	47.01
Bi-LSTM (ours)	50.65
Self-Attn (our)	**52.95**

Table 5.4: Event proposal results from ActivityNet Captions dataset. We compare our proposed methods with our baseline method ProcNets-prop on the validation set.

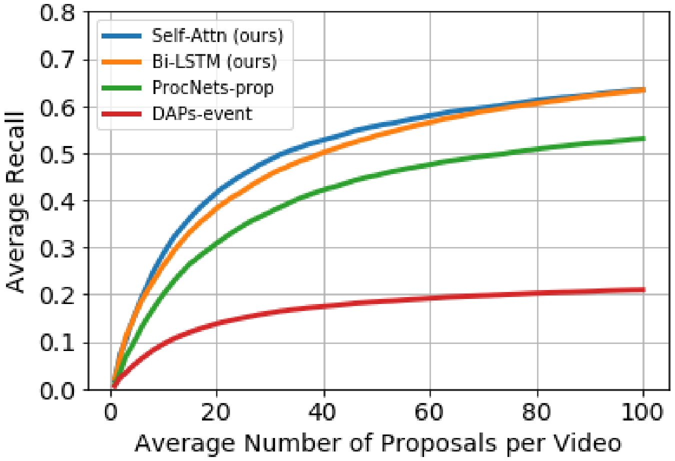


Figure 5.3: Event proposal recall curve under tIoU threshold 0.8 with average 100 proposals per video.

all three methods. Our proposed Bi-LSTM model gains a 7% relative improvement from the baseline results from the deeper proposal network and more balanced anchor candidates. Our video encoder further yields a 4.5% improvement from our recurrent nets-based model. We show the recall curve under high tIoU threshold (0.8) in Fig. 5.3 follow the convention [1]. DAPs [49], is initially proposed for short action proposals and adapted later for long event proposal [1]. The proposed models outperforms DAPs-event and ProcNets-prop by significant margins. Transformer based and Bi-LSTM based models yield similar recall results given sufficient number of proposals (100), while our self-attention encoding model is more accurate when the allowed number of proposals is small.

Dense Video Description. Next, we look at the Dense Video Description results in an ideal setting: doing the captioning based on the ground-truth event segments. This will give us an ideal captioning performance since all event proposals are accurate. Because we need

Method	B@3	B@4	M
Bi-LSTM +TempoAttn	4.8	2.1	10.02
Our Method			
1-layer	**5.80**	2.66	10.92
2-layer	5.69	2.67	11.06
4-layer	**5.70**	**2.77**	**11.11**
6-layer	5.66	**2.71**	**11.10**

Table 5.5: Captioning results from ActivityNet Caption Dataset with ground-truth proposals. All results are on the validation set. Top two scores are highlighted. The improvement of our selected method (2-layer) over the baseline method is statistically significant (p-value$\ll$0.05).

	GT Proposals		**Learned Proposals**	
Method	B@4	M	B@4	M
Bi-LSTM +TempoAttn	0.84	5.39	0.42	3.99
Our Method	**1.13**	**5.90**	**1.04**	**5.93**

Table 5.6: Evaluating only long events from ActivityNet Caption Dataset. GT proposals indicate the ground-truth segments are given during inference.

access to ground-truth event proposal during test time, we report the results on validation set (see Tab. 5.5).[4] The proposed Masked Transformer (section 5.2.4) outperforms the baseline by a large margin (by more than 1 METEOR point). This directly substantiates the effectiveness of the transformer on both visual and language encoding and multi-head temporal attention. We notice that as the number of encoder and decoder layers increases, the performance gets further boosts by 1.3%-1.7%. As can be noted here, the 2-layer transformer strikes a good balance point between performance and computation, and thus we use 2-layer transformer for all our experiments.

Analysis on Long Events. As mentioned in section 5.2.2, learning long-range dependencies should be easier with self-attention, since the next layer observes information from

[4] The results are overly optimistic, however, it is fine here since we are interested in the best situation performance. The comparison is also fair, since all methods are tuned to optimize the validation set performance.

all time steps of the previous layer. To validate this hypothesis directly, we test our model against the LSTM baseline on longer event segments (where the events are at least 50s long) from the ActivityNet Caption dataset, where learning the long-range dependencies are crucial for achieving good performance. It is clear from the result (see Tab. 5.6) that our transformer based model performs significantly better than the LSTM baseline. The discrepancy is even larger when the model needs to learn both the proposal and captioning, which demonstrate the effectiveness of self-attention in facilitate learning long range dependencies.

5.5 Discussion

We propose an end-to-end model for Dense Video Description. The model is composed of an encoder and two decoders. The encoder encodes the input video to proper visual representations. The proposal decoder then decodes from this representation with different anchors to form video event proposals. The captioning decoder employs a differentiable masking network to restrict its attention to the proposal event, ensures the consistency between the proposal and captioning during training. We achieved significant performance improvement on both event proposal and captioning tasks as compared to RNN-based models. We demonstrate the effectiveness of our models on ActivityNet Captions and YouCook2 dataset. Despite the progress, one issue we suffer with most of concurrent end-to-end systems is that we have not shed enough light on why the system is performing well due to the low system interpretability. In the next chapter, we explore how could visual grounding further improve the model performance and interpretability.

CHAPTER VI

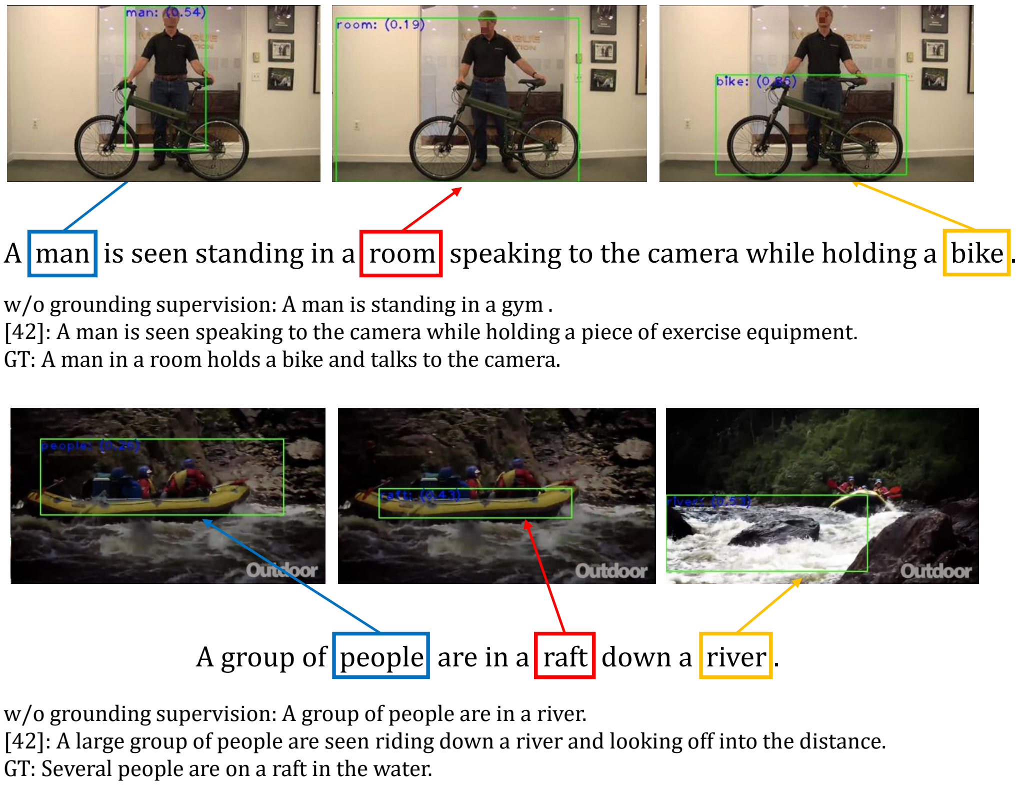
Grounded Video Description

6.1 Introduction

Image and video description models are frequently not well grounded [80] which can increase their bias [142] and lead to hallucination of objects [143], *i.e.*, the model mentions objects which are not in the image or video *e.g.*because they might have appeared in similar contexts during training. This makes models less accountable and trustworthy, which is important if we hope such models will eventually assist people in need [144, 145]. Additionally, grounded models can help to explain the model's decisions to humans and allow humans to diagnose them [146]. While researchers have started to discover and study these problems for image description [80, 142, 143, 146],[1] they are even more pronounced for video description due to the increased difficulty and diversity, both on the visual and the language side.

Fig. 6.1 illustrates this problem. A video description approach (without grounding supervision) generated the sentence "A man standing in a gym" which correctly mentions "a man" but hallucinates "gym" which is not visible in the video. Although a man is in the video it is not clear if the model looked at the bounding box of the man to say this word [142, 143]. For the sentence "A man [...] is playing the piano" in Fig. 6.2, it is impor-

[1] We use *description* instead of *captioning* as *captioning* is often used to refer to transcribing the speech in the video, not *describing* the content.



A man is seen standing in a room speaking to the camera while holding a bike.

w/o grounding supervision: A man is standing in a gym .
[42]: A man is seen speaking to the camera while holding a piece of exercise equipment.
GT: A man in a room holds a bike and talks to the camera.

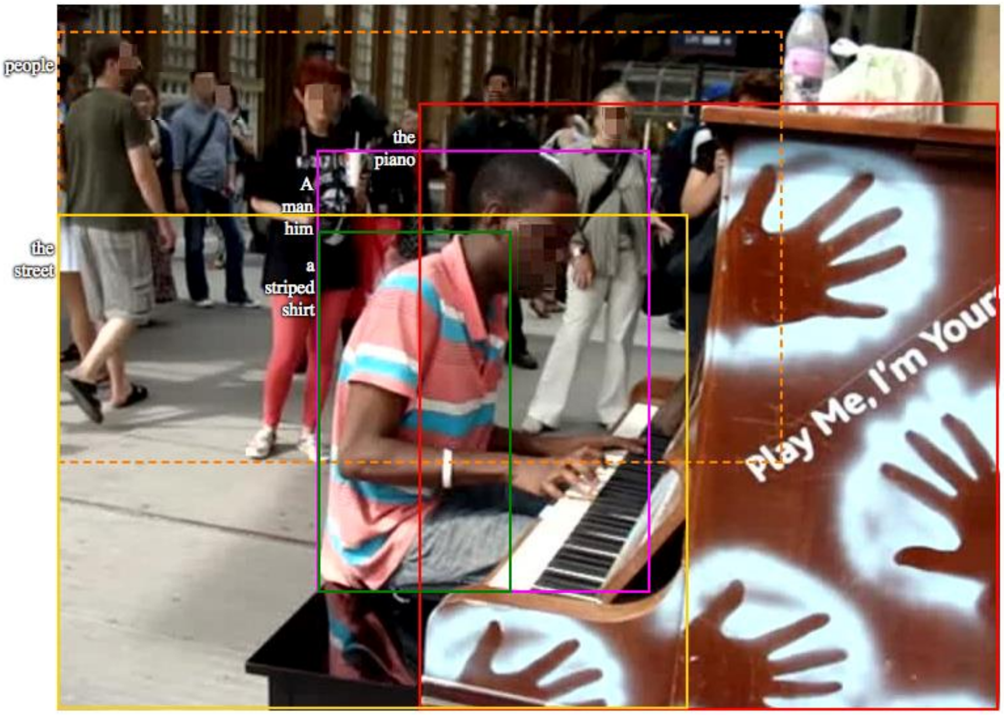
A group of people are in a raft down a river.

w/o grounding supervision: A group of people are in a river.
[42]: A large group of people are seen riding down a river and looking off into the distance.
GT: Several people are on a raft in the water.

Figure 6.1: Word-level grounded video descriptions generated by our model on two segments from our ActivityNet-Entities dataset. We also provide the descriptions generated by our model without explicit bounding box supervision, the descriptions generated by [2] and the ground-truth descriptions (GT) for comparison.

tant to understand that which "man" in the image "A man" is referring to, to determine if a model is correctly grounded. Such understanding is crucial for many applications when trying to build accountable systems or when generating the next sentence or responding to a follow up question of a blind person: *e.g.*answering "Is *he* looking at me?" requires an understanding which of the people in the image the model talked about.

The goal of our research is to build such grounded systems. As one important step in this direction, we collect ActivityNet-Entities (short as ANet-Entities) which grounds or links noun phrases in sentences with bounding boxes in the video frames. It is based on ActivityNet Captions [1], one of the largest benchmarks in video description. When annotating objects or noun phrases we specifically annotate the bounding box which corresponds to the instance referred to in the sentence rather than all instances of the same



A man in a striped shirt is playing the piano on the street while people watch him.

Figure 6.2: An annotated example from our dataset. The dashed box ("people") indicates a group of objects. Same as Fig. 3.7.

object category, *e.g.*in Fig. 6.2, for the noun phrase "the man" in the video description, we only annotate the sitting man and not the standing man or the woman, although they are all from the object category "person". We note that annotations are sparse, in the sense that we only annotate a single frame of the video for each noun phrase. ANet-Entities has a total number of 51.8k annotated video segments/sentences with 157.8k labeled bounding boxes, more details can be found in Sec. 3.2.1.

Our new dataset allows us to introduce a novel grounding-based video description model that learns to jointly generate words and refine the grounding of the objects generated in the description. We explore how this explicit supervision can benefit the description generation compared to unsupervised methods that might also utilize region features but do not penalize grounding.

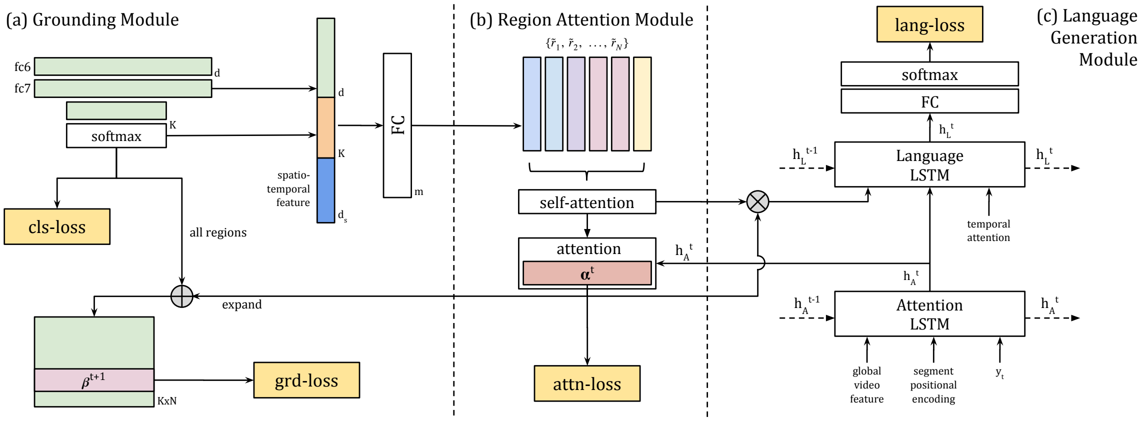


Figure 6.3: The proposed framework consists of three parts: the grounding module (a), the region attention module (b) and the language generation module (c). Region proposals are first represented with grounding-aware region encodings. The language model then dynamically attends on the region encodings to generate each word. Losses are imposed on the attention weights (attn-loss), grounding weights (grd-loss), and the region classification probabilities (cls-loss). For clarity, the details of the temporal attention are omitted.

6.2 Description with Grounding Supervision

In this section we describe the proposed grounded video description framework (see Fig. 6.3). The framework consists of three modules: grounding, region attention and language generation. The grounding module detects visual clues from the video, the region attention dynamically attends on the visual clues to form a high-level impression of the visual content and feeds it to the language generation module for decoding. We illustrate three options for incorporating the object-level supervision: region classification, object grounding (localization), and supervised attention.

6.2.1 Overview

We formulate the problem as a joint optimization over the language and grounding tasks. The overall loss function consists of four parts:

$$L = L_{sent} + \lambda_\alpha L_{attn} + \lambda_c L_{cls} + \lambda_\beta L_{grd}, \tag{6.1}$$

where L_{sent} denotes the teacher-forcing language generation cross-entropy loss, commonly used for language generation tasks (details in Sec. 6.2.2). L_{attn} corresponds to the cross entropy region attention loss which is presented in Sec. 6.2.3. L_{cls} and L_{grd} are cross-entropy losses that correspond to the grounding module for region classification and supervised object grounding (localization), respectively (Sec. 6.2.4). The three grounding-related losses are weighted by coefficients λ_α, λ_c, and λ_β which we selected on the dataset validation split.

We denote the input video (segment) as V and the target/generated sentence description (words) as S. We uniformly sample F frames from each video as $\{v_1, v_2, \dots, v_F\}$ and define N_f object regions on sampled frame f. Hence, we can assemble a set of regions $R = [R_1, \dots, R_F] = [r_1, r_2, \dots, r_N] \in \mathbb{R}^{d \times N}$ to represent the video, where $N = \sum_{f=1}^{F} N_f$ is the total number of regions. We overload the notation here and use r_i ($i \in \{1, 2, \dots, N\}$) to also represent region feature embeddings, as indicated by fc6 in Fig. 6.3. We represent words in S with one-hot vectors which are further encoded to word embeddings $y_t \in \mathbb{R}^e$ where $t \in \{1, 2, \dots, T\}$, where T indicates the sentence length and e is the embedding size.

6.2.2 Language Generation Module

For language generation, we adapt the language model from [74] for video inputs, *i.e.*, extend it to incorporate temporal information. The model consists of two LSTMs: the first one for encoding the global video feature and the word embedding y_t into the hidden state $h_A^t \in \mathbb{R}^m$ where m is the dimension and the second one for language generation (see Fig. 6.3c). The language model dynamically attends on videos frames or regions for visual clues to generate words. We refer to the attention on video frames as temporal attention and the one on regions as region attention.

The temporal attention takes in a sequence of frame-wise feature vectors and determines by the hidden state how significant each frame should contribute to generate a description word. We deploy a similar module as in [2], except that we replace the self-attention context encoder with Bi-directional GRU (Bi-GRU) which yields superior results. We train with cross-entropy loss L_{sent}.

6.2.3 Region Attention Module

Unlike temporal attention that works on a frame level, the region attention [73, 74] focuses on more fine-grained details in the video, *i.e.*, object regions [47]. We denote the region encoding as $\tilde{R} = [\tilde{r}_1, \tilde{r}_2, \dots, \tilde{r}_N]$, more details are defined later in Eq. 6.5. At time t of the caption generation, the attention weight over region i is formulated as:

$$\alpha_i^t = w_\alpha^\top \text{tanh}(W_r \tilde{r}_i + W_h h_A^t), \quad \alpha^t := \text{Softmax}(\alpha^t), \tag{6.2}$$

where $W_r \in \mathbb{R}^{m \times d}$, $W_h \in \mathbb{R}^{m \times m}$, $w_\alpha \in \mathbb{R}^m$, and $\alpha^t = [\alpha_1^t, \alpha_2^t, \dots, \alpha_N^t]$. The region attention encoding is then $\tilde{R}\alpha^t$ and along with the temporal attention encoding, fed into the language LSTM.

Supervised Attention. We want to encourage the language model to attend on the correct region when generating a visually-groundable word. As this effectively assists the language model in learning to attend to the correct region, we call this *attention supervision*. Denote the indicators of positive/negative regions as $\gamma^t = [\gamma_1^t, \gamma_2^t, \dots, \gamma_N^t]$, where $\gamma_i^t = 1$ when the region r_i has over 0.5 IoU with the GT box r_{GT} and otherwise 0. We regress α^t to γ^t and hence the attention loss for object word s_t can be defined as:

$$L_{attn} = -\sum_{i=1}^{N} \gamma_i^t \log \alpha_i^t. \tag{6.3}$$

6.2.4 Grounding Module

Assume we have a set of visually-groundable object class labels $\{c_1, c_2, \dots, c_\mathcal{K}\}$, short as object classes, where $\mathcal{K}$ is the total number of classes. Given a set of object regions from all sampled frames, the grounding module estimates the class probability distribution for each region.

We define a set of object classifiers as $W_c = [w_1, w_2, \dots, w_{\mathcal{K}}] \in \mathbb{R}^{d \times \mathcal{K}}$ and the learnable scalar biases as $B = [b_1, b_2, \dots, b_{\mathcal{K}}]$. So, a naive way to estimate the class probabilities for all regions (embeddings) $R = [r_1, r_2, \dots, r_N]$ is through dot-product:

$$M_s(R) = \text{Softmax}(W_c^\top R + B\mathbb{1}^\top), \tag{6.4}$$

where $\mathbb{1}$ is a vector with all ones, $W_c^\top R$ is followed by a ReLU and a Dropout layer, and M_s is the *region-class similarity matrix* as it captures the similarity between regions and object classes. For clarity, we omit the ReLU and Dropout layer after the linear embedding layer throughout Sec. 6.2 unless otherwise specified. The Softmax operator is applied along the object class dimension of M_s to ensure the class probabilities for each region sum up to 1.

We transfer detection knowledge from an off-the-shelf detector that is pre-trained on a general source dataset, *i.e.*, Visual Genome (VG) [147], to our object classifiers. We find the nearest neighbor for each of the $\mathcal{K}$ object classes from the VG object classes according to their distances in the embedding space (glove vectors [148]). We then initialize W_c and B with the corresponding classifier, *i.e.*, the weights and biases, from the last linear layer of the detector.

On the other hand, we represent the spatial and temporal configuration of the region as a 5-D tuple, including 4 values for the normalized spatial location and 1 value for the normalized frame index. Then, the 5-D feature is projected to a $d_s = 300$-D location embedding for all the regions $M_l \in \mathbb{R}^{300 \times N}$. Finally, we concatenate all three components: i) region feature, ii) region-class similarity matrix, and iii) location embedding together and project into a lower dimension space (m-D):

$$\tilde{R} = W_g[\ R \mid M_s(R) \mid M_l\ ], \tag{6.5}$$

where $[\cdot|\cdot]$ indicates a row-wise concatenation and $W_g \in \mathbb{R}^{m\times(d+K+d_s)}$ are the embedding weights. We name $\tilde{R}$ the *grounding-aware region encoding*, corresponding to the right portion of Fig. 6.3a. To further model the relations between regions, we deploy a self-attention layer over $\tilde{R}$ [82, 2]. The final region encoding is fed into the region attention module (see Fig. 6.3b).

So far the object classifier discriminates classes without the prior knowledge about the semantic context, *i.e.*, the information the language model has captured. To incorporate semantics, we condition the class probabilities on the sentence encoding from the Attention LSTM. A memory-efficient approach is treating attention weights α^t as this semantic prior, as formulated below:

$$M_s^t(R, \alpha^t) = \text{Softmax}(W_c^\top R + B\mathbb{1}^\top + \mathbb{1}{\alpha^t}^\top), \tag{6.6}$$

where the region attention weights α^t are determined by Eq. 6.2. Note that here the Softmax operator is applied row-wise to ensure the probabilities on regions sum up to 1. To learn a reasonable object classifier, we can deploy a region classification task on $M_s(R)$ or a sentence-conditioned grounding task on $M_s^t(R, \alpha^t)$, with the word-level grounding annotations from Sec. 3.2.1. Next, we describe them both.

Region Classification. We first define a positive region as a region that has over 0.5 intersection over union (IoU) with an arbitrary ground-truth (GT) box. If a region matches to multiple GT boxes, the one with the largest IoU is the final matched GT box. Then we classify the positive region, say region i to the same class label as in the GT box, say class c_j. The normalized class probability distribution is hence $M_s[:, i]$ and the cross-entropy loss on class c_j is

$$L_{cls} = -\log M_s[j, i]. \tag{6.7}$$

The final L_{cls} is the average of losses on all positive regions.

Object Grounding. Given a visually-groundable word s_{t+1} at time step $t + 1$ and the encoding of all the previous words, we aim to localize s_{t+1} in the video as one or a few of the region proposals. Supposing s_{t+1} corresponds to class c_j, we regress the confidence score of regions $M_s^t[j,:] = \beta^{t+1} = [\beta_1^{t+1}, \beta_2^{t+1}, \ldots, \beta_N^{t+1}]$ to indicators γ^t as defined in Sec. 6.2.3. The grounding loss for word s_{t+1} is defined as:

$$L_{grd} = -\sum_{i=1}^{N} \gamma_i^t \log \beta_i^{t+1}. \tag{6.8}$$

Note that the final loss on L_{attn} or L_{grd} is the average of losses on all visually-groundable words. The difference between the attention supervision and the grounding supervision is that, in the latter task, the target object c_j is known beforehand, while the attention module is not aware of which object to seek in the scene.

6.3 Experiments

Datasets. We conduct most experiments and ablation studies on the newly-collected ActivityNet-Entities dataset on video description given the set of temporal segments (*i.e.*, using the ground-truth events from [1]) and video paragraph description [149]. We also demonstrate our framework can easily be applied to image description and evaluate it on the Flickr30k Entities dataset [89]. Note that we did not apply our method to COCO captioning as there is no exact match between words in COCO captions and object annotations in COCO (limited to only 80). We use the same process described in Sec. 3.2.3 to convert NPs to object labels. Since Flickr30k Entities contains more captions, labels that occur at least 100 times are taken as object labels, resulting in 480 object classes [74].

Pre-processing. For ANet-Entities, we truncate captions longer than 20 words and build a vocabulary on words with at least 3 occurrences. For Flickr30k Entities, since the captions are generally shorter and it is a larger corpus, we truncate captions longer than 16 words and build a vocabulary based on words that occur at least 5 times.

Compared Methods The state-of-the-art (SotA) video description methods on ActivityNet

Captions include Masked Transformer and Bi-LSTM+TempoAttn [2]. We re-train the models on our dataset splits with the original settings. For a fair comparison, we use exactly the same frame-wise feature from this work for our temporal attention module. For video paragraph description, we compare our methods against the SotA method MFT [149] with the evaluation script provided by the authors [149]. For image captioning, we compare against two SotA methods, Neural Baby Talk (NBT) [74] and BUTD [73]. For a fair comparison, we provide the same region proposal and features for both the baseline BUTD and our method, *i.e.*, from Faster R-CNN pre-trained on Visual Genome (VG). NBT is specially tailored for each dataset (*e.g.*, detector fine-tuning), so we retain the same feature as in the work, *i.e.*, from ResNet pre-trained on ImageNet. All our experiments are performed three times and the average scores are reported.

6.3.1 Evaluation Metrics

Localization Metrics. To measure the object grounding and attention correctness, we first compute the localization accuracy (*Grd.* and *Attn.* in the tables) over GT sentences following [94, 25]. Given an unseen video, we feed the GT sentence into the model and measure the localization accuracy at each annotated object word. We compare the region with the highest attention weight (α_i) or grounding weight (β_j) against the GT box. An object word is correctly localized if the IoU is over 0.5. We also study the attention accuracy on generated sentences, *i.e.*, given a test video segment, we perform the standard language generation inference and compute attention localization accuracy (no grounding measurement here because it is usually evaluated on GT sentences). The metrics are denoted by $F1_{all}$ and $F1_{loc}$ in the tables. In $F1_{all}$, a region prediction is considered correct if the object word is correctly predicted and also correctly localized. We also compute $F1_{loc}$, which only considers correctly-predicted object words. We explain $F1_{all}$ and $F1_{loc}$ in details as follows.

We define the number of object words in the generated sentences as A, the number of object words in the GT sentences as B, the number of correctly predicted object words in the generated sentences as C and the counterpart in the GT sentences as D, and the number of correctly predicted and localized words as E. A region prediction is considered correct if the object word is correctly predicted and also correctly localized (*i.e.*, IoU with GT box $>$ 0.5).

In *F1*$_{all}$, the precision and recall can be defined as:

$$\text{Precision}_{all} = \frac{E}{A}, \quad \text{Recall}_{all} = \frac{E}{B} \tag{6.9}$$

However, since having box annotation for every single object in the scene is unlikely, an incorrectly-predicted word might not necessarily be a hallucinated object. Hence, we also compute *F1*$_{loc}$, which only considers correctly-predicted object words, *i.e.*, only measures the localization quality and ignores errors result from the language generation. The precision and recall for *F1*$_{loc}$ are defined as:

$$\text{Precision}_{loc} = \frac{E}{C}, \quad \text{Recall}_{loc} = \frac{E}{D} \tag{6.10}$$

If multiple instances of the same object exist in the target sentence, we only consider the first instance. The precision and recall for the two metrics are computed for each object class, but it is set to zero if an object class has never been predicted. Finally, we average the scores by dividing by the total number of object classes in a particular split (val or test). To address the sparsity of the annotation, we twist our evaluation protocol in the following way. During model training, we restrict the grounding region candidates within the target frame (w/ GT box), *i.e.*, only consider the N_f proposals on the frame f with the GT box.

Other Metrics. For the region classification task, we compute the top-1 classification accuracy (*Cls.* in the tables) for positive regions. For all metrics, we average the scores across object classes. To evaluate the sentence quality, we use standard language evaluation metrics, including Bleu@1, Bleu@4, METEOR, CIDEr, and SPICE, and the official evaluation script.[2] We additionally perform human evaluation to judge the sentence quality.

6.3.2 Implementation Details

Region proposal and feature. We uniformly sample 10 frames per video segment (an event in ANet-Entities) and extract region features. For each frame, we use a Faster R-CNN detector [47] with ResNeXt-101 backbone [150] for region proposal and feature extraction (fc6). The detector is pretrained on Visual Genome [147]. More model and training details are in the Appendix B.

Feature map and attention. The temporal feature map is essentially a stack of frame-wise appearance and motion features from [2, 137]. The spatial feature map is the conv4 layer output from a ResNet-101 [74, 123] model. Note that an average pooling on the temporal or spatial feature map gives the global feature. In video description, we augment the global feature with segment positional information (*i.e.*, total number of segments, segment index, start time and end time), which is empirically important.

Hyper-parameters. Coefficients $\lambda_\alpha \in \{0.05, 0.1, 0.5\}$, $\lambda_\beta \in \{0.05, 0.1, 0.5\}$, and $\lambda_c \in \{0.1, 0.5, 1\}$ vary in the experiments as a result of model validation. We set $\lambda_\alpha = \lambda_\beta$ when they are both non-zero considering the two losses have a similar functionality. The region encoding size $d = 2048$, word embedding size $e = 512$ and RNN encoding size $m = 1024$ for all methods. Other hyper-parameters in the language module are the same as in [74]. We use a 2-layer 6-head Transformer encoder as the self-attention module [2].

[2] https://github.com/ranjaykrishna/densevid_eval

Method	λ_α	λ_β	λ_c	B@1	B@4	M	C	S	Attn.	Grd.	F1$_{all}$	F1$_{loc}$	Cls.
Unsup. (w/o SelfAttn)	0	0	0	23.2	2.28	10.9	45.6	**15.0**	14.9	21.3	3.70	12.7	6.89
Unsup.	0	0	0	23.0	2.27	10.7	44.6	13.8	2.42	19.7	0.28	1.13	6.06
Sup. Attn.	0.05	0	0	23.7	2.56	**11.1**	47.0	14.9	34.0	37.5	6.72	22.7	0.42
Sup. Grd.	0	0.5	0	23.5	2.50	11.0	46.8	14.7	31.9	43.2	6.04	21.2	0.07
Sup. Cls.	0	0	0.1	23.3	2.43	10.9	45.7	14.1	2.59	25.8	0.35	1.43	**14.9**
Sup. Attn.+Grd.	0.5	0.5	0	**23.8**	2.44	**11.1**	46.1	14.8	**35.1**	40.6	6.79	23.0	0
Sup. Attn.+Cls.	0.05	0	0.1	**23.9**	**2.59**	**11.2**	**47.5**	**15.1**	34.5	41.6	**7.11**	**24.1**	**14.2**
Sup. Grd. +Cls.	0	0.05	0.1	**23.8**	**2.59**	**11.1**	**47.5**	**15.0**	27.1	**45.7**	4.79	17.6	13.8
Sup. Attn.+Grd.+Cls.	0.1	0.1	0.1	**23.8**	2.57	**11.1**	46.9	**15.0**	**35.7**	**44.9**	**7.10**	**23.8**	12.2

Table 6.1: Results on ANet-Entities val set. "w/o SelfAttn" indicates self-attention is not used for region feature encoding. Notations: B@1 - Bleu@1, B@4 - Bleu@4, M - METEOR, C - CIDEr, S - SPICE. Attn. and Grd. are the object localization accuracies for attention and grounding on GT sentences. F1$_{all}$ and F1$_{loc}$ are the object localization accuracies for attention on generated sentences. Cls. is classification accuracy. All accuracies are in %. Top two scores on each metric are in bold.

Method	B@1	B@4	M	C	S	Attn.	Grd.	F1$_{all}$	F1$_{loc}$	Cls.
Masked Transformer [2]	22.9	**2.41**	10.6	**46.1**	13.7	–	–	–	–	–
Bi-LSTM+TempoAttn [2]	22.8	2.17	10.2	42.2	11.8	–	–	–	–	–
Our Unsup. (w/o SelfAttn)	23.1	2.16	10.8	44.9	**14.9**	16.1	22.3	3.73	11.7	6.41
Our Sup. Attn.+Cls. (GVD)	**23.6**	2.35	**11.0**	45.5	14.7	**34.7**	**43.5**	**7.59**	**25.0**	**14.5**

Table 6.2: Results on ANet-Entities test set. The top one score for each metric is in bold. On language evaluation, the improvements of our GVD model over the SotA method Masked Transformer on metrics METEOR and SPICE are statistically significant (p-value<0.02).

6.3.3 Results on ActivityNet-Entities

6.3.3.1 Video Event Description

Although dense video description [147] further entails localizing the segments to describe on the temporal axis, in this work we focus on the language generation part and assume the temporal boundaries for events are given. We name this task Video Event Description. Results on the validation and test splits of our ActivityNet-Entities dataset are shown in Tab. 6.1 and Tab. 6.2, respectively. Given the selected set of region proposals, the localization upper bound on the val/test sets is 82.5%/83.4%, respectively.

In general, methods with some form of grounding supervision work consistently better than the methods without. Moreover, combining multiple losses, *i.e.*, stronger supervision, leads to higher performance. On the val set, the best variant of supervised methods (*i.e.*, Sup. Attn.+Cls.) ourperforms the best variant of unsupervised methods (*i.e.*, Unsup. (w/o SelfAttn)) by a relative 1-13% on all the metrics. On the test set, the gaps are small for Bleu@1, METEOR, CIDEr, and SPICE (within ± 2%), but the supervised method has a 8.8% relative improvement on Bleu@4.

The results in Tab. 6.2 show that adding box supervision dramatically improves the grounding accuracy from 22.3% to 43.5%. Hence, our supervised models can better localize the objects mentioned which can be seen as an improvement in their ability to explain or justify their own description. The attention accuracy also improves greatly on both GT and generated sentences, implying that the supervised models learn to attend on more relevant objects during language generation. However, grounding loss alone fails with respect to classification accuracy (see Tab. 6.1), and therefore the classification loss is required in that case. Conversely, the classification loss alone can implicitly learn grounding and maintains a fair grounding accuracy.

Temporal attention & region attention. We conduct ablation studies on the two attention modules to study the impact of each component on the overall performance (see Tab. 6.3). Each module alone performs similarly and the combination of two performs the best, which indicates the two attention modules are complementary. We hypothesize that the temporal attention captures the coarse-level details while the region attention captures more fine-grained details. Note that the region attention module takes in a lower sampling rate input than the temporal attention module, so we expect it can be further improved if having a higher sampling rate and the context (other events in the video). We leave this for future studies.

Comparison to existing methods. We refer to our best model (Sup. Attn.+Cls.) as GVD (Grounded Visual Description) and show that it sets the new SotA on ActivityNet Captions for the Bleu@1, METEOR and SPICE metrics, with relative gains of 2.8%, 3.9% and 6.8%, respectively over the previous best [2]. We observe slightly inferior results on Bleu@4 and

Method	B@1	B@4	M	C	S
Region Attn.	23.2	2.55	10.9	43.5	14.5
Tempo. Attn.	23.5	2.45	11.0	44.3	14.0
Both	**23.9**	**2.59**	**11.2**	**47.5**	**15.1**

Table 6.3: Ablation study for two attention modules using our best model. Results reported on val set.

	vs. Unsupervised		vs. [2]	
	Judgments		Judgments	
Method	%	Δ	%	Δ
About Equal	34.9		38.9	
Other is better	29.3	6.5	27.5	6.1
GVD is better	**35.8**		**33.6**	

Table 6.4: Human evaluation of sentence quality. We present results for our supervised approach vs. our unsupervised baseline and vs. Masked Transformer [2].

CIDEr (-2.8% and -1.4%, respectively) but after examining the generated sentences (see qualitative examples below) we see that [2] generates repeated words way more often. This may increase the aforementioned evaluation metrics, but the generated descriptions are of lower quality. Another noteworthy observation is that the self-attention context encoder (on top of $\tilde{R}$) brings consistent improvements on methods with grounding supervision, but hurts the performance of methods without, *i.e.*, "Unsup.". We hypothesize that the extra context and region interaction introduced by the self-attention confuses the region attention module and without any grounding supervision makes it fail to properly attend to the right region, something that leads to a huge attention accuracy drop from 14.9% to 2.42%.

Human Evaluation. Automatic metrics for evaluating generated sentences have frequently shown to be unreliable and not consistent with human judgments, especially for video description when there is only a single reference [145]. Hence, we conducted a human evaluation to evaluate the sentence quality on the test set of ActivityNet-Entities. We randomly sampled 329 video segments and presented the segments and descriptions to the judges. From Tab. 6.4, we observe that, while they frequently produce captions with similar qual-

Method	B@1	B@4	M	C
MFT [149]	45.5	9.78	14.6	20.4
Our Unsup. (w/o SelfAttn)	49.8	10.5	15.6	21.6
Our GVD	**49.9**	**10.7**	**16.1**	**22.2**

Table 6.5: Results of video paragraph description on test set.

ity, our GVD works better than the unsupervised baseline (with a significant gap of 6.1%). We can also see that our GVD approach works better than the Masked Transformer [2] with a significant gap of 6.5%. We believe these results are a strong indication that our approach is not only better grounded but also generates better sentences, both compared to baselines and prior work [2] (see our qualitative results below).

Qualitative examples. See Figs. 6.4 and 6.5 at the end of this chapter for qualitative results of our methods and the Masked Transformer on ANet-Entities val set. We visualize the proposal with the highest attention weight in the corresponding frame. In (a), the supervised model correctly attends to "man" and "Christmas tree" in the video when generating the corresponding words. The unsupervised model mistakenly predicts "Two boys". In (b), both "man" and "woman" are correctly grounded. In (c), both "man" and "saxophone" are correctly grounded by our supervised model while Masked Transformer hallucinates a "bed". In (d), all the object words (*i.e.*, "people", "beach", "horses") are correctly localized. The caption generated by Masked Transformer is incomplete. In (e), surprisingly, not only major objects "woman" and "court" are localized, but also the small object "ball" is attended with a high precision. Masked Transformer incorrectly predicts the gender of the person. In (f), the Masked Transformer outputs an unnatural caption "A group of people are in a raft and a man in red raft raft raft raft raft" containing consecutive repeated words "raft".

6.3.3.2 Video Paragraph Description

Besides measuring the quality of each individual description, we also evaluate the coherence among sentences within a video as in [149]. We obtained the result file and evaluation script from [149][3] and evaluated both methods on *our* test split. The results are shown in Tab. 6.5. We outperform the SotA method [149] by a large margin, with relative improvements of 8.9-10% on all the metrics. The results are even more surprising given that we generate description for each event separately, without conditioning on previously-generated sentences. We hypothesize that the temporal attention module can effectively model the event context through the Bi-GRU context encoder and context benefits the coherence of consecutive sentences.

6.3.4 Results on Flickr30k Entities

We show the overall results on image description in Tab. 6.6 (val) and Tab. 6.7 (test). The upper bounds on the val/test sets are 90.0%/88.5%, respectively. On the val set, we perform a light hyper-parameter search on supervised methods and notice the setting $\lambda_\alpha = 0.1$, $\lambda_\beta = 0.1$ and $\lambda_c = 1$ generally works well. The supervised methods outperform the unsupervised baseline by a decent amount in all the metrics with only one exceptions: Sup. Cls., which has a slightly inferior result in CIDEr. The best supervised method (Sup. Attn.+Grd.+Cls., denoted also as GVD) outperforms the best unsupervised baseline by a relative 0.9-4.8% over all the metrics. On the test set, we see that the supervised method outperforms the unsupervised baseline by a relative 1-3.7% over all the metrics. Our GVD model sets new SotA for all the five metrics with relative gains up to 10%. In the meantime, object localization and region classification accuracies are significantly boosted, showing that our captions can be better visually explained and understood.

[3] The authors kindly provided us with their result file and evaluation script, but as they were unable to provide us with their splits, we evaluated both methods on *our* test split. Even though we are under an unfair disadvantage, *i.e.*, the authors' val split might contain videos from our test split, we still outperform the SotA method by a large margin.

Method	λ_α	λ_β	λ_c	B@1	B@4	M	C	S	Attn.	Grd.	F1$_{all}$	F1$_{loc}$	Cls.
Unsup. (w/o SelfAttn)	0	0	0	70.0	27.5	22.0	60.4	15.9	22.0	25.9	4.44	12.8	17.6
Unsup.	0	0	0	69.3	26.8	22.1	59.4	15.7	4.04	16.3	0.80	2.09	1.35
Sup. Attn.	0.1	0	0	**71.0**	**28.2**	**22.7**	63.0	**16.3**	**42.3**	44.1	8.08	22.4	6.59
Sup. Grd.	0	0.1	0	70.1	27.6	22.5	**63.1**	16.1	38.5	49.5	7.59	21.0	0.03
Sup. Cls. (w/o SelfAttn)	0	0	1	70.1	27.6	22.0	60.2	15.8	20.9	32.1	4.12	11.5	**19.9**
Sup. Attn.+Grd.	0.1	0.1	0	70.2	27.6	22.5	62.3	**16.3**	**42.7**	49.8	**8.62**	**23.6**	0
Sup. Attn.+Cls.	0.1	0	1	70.0	27.9	22.6	62.4	**16.3**	42.1	46.5	**8.35**	**23.2**	**19.9**
Sup. Grd. +Cls.	0	0.1	1	70.4	28.0	**22.7**	62.8	**16.3**	29.0	**51.2**	5.19	13.7	19.7
Sup. Attn.+Grd.+Cls.	0.1	0.1	1	**70.6**	**28.1**	22.6	**63.3**	**16.3**	41.2	**50.8**	8.30	**23.2**	19.6

Table 6.6: Results on Flickr30k Entities val set. The top two scores on each metric are in bold.

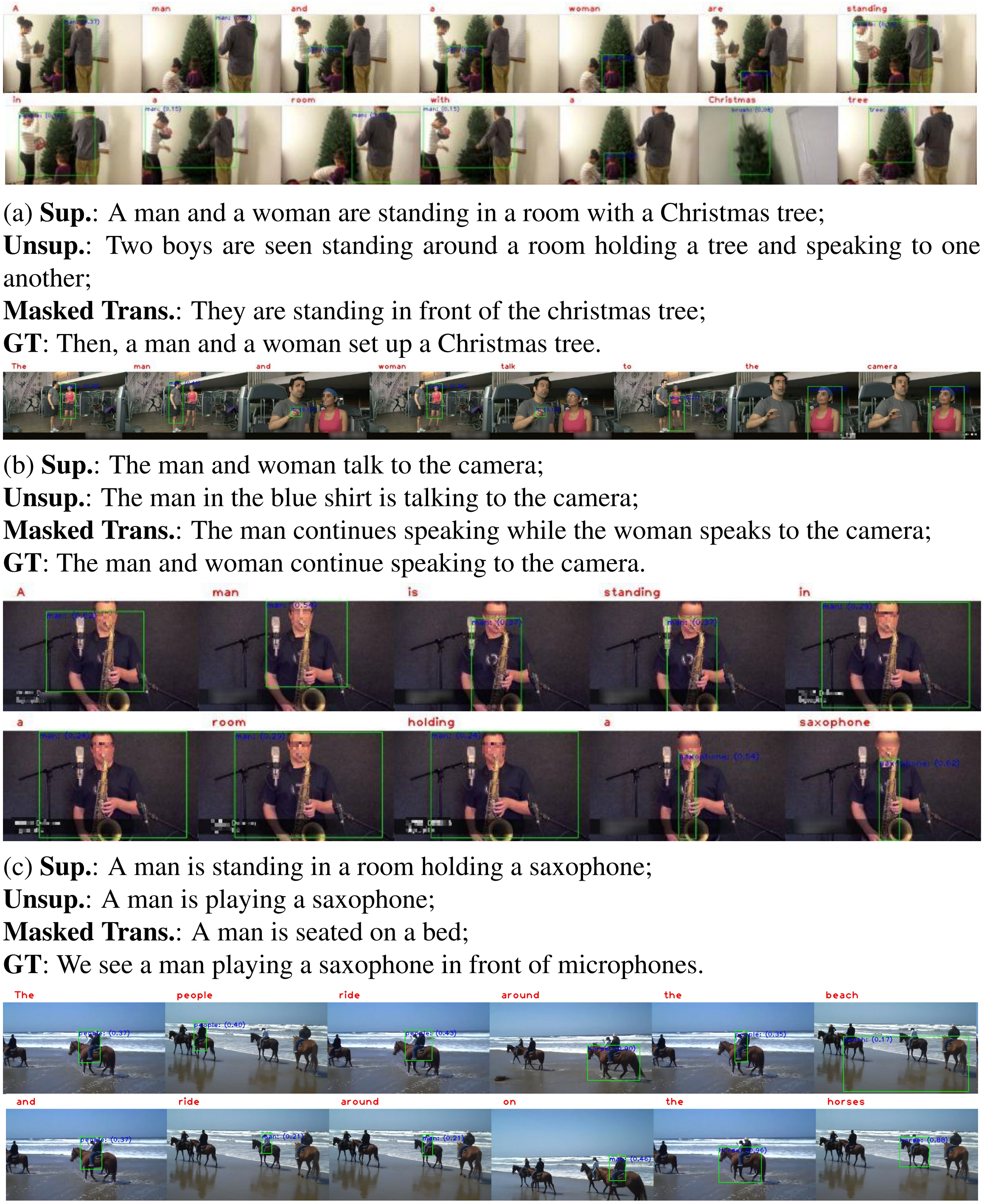
Method	VG	Box	B@1	B@4	M	C	S	Attn.	Grd.	F1$_{all}$	F1$_{loc}$	Cls.
ATT-FCN* [67]			64.7	19.9	18.5	–	–	–	–	–	–	–
NBT* [74]		✓	69.0	27.1	21.7	57.5	15.6	–	–	–	–	–
BUTD [73]	✓		69.4	**27.3**	21.7	56.6	16.0	24.2	32.3	4.53	13.0	1.84
Our Unsup. (w/o SelfAttn)	✓		69.2	26.9	22.1	60.1	16.1	21.4	25.5	3.88	11.7	17.9
Our GVD model	✓	✓	**69.9**	**27.3**	**22.5**	**62.3**	**16.5**	**41.4**	**50.9**	**7.55**	**22.2**	**19.2**

Table 6.7: Results on Flickr30k Entities test set. * indicates the results are obtained from the original papers. GVD refers to our Sup. Attn.+Grd.+Cls. model. "VG" indicates region features are from VG pre-training. The top one score is in bold. On language evaluation, the improvements of our GVD model over the SotA method BUTD on metrics METEOR and CIDEr are statistically significant (p-value$<$0.02).

Qualitative examples. See Fig. 6.6 for the qualitative results by our methods and the BUTD on Flickr30k Entities val set. We visualize the proposal with the highest attention weight as the green box. The corresponding attention weight and the most confident object prediction of the proposal are displayed as the blue text inside the green box. In (a), the supervised model correctly attends to "man", "dog" and "snow" in the image when generating the corresponding words. The unsupervised model misses the word "snow" and BUTD misses the word "man". In (b), the supervised model successfully incorporates the detected visual clues (*i.e.*, "women", "building") into the description. We also show a negative example in (c), where interestingly, the back of the chair looks like a laptop, which confuses our grounding module. The supervised model hallucinates a "laptop" in the scene.

6.4 Discussion

In this work, we collected ActivityNet-Entities, a novel dataset that allows joint study of video description and grounding. We show how to leverage the noun phrase annotations to generate grounded video descriptions. We also use our dataset to evaluate how well the generated sentences are grounded. We believe our large-scale annotations will also allow for more in-depth analysis which have previously only been able on images, *e.g.*about hallucination [143] and bias [142] as well as studying co-reference resolution. Besides, we showed in our comprehensive experiments on video and image description, how the box supervision can improve the accuracy and the explainability of the generated description by not only generating sentences but also pointing to the corresponding regions in the video frames or image. According to automatic metrics and human evaluation, on ActivityNet-Entities our model performs state-of-the-art description quality, both when evaluated per sentence or on paragraph level with a significant increase in grounding performance. We also adapted our model to image description and evaluated it on the Flickr30k Entities dataset where our model outperforms existing methods, both description quality and grounding accuracy.



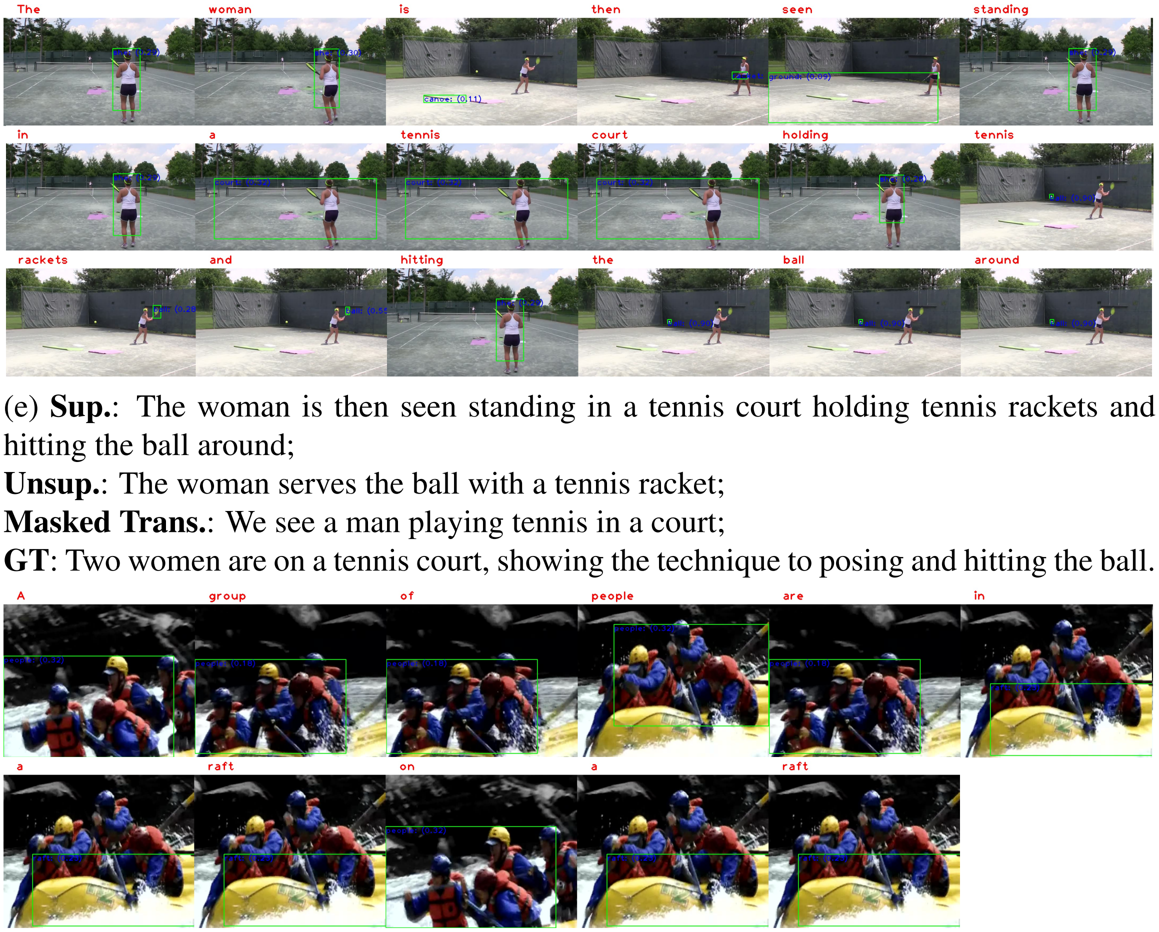
(a) **Sup.**: A man and a woman are standing in a room with a Christmas tree;
Unsup.: Two boys are seen standing around a room holding a tree and speaking to one another;
Masked Trans.: They are standing in front of the christmas tree;
GT: Then, a man and a woman set up a Christmas tree.

(b) **Sup.**: The man and woman talk to the camera;
Unsup.: The man in the blue shirt is talking to the camera;
Masked Trans.: The man continues speaking while the woman speaks to the camera;
GT: The man and woman continue speaking to the camera.

(c) **Sup.**: A man is standing in a room holding a saxophone;
Unsup.: A man is playing a saxophone;
Masked Trans.: A man is seated on a bed;
GT: We see a man playing a saxophone in front of microphones.

(d) **Sup.**: The people ride around the beach and ride around on the horses;
Unsup.: The people ride around the beach and ride around;
Masked Trans.: The camera pans around the area and the girl leading the horse and the woman leading the;
GT: We see four people on horses on the beach.

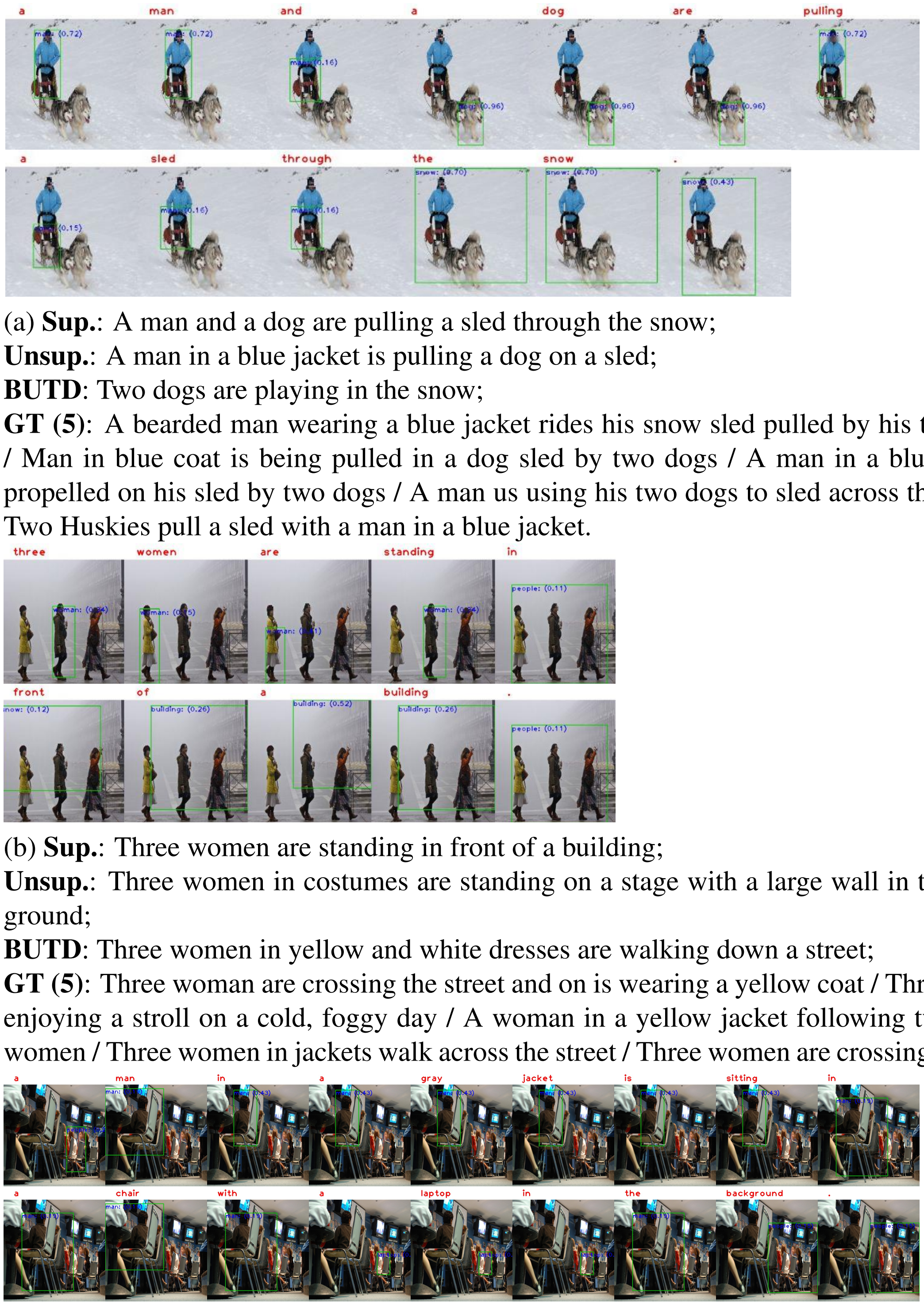
Figure 6.4: Qualitative results on ANet-Entities val set. The red text at each frame indicates the generated word. The green box indicates the proposal with the highest attention weight. The blue text inside the green box corresponds to i) the object class with the highest probability and ii) the attention weight. Better zoomed and viewed in color.



(e) **Sup.**: The woman is then seen standing in a tennis court holding tennis rackets and hitting the ball around;
Unsup.: The woman serves the ball with a tennis racket;
Masked Trans.: We see a man playing tennis in a court;
GT: Two women are on a tennis court, showing the technique to posing and hitting the ball.

(f) **Sup.**: A group of people are in a raft on a raft;
Unsup.: A group of people are in a raft;
Masked Trans.: A group of people are in a raft and a man in red raft raft raft raft raft;
GT: People are going down a river in a raft.

Figure 6.5: (Continued) Qualitative results on ANet-Entities val set. See the caption in Fig. 6.4 for more details.



(a) **Sup.**: A man and a dog are pulling a sled through the snow;
Unsup.: A man in a blue jacket is pulling a dog on a sled;
BUTD: Two dogs are playing in the snow;
GT (5): A bearded man wearing a blue jacket rides his snow sled pulled by his two dogs / Man in blue coat is being pulled in a dog sled by two dogs / A man in a blue coat is propelled on his sled by two dogs / A man us using his two dogs to sled across the snow / Two Huskies pull a sled with a man in a blue jacket.

(b) **Sup.**: Three women are standing in front of a building;
Unsup.: Three women in costumes are standing on a stage with a large wall in the background;
BUTD: Three women in yellow and white dresses are walking down a street;
GT (5): Three woman are crossing the street and on is wearing a yellow coat / Three ladies enjoying a stroll on a cold, foggy day / A woman in a yellow jacket following two other women / Three women in jackets walk across the street / Three women are crossing a street.

(c) **Sup.**: A man in a gray jacket is sitting in a chair with a laptop in the background;
Unsup.: A man in a brown jacket is sitting in a chair at a table;
BUTD: A man in a brown jacket is sitting in a chair with a woman in a brown jacket in a;
GT (5): Several chairs lined against a wall, with children sitting in them / A group of children sitting in chairs with monitors over them / Children are sitting in chairs under some television sets / Pre-teen students attend a computer class / Kids conversing and learning in class.

Figure 6.6: Qualitative results on Flickr30k Entities val set. Better zoomed and viewed in color. See Sec. B for discussion.

CHAPTER VII

Weakly-Supervised Object Grounding

7.1 Introduction

Like most fine-grained recognition problems [47, 89], grounding can be extremely data intensive, especially in the context of untrimmed videos (*e.g.*, our work [26] from the last chapter). On the other hand, video-sentence pairs are easier to obtain than object region annotations (*e.g.*, YouTube subtitles or ASR scripts). We focus on the weakly-supervised version of the grounding problem where the only supervision is sentence descriptions; no spatially-aligned object bounding boxes are available for training. Sentence grounding can involve multiple interacting objects, which sets our work apart from the relatively well-studied weakly-supervised object localization problem, where one or more objects are localized independently [151, 152].

Existing work on visual grounding falls into two categories: multiple instance learning [97, 98] and visual attention [94]. In either case, the visual-semantic similarity is first measured between the target object/phrase and all the image-level, *i.e.*, spatial object region proposals. Then, either a ranking loss or a reconstruction loss—both of which we refer to here as matching losses—measures the quality of the matching. A naive extension of the existing approaches to the video domain is to treat the entire video segment as a bag of spatial object proposals. However, this presents two issues. First, existing methods rely on the assumption that the target object appears in *at least one* of the proposal regions. This assumption is weak when it comes to video, since a query object might appear sparsely across multiple frames[1] and might not be detected completely. The *segment-level supervision*, *i.e.*, object labels, could be potentially strengthened if applied to individual frames. Second, a video segment can last up to several minutes. Even with temporal down-sampling, this can bring in tens or hundreds of frames and hence thousands of proposals, which compromise the visual-semantic alignment accuracy.

To address these two issues, we propose a frame-wise loss weighting framework for video grounding. We ground the target objects on a frame-by-frame basis. We face the challenge that the segment-level supervision is not applicable to individual frames where the query object is off-screen, occluded, or just not present in the proposals for that frame. Our solution is to first estimate the likelihood that the query object is present in (a proposal in) each video frame. If the likelihood is high, we judge the matching quality mainly on the matching loss. Otherwise, we down-weight the matching loss while bringing in a penalty loss. The lower the confidence, the higher the penalty. With the conditioned frame-wise grounding framework, the proposed model can avoid being flooded with massive proposals even when the sampling rate is high and only make predictions for applicable frames.

We propose two approaches to estimate frame-wise object likelihood (confidence) scores. The first one is conditioned on both visual and textual inputs, namely, the maximum visual-semantic similarity scores in each frame. The second approach is inspired by the fact that the combination of objects can imply their order of appearance in the video. For example, when a sequence of objects “tomatoes”, “pan” and “plate” appears in the description, the video scene is likely to include a shot of tomatoes being grilled in the pan at the beginning, and a shot of tomatoes being moved to the plate at the end. In the temporal domain, “pan” appears mostly ahead of “plate” while “tomatoes” intersects with both. We implicitly model the object interaction with self-attention [82] and use textual guidance to estimate the frame-wise object likelihood.

[1] In YouCook2-BoundingBox, the target object appears in 60.7% of the total frames, on average.

For evaluation, due to lack of existing video grounding benchmarks, we have collected annotations over the large-scale instructional video dataset YouCook2, which provides over 15,000 video segment-description pairs (see Sec. 3.1). We sample the validation and testing videos at 1fps and draw bounding box for the 67 most frequent objects when they are present in both the video segment and the description. We compare our methods against competitive baselines on video grounding and our proposed methods achieve state-of-the-art performances.

7.2 Preliminary

In this section we provide background on visual-semantic alignment framework (grounding by ranking), which is the backbone of our model. The background on Transformer is included in Appendix A.

Grounding by Ranking. We start by describing ranking-based grounding approach from [97]. Given a sentence description including O query objects/phrases and a set of N object region proposals from an image, the goal is to target each referred object in the query as one of the object proposals. Queries and visual region proposals are first encoded in a common d-dimensional space. Denote the object query feature vectors as $\{q_k\}$, $k = 1, 2, \ldots, O$ and the region proposal feature vectors as $\{r_i\}$, $i = 1, 2, \ldots, N$. We pack the feature vectors into matrices $Q = (q_1, \ldots, q_O)$ and $R = (r_1, \ldots, r_N)$. The visual-semantic matching score of the description and the image is formulated as:

$$S(Q, R) = \frac{1}{O} \sum_{k=1}^{O} \max_{i} a_k^i, \tag{7.1}$$

where $a_k^i = q_k^\top r_i$ measures the similarity between query q_k and proposal r_i. Defining negative samples Q' and R' as the query and proposal from texts and images that are not paired with R nor Q, the grounding by ranking framework minimizes the following margin loss:

$$L_{rank} = \sum_{R' \neq R} \sum_{Q' \neq Q} [\max(0, S(Q,R') - S(Q,R) + \Delta) + \max(0, S(Q',R) - S(Q,R) + \Delta)], \quad (7.2)$$

where the first ranking term encourages the correct region proposal matching and the second ranking term encourages the correct sentence matching. Δ is the ranking margin. During inference, the proposal with the maximal similarity score a_k^i with each object query is selected.

7.3 Methods

In Sec. 7.3.1, we describe the video object grounding baseline. We then propose our framework in Sec. 7.3.2 by extending the segment-level object label supervision to the frame-level. Two novel approaches are proposed in judging under what circumstances the frame-level supervision is applicable.

7.3.1 Video Object Grounding

We adapt the Grounding by Ranking framework [97] to the video domain, and this adaptation will serve as our baseline. Denote the set of T frames in a video segment as $\{f_t\}$ and the object proposals in frame t as r_i^t, $i = 1, 2, \ldots, N$. As before, define the object queries as q_k, we compute the similarity between the query object and all the proposals $\{r_i^t\}$ in a segment. Note that the similarity dot product might grow large in magnitude as d increases [82]. Hence, we scale the dot-product by $\frac{1}{\sqrt{d}}$ and restrict $a_k^{t,i}$ to be between 0 and 1 with a Sigmoid function. The similarity function and segment-description matching score are then:

$$a_k^{t,i} = \text{Sigmoid}(q_k^\top r_i^t / \sqrt{d}), \quad S(Q,R) = \frac{1}{O} \sum_{k=1}^{O} \max_{t,i} a_k^{t,i}, \quad (7.3)$$

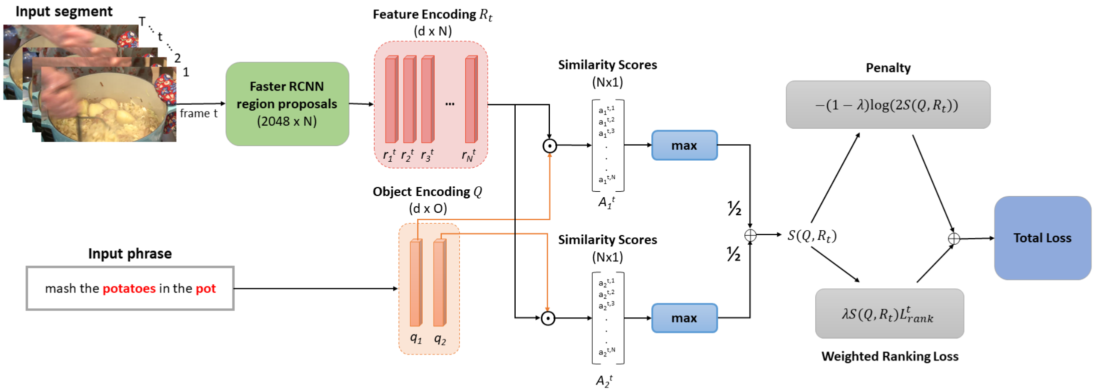

Input segment
T
t
2
1
frame t
Faster RCNN region proposals (2048 x N)
Feature Encoding R_t
(d x N)
r_1^t r_2^t r_3^t r_N^t
Object Encoding Q
(d x O)
q_1 q_2
Input phrase
mash the potatoes in the pot
Similarity Scores
(Nx1)
$a_1^{t,1}$ $a_1^{t,2}$ $a_1^{t,3}$ $a_1^{t,N}$
A_1^t
max
Similarity Scores
(Nx1)
$a_2^{t,1}$ $a_2^{t,2}$ $a_2^{t,3}$ $a_2^{t,N}$
A_2^t
max
½
½
$S(Q, R_t)$
Penalty
$-(1-\lambda)\log(2S(Q, R_t))$
$\lambda S(Q, R_t) L_{rank}^t$
Weighted Ranking Loss
Total Loss

Figure 7.1: An overview of our framework. Inputs to the system are a video segment and a phrase that describes the segment. The objects from the phrase are grounded for each sampled frame t. Object and proposal features are encoded to size d and visual-semantic similarity scores are computed. The ranking loss is weighted by a confidence score which combined with the penalty form the final loss. The object relations are further encoded to guide the loss weights (see Sec. 7.3.3 for details). During inference, the region proposal with the maximum similarity score with the object query is selected for grounding.

where matrix $R = (r_1^1, \ldots, r_N^1, r_1^2, \ldots, r_N^T)$ indicates the pack of all proposal features.

This "brute-force" extension of Grounding by Ranking framework to the video domain presents two issues. First, depending on the video sampling rate, the total number of proposals per segment ($T \times N$) could be extremely large. Hence this solution does not scale well to long frame sequences. Second, an object existing sparsely across multiple frames might not be detected completely since successfully spotting it from one single frame would trigger a satisfactory match. We explain next how we propagate this weak supervisory signal from the segment level to frames that likely contain the target object.

7.3.2 Frame-wise Loss Weighting

In our framework, each frame is considered separately to ground the same target objects. Fig. 7.1 shows an overview of our model. We first estimate the likelihood that the query object is present in each video frame. If the likelihood is high, we judge the matching quality mainly on the matching loss (e.g., ranking loss). Otherwise, we down-weight the matching loss while bringing in a penalty loss. The lower the confidence, the higher the penalty. For clarity, we explain our idea when the matching loss is the ranking loss L_{rank} but note that this can be generalized to other loss functions.

Let the ranking loss for frame t be L^t_{rank} and the similarity score between query k and proposal i be $a^{t,i}_k$. Let $Q = (q_1, \ldots, q_O)$ and $R_t = (r^t_1, \ldots, r^t_N)$. We define the *confidence score* of the prediction at frame t as the visual-semantic matching score:

$$C_t = \frac{1}{O}\sum_{k=1}^{O}\max_i(a^{t,i}_k) \equiv S(Q, R_t), \tag{7.4}$$

where $S(\cdot,\cdot)$ is defined in Eq. 7.1. The corresponding *penalty* is:

$$D_t = -\log(2C_t) = -\log[\frac{2}{O}\sum_{k=1}^{O}\max_i(a^{t,i}_k)], \tag{7.5}$$

inspired by [153]. The final loss for the segment is a weighted sum of frame-wise ranking losses and penalties:

$$L = \frac{1}{T}\sum_{t=1}^{T}[\lambda C_t L^t_{rank} + (1-\lambda)D_t], \tag{7.6}$$

$$L^t_{rank} = \sum_{R'_t \neq R_t}\sum_{Q' \neq Q}[\max(0, S(Q, R'_t) - S(Q, R_t) + \Delta) + \max(0, S(Q', R_t) - S(Q, R_t) + \Delta)], \tag{7.7}$$

where λ is a static coefficient to balance the ranking loss and the penalty and can be validated on the validation set. A low λ might cause the system to be over-confident on the prediction.

7.3.3 Object Interaction

We assume that the object types and their order in the language description can roughly determine when they appear in the video content, as motivated in Sec. 8.1. We show that this language prior can work as the frame-wise confidence score. To consider the interaction among objects, we further encode each object query feature q_k as:

$$J(q_k) = \text{MA}(q_k, Q, Q), \tag{7.8}$$

where $\text{MA}(\cdot, \cdot, \cdot)$ is the multi-head self-attention layer [82], taking in the (query, key, value) triplet. It represents each query as the combination of all other queries based on their inter-relations. The built-in positional encoding layer [82] in multi-head attention captures the order of objects appearing in the description. Note that the formulation is non-autoregressive, *i.e.*, all the objects in the same description can interact with each other.

We evenly divide each video segment into T' snippets and predict the confidence score for object k to appear in each snippet based upon the concatenation of $J(q_k)$ and q_k. Note that T' is a pre-specified constant that satisfies $T' \leq T$. The language-based confidence score $C_{lang} = (C^1_{lang}, \ldots, C^{T'}_{lang})$ is formulated as:

$$C_{lang} = \frac{1}{O} \sum_{k=1}^{O} \text{Sigmoid}(W_{lang}[J(q_k); q_k] + b_{lang}), \tag{7.9}$$

where $[\cdot\ ;\ \cdot]$ indicates the feature concatenation, $W_{lang} \in \mathbb{R}^{T' \times 2d}$ and $b_{lang} \in \mathbb{R}^{T'}$ are embedding weights and biases. We average the language-based and the similarity-based confidence score and rewrite Eq. 7.6 as:

$$L = \frac{1}{T} \sum_{t=1}^{T} [\lambda \frac{1}{2}(C_t + C^{t_s}_{lang}) L^t_{rank} - (1 - \lambda) \log(C_t + C^{t_s}_{lang})] \tag{7.10}$$

where $t_s = \min(\lceil t / \lceil \frac{T}{T'} \rceil \rceil, T)$ is the snippet index and $\lceil \cdot \rceil$ stands for the ceiling operator.

7.4 Experiments

7.4.1 Baselines and Metrics

Baselines. We include two competitive baselines from published work: DVSA [97] and GroundeR [94]. DVSA is the Grounding by Ranking method which we build all our methods upon. For fair comparison, all the approaches take in the same object proposals generated by Faster-RCNN [47] (pre-trained on MSCOCO). Following the convention from [97, 98], we select the top $N = 20$ proposals per frame and sample $T = 5$ frames per segment unless otherwise specified. We also evaluate the Baseline Random, which chooses a random proposal as the output.

Metrics. We evaluate the grounding quality by bounding box localization accuracy (denoted as Box Accuracy). The output is positive if the proposed box has over 50% IoU with the ground-truth annotation, otherwise negative. We compute accuracy for each object and average across all the object types.

7.4.2 Implementation Details

The number of snippets T' in Sec. 7.3.3 is set to 5. The encoding size d is 128 for all the methods. Object labels are represented as one-hot vectors, which are encoded by a linear layer without the bias term. The loss factor λ is cross-validated on the validation set and is set to 0.9. The ranking margin Δ is set to 0.1. For training, we use stochastic gradient descent (SGD) with Nesterov momentum. The learning rate is set at 0.05 and the momentum is 0.9. We implement the model in PyTorch and train it using either a single Titan Xp GPU with SGD or 4 GPUs with synchronous SGD, depending on the validation accuracy. The model typically takes 30 epochs, *i.e.*, 4 hours to converge.

When sampling frames from a segment, we evenly divide the segment into T clips and randomly sample one frame from each clip as temporal data augmentation. The negative sample sentence Q' is randomly sampled from all available sentences, but we exclude sen-

Method	Box Accuracy (%) Val.	Test
Compared methods		
Baseline Random	13.30	14.18
GroundeR [94]	19.63	19.94
DVSA [97]	30.51	30.80
Our methods		
Loss Weighting	30.07	31.23
Object Interaction	29.61	30.06
Full Model	30.31	31.73
Upper bound	57.77	58.56

Table 7.1: Evaluation on localizing objects from the grounding-truth captions.

tences that have overlapped objects with the positive sample Q. For self attention, we use a 2-layer 6-head multi-head attention module with the hidden size set to 256 and the dropout ratio set to 0.2.

For fair comparison, all the approaches take in the same object proposals generated by Faster-RCNN [47]. The model is based upon ResNet-101 and pre-trained on MSCOCO for the object detection task.[2] We take the 2048-dimension output after the RoI pooling as the region feature. We reduce the size of the region feature from 2048 to 128 with two linear layers, followed with dropout ($p = 0.2$) and ReLU.

7.4.3 Results on Object Grounding

The quantitative results on object grounding are shown in Tab. 7.1. The model with the highest score on the validation set is evaluated on the test split. We compute the upper bound as the accuracy when proposing all 20 proposals, to see how far the methods are from the performance limit. Note that the upper bound reported here is lower than that in [94]. This is largely due to the domain shift from general scenes to cooking scenes and the large variance in our object states, e.g. zoom-in and zoom-out views, onions v.s. fried onion rings.

[2] Details see `https://github.com/jwyang/faster-rcnn.pytorch`.

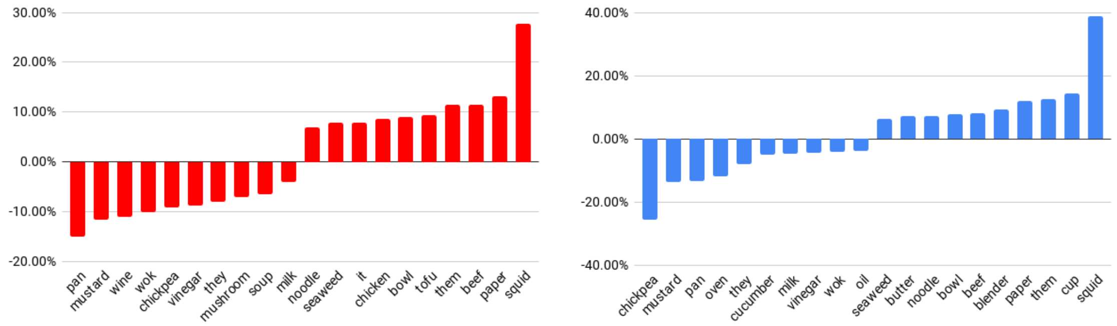
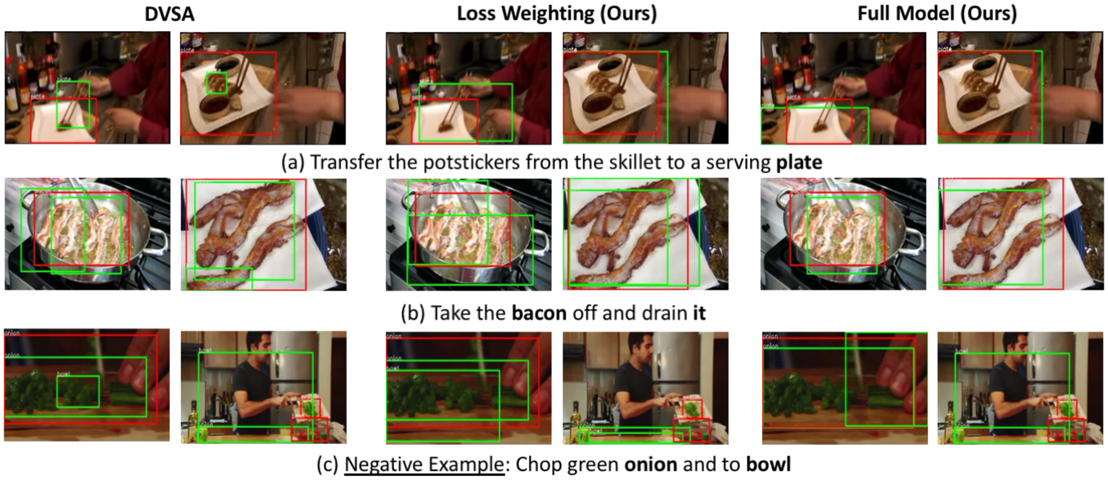


Figure 7.2: Top 10 accuracy increases & decreases by object category. (Left) Improvements of our Loss Weighting model over DVSA. (Right) Improvements of our Full Model over DVSA.

We show results on our proposed models, where the “Loss Weighting” model computes the confidence score with visual-semantic matching and the “Object Interaction” model computes the confidence score with textual guidance (Sec. 7.3.3). Our full model averages these two scores as the final confidence score (Eq. 7.10). The proposed methods demonstrate a steady improvement from the DVSA baseline, with a relative 1.40% boost from loss weighting and another 1.62% from combining object interaction, a total improvement of 3.02%. On the other hand, the baseline has a higher validation score, which indicates model overfitting. Note that text guidance alone (“Object Interaction”) works slightly worse than the baseline, showing that both visual and textual information are critical for inferring the frame-wise loss weights. Our methods also outperform other compared methods, GroundeR and Baseline Random by a large margin.

Analysis. We show in Fig. 7.2 the top 10 accuracy increases and decreases of our methods over the DVSA baseline, by object category. Our methods make better predictions on static objects such as “squid”, “beef”, and “noodle” and worse predictions on cookwares, such as “wok”, “pan”, and “oven”, which involves more state changes, such as containing/not containing food or different camera perspectives. Our hypothesis is, our loss weighting framework favors consistent objects across frames, due to the shared frame-wise supervision.

Impact of Sampling Rate. We investigate the impact of high video sampling rate on



(a) Transfer the potstickers from the skillet to a serving **plate**

(b) Take the **bacon** off and drain **it**

(c) Negative Example: Chop green **onion** and to **bowl**

Figure 7.3: Visualization of localization output from baseline DVSA and our proposed methods. Red boxes indicate ground-truths and green boxes indicate proposed regions. The first two rows show examples where our methods perform better than DVSA. The last row displays a negative example where all methods perform poorly. Better viewed in color.

grounding accuracy by increasing the total number of frames per segment (T) from 5 to 20. The accuracy from DVSA drops from 30.80% to 29.90% and the accuracy from our Loss Weighted model drops from 31.23% to 30.93%. We expected these inferior performances, due to the excessive object proposals. However, our loss weighted method only compromises 0.96% of the accuracy while the accuracy from DVSA drops by 2.92%, showing that our method is less sensitive to high sampling rate and predicts better on long frame sequences.

Qualitative Results. Fig. 7.3 visualizes the grounded objects with DVSA and our proposed methods. The first two rows show some positive examples. In Fig. 7.3 (a), we see with DVSA baseline the ”plate” object is grounded to the incorrect regions in the frames. However our methods correctly select regions with a large IOU with the ground truth box. In Fig. 7.3 (b) the labels ”bacon” and ”it” refer to the same target object. Per our annotation requirements, there is only one ground truth box instead of two. The full model correctly combines both ”bacon” and ”it” grounds them to the same region proposal. The last row that shows where all methods fail to ground the target objects adequately. This may be a result of errors in the top object proposals proposed since the scene is rather complicated.

An additional explanation may be bias in the dataset, where during training the ”bowl” object typically occupies the majority of the frame.

Limitations. There are two limitations in our method we hope to address in our future work. First, even though the frame-wise loss can to some degree enforce the temporal consistency between frames, we do not explicitly model the relation between frames, for instance motion information. The transition between object states across frames, e.g., raw meat to cooked meat, should be further studied. Second, our grounding performance is upper-bounded by the object proposal accuracy and we have no control over the errors from the proposals. An end-to-end version of the proposed method that solves both the proposing and the grounding problem can potentially improve the grounding accuracy.

7.5 Discussion

We propose a frame-wise loss-weighted model for weakly-supervised video object grounding. Our model applies segment-level labels to the frames in each segment, while being robust to inconsistencies between the segment-level label and each individual frame. We also leverage object interaction as textual guidance for grounding. We evaluate the effectiveness of our models on the newly-collected video grounding dataset YouCook2-BoundingBox. We show that our proposed methods outperform competitive baseline methods. However, even in the weakly-supervised setting, the model training still requires descriptions, obtained from tedious manual annotation. In the next chapter, we explore self-supervised grounding where descriptions are collected automatically from the web.

CHAPTER VIII

Grounding as Commonsense: Vision-Language Pre-training

8.1 Introduction

Inspired by the recent success of pre-trained language models such as BERT [27] and GPT [154, 155], there is a growing interest in extending these models to learning cross-modal representations like image-text [30, 88] and video-text [36, 37], for various vision-language tasks such as Visual Question Answering (VQA) and video captioning, where traditionally tedious task-specific feature designs and fine-tuning are required.

Table 8.1 summarizes some of the recent works on vision-language pre-training where all the models are unexceptionally built upon Bidirectional Encoder Representations from Transformers (BERT) [27]. These models use a two-stage training scheme. The first stage, called pre-training, learns the contextualized vision-language representations by predicting the masked words or image regions based on their intra-modality or cross-modality relationships on large amounts of image-text pairs. Then, in the second stage, the pre-trained model is fine-tuned to adapt to a downstream task.

Although significant improvements have been reported on individual downstream tasks using different pre-trained models, it remains challenging to pre-train a *single, unified* model that is universally applicable, via fine-tuning, to a wide range of vision-language

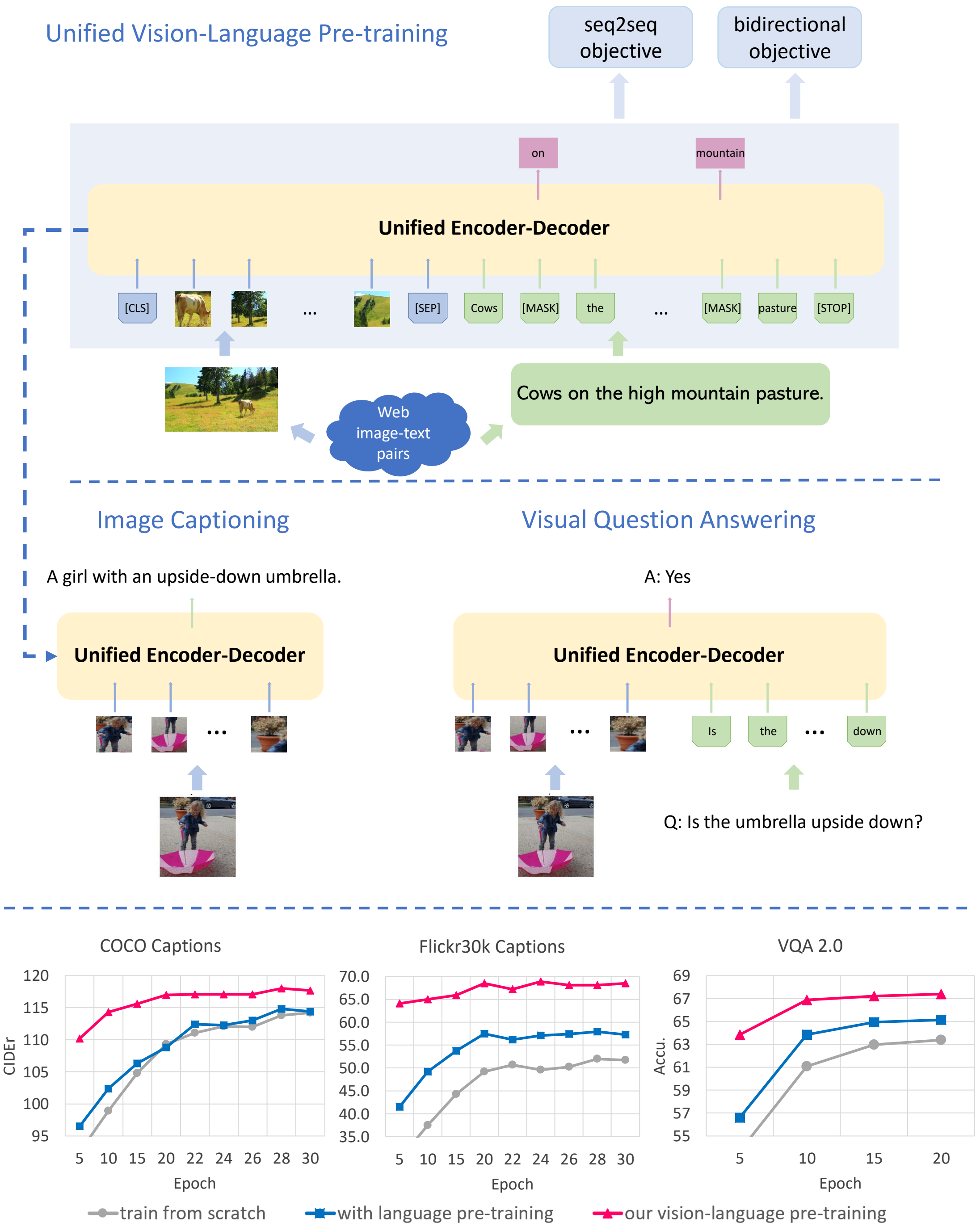


Figure 8.1: We propose a unified encoder-decoder model for general vision-language pre-training. The pre-trained model is then fine-tuned for image captioning and visual question answering. Thanks to our vision-language pre-training, both training speed and overall accuracy have been significantly improved on the downstream tasks compared to random initialization or language-only pre-training. All the results are evaluated on the validation set of the corresponding dataset.

Type	Method	Domain	Architecture	Downstream Tasks
Understanding-based only	LXMERT [88], ViLBERT [30], other similar works [35], [32], [31], [34], [33]	Image	Single-stream or two stream Transformer	Visual question answering Visual commonsense reasoning Image retrieval Grounding referring expressions
Generation-based and understanding-based	VideoBERT [36]	Video	Single-stream Transformer+ Masked Transformer [2]	Zero-shot action classification Video captioning
	CBT [37]	Video	Two-stream Transformer encoder+ Transformer decoder	Action anticipation Video captioning
	Our VLP	Image	Single unified encoder-decoder	Visual question answering Image captioning

Table 8.1: Comparison between our method and other vision-language pre-training works.

tasks as disparate as vision-language generation (*e.g.*, image captioning) and understanding (*e.g.*, VQA). Most existing pre-trained models are either developed only for understanding tasks, as denoted by "understanding-based only" in Tab. 8.1, or designed as hybrid models that consist of multiple modality-specific encoders and decoders which have to be trained separately in order to support generation tasks. For example, VideoBERT and CBT in Tab. 8.1 perform pre-training only for the encoder, not for the decoder. This causes a discrepancy between the cross-modal representations learned by the encoder and the representation needed by the decoder for generation, which could hurt the generality of the model. In this work, we strive to develop a new method of pre-training a unified representation for both encoding and decoding, eliminating the aforementioned discrepancy. In addition, we expect that such a unified representation would also allow more effective cross-task knowledge sharing, reducing the development cost by eliminating the need of pre-training different models for different types of tasks.

To this end, we propose a unified encoder-decoder model, called the Vision-Language Pre-training (VLP) model, which can be fine-tuned for both vision-language generation and understanding tasks. The VLP model uses a shared multi-layer Transformer network [82] for encoding and decoding, pre-trained on large amounts of image-caption pairs [156], and optimized for two unsupervised vision-language prediction tasks: bidirectional and sequence to sequence (seq2seq) masked language prediction. The two tasks differ solely in what context the prediction conditions on. This is controlled by utilizing specific self-attention masks for the shared Transformer network. In the bidirectional prediction task, the context of the masked caption word to be predicted consists of all the image regions and all the words on its right and left in the caption. In the seq2seq task, the context consists of all the image regions and the words on the left of the to-be-predicted word in the caption.

We validate VLP in our experiments on both the image captioning and VQA tasks using three challenging benchmarks: COCO Captions [114], Flickr30k [115], and VQA 2.0 dataset [104]. We observe that compared to the two cases where we do not use any pre-trained model or use only the pre-trained language model (*i.e.*, BERT), using VLP significantly speed-ups the task-specific fine-tuning and leads to better task-specific models, as shown in Fig. 8.1. More importantly, without any bells and whistles, our models achieve state-of-the-art results on both tasks across all three datasets.

8.2 Vision-Language Pre-training

We denote the input image as I and the associated/target sentence description (words) as S. We extract a fixed number N of object regions from the image using an off-the-shelf object detector, denoted as $\{r_1, \ldots, r_N\}$ and the corresponding region features as $R = [R_1, \ldots, R_N] \in \mathbb{R}^{d \times N}$, region object labels (probabilities) as $C = [C_1, \ldots, C_N] \in \mathbb{R}^{l \times N}$, and region geometric information as $G = [G_1, \ldots, G_N] \in \mathbb{R}^{o \times N}$, where d is the embedding size, l indicates the number of the object classes of the object detector, and $o = 5$ consists of four values for top left and bottom right corner coordinates of the region bounding box (normalized between 0 and 1) and one value for its relative area (*i.e.*, ratio of the bounding box area to the image area, also between 0 and 1). The words in S are represented as one-hot vectors which are further encoded to word embeddings with embedding size e: $y_t \in \mathbb{R}^e$ where $t \in \{1, 2, \ldots, T\}$ and T indicates the length of the sentence.

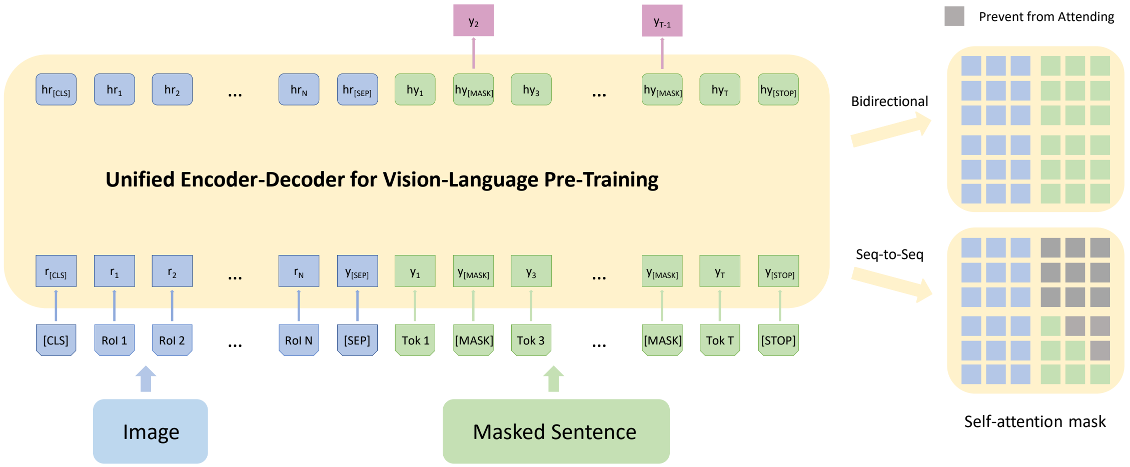


Figure 8.2: Model architecture for pre-training. The input comprises of image input, sentence input, and three special tokens ([CLS], [SEP], [STOP]). The image is processed as N Region of Interests (RoIs) and region features are extracted according to Eq. 8.1. The sentence is tokenized and masked with [MASK] tokens for the later masked language modeling task. Our Unified Encoder-Decoder consists of 12 layers of Transformer blocks, each having a masked self-attention layer and feed-forward module, where the self-attention mask controls what input context the prediction conditions on. We implemented two self-attention masks depending on whether the objective is bidirectional or seq2seq. Better viewed in color.

8.2.1 Vision-Language Transformer Network

Our vision-language Transformer network, which unifies the Transformer encoder and decoder into a single model, is depicted in Fig. 8.2 (left). The model input consists of the class-aware region embedding, word embedding and three special tokens. The region embedding is defined as:

$$r_i = W_r R_i + W_p[\text{LayerNorm}(W_c C_i)|\text{LayerNorm}(W_g G_i)] \tag{8.1}$$

where $[\cdot|\cdot]$ indicates the concatenation on the feature dimension, LayerNorm represents Layer Normalization. The second term mimics the positional embedding in BERT, but adding extra region class information, and W_r, W_p, W_c, W_g are the embedding weights (the bias term and the nonlinearity term are omitted). Note that here we overload the notation of $r_i \in \mathbb{R}^d$ $(i \in \{1, 2, ..., N\})$ to also represent class-aware region embeddings. In addition, we add segment embeddings to r_i as in BERT where all the regions share the same segment embedding where the values depend on the objectives (*i.e.*, seq2seq and bidirectional, see the following section).

The word embeddings are similarly defined as in [27], adding up y_t with positional embeddings and segment embeddings, which is again overloaded as y_t. We define three special tokens [CLS], [SEP], [STOP], where [CLS] indicates the start of the visual input, [SEP] marks the boundary between the visual input and the sentence input, and [STOP] determines the end of the sentence. The [MASK] tokens indicate the masked words which will be explained in the next section.

8.2.2 Pre-training Objectives

In the BERT masked language modeling objective, 15% of the input text tokens are first replaced with either a special [MASK] token, a random token or the original token, at random with chances equal to 80%, 10%, and 10%, respectively. Then, at the model output, the hidden state from the last Transformer block is projected to word likelihoods where the masked tokens are predicted in the form of a classification problem. Through this reconstruction, the model learns the dependencies in the context and forms a language model. We follow the same scheme and consider two specific objectives: the bidirectional objective (bidirectional) as in BERT and the sequence to sequence objective (seq2seq), inspired by [86].

As shown in Fig. 8.2 (right), the only difference between the two objectives lie in the self-attention mask. The mask used for the bidirectional objective allows unrestricted message passing between the visual modality and the language modality while in seq2seq, the to-be-predicted word cannot attend to the words in the future, *i.e.*, it satisfies the auto-regressive property. More formally, we define the input to the first Transformer block as $H^0 = [r_{\text{[CLS]}}, r_1, \ldots, r_N, y_{\text{[SEP]}}, y_1, \ldots, y_T, y_{\text{[STOP]}}] \in \mathbb{R}^{d \times U}$ where $U = N+T+3$, and then the encoding at different levels of Transformer as $H^l = \text{Transformer}(H^{l-1}),\ l \in [1, L]$.

We further define a self-attention mask as $M \in \mathbb{R}^{U\times U)}$, where

$$M_{jk} = \begin{cases} 0, & \text{allow to attend} \\ -\infty, & \text{prevent from attending} \end{cases} \qquad j, k = 1, \ldots, U. \tag{8.2}$$

For simplicity, we assume a single attention head in the self-attention module. Then, the self-attention output on H^{l-1} can be formulated as:

$$A^l = \text{softmax}\left(\frac{Q^\top K}{\sqrt{d}} + M\right) V^\top, \tag{8.3}$$

$$V = W_V^l H^{l-1},\ Q = W_Q^l H^{l-1},\ K = W_K^l H^{l-1}, \tag{8.4}$$

where W_V^l, W_Q^l, and W_K^l are the embedding weights (the bias terms are omitted). The intermediate variables V, Q, and K indicate values, queries and keys, respectively, as in the self-attention module [82]. A^l is further encoded by a feed-forward layer with a residual connection to form the output H^l. During the pre-training, we alternate per-batch between the two objectives and the proportions of seq2seq and bidirectional are determined by hyper-parameters λ and $1 - \lambda$, respectively.

It is worth noting that in our experiments we find that incorporating the region class probabilities (C_i) into region feature (r_i) leads to better performance than having a masked region classification pretext as in [30, 88]. Therefore, differing from existing works where masked region prediction tasks are used to refine the visual representation, we indirectly refine the visual representation by utilizing it for masked language reconstruction. We also choose not to use the Next Sentence Prediction task as in BERT, or in our context predicting the correspondence between image and text, because the task is not only weaker than seq2seq or bidirectional but also computationally expensive. This coincidentally agrees with a concurrent work of RoBERTa [157].

Sequence-to-sequence inference. Similar to the way seq2seq training is performed, we can directly apply VLP to sequence-to-sequence inference, in the form of beam search.

More details follow next in the Image Captioning section.

8.3 Fine-Tuning for Downstream Tasks

8.3.1 Image Captioning

We fine-tune the pre-trained VLP model on the target dataset using the seq2seq objective. During inference, we first encode the image regions along with the special [CLS] and [SEP] tokens and then start the generation by feeding in a [MASK] token and sampling a word from the word likelihood output (*e.g.*, greedy sampling). Then, the [MASK] token in the previous input sequence is replaced by the sampled word and a new [MASK] token is appended to the input sequence to trigger the next prediction. The generation terminates when the [STOP] token is chosen. Other inference approaches like beam search could apply as well.

8.3.2 Visual Question Answering

We frame VQA as a multi-label classification problem. In this work we focus on open domain VQA where top k most frequent answers are selected as answer vocabulary and used as class labels. Following [73] we set k to 3129.

During the fine-tuning, a multi-layer Perceptron (Linear+ReLU+Linear+Sigmoid) on top of the element-wise product of the last hidden states of [CLS] and [SEP] is learned, similar to [30]. We optimize the model output scores with respect to the soft answer labels using cross-entropy loss. Note that unlike [88] where the task-specific objective (*i.e.*, VQA) is exploited during pre-training by using the target datasets (from intensive human annotations), our pre-training does not have this requirement and is therefore more general.

Dataset	Batch size	LR	# of epochs	GPUs	T/E
CC	64(x8)	1e-4(x8)	30	8x V100	5hr
COCO	64(x8)	3e-5(x8)	30	8x V100	12min
VQA 2.0	64(x2)	2e-5(x2)	20	2x V100	32min
Flickr30k	64(x8)	3e-5(x8)	30	8x V100	3min
COCO (w/o pre-training)	64(x8)	3e-4(x8)	30	8x V100	12min
COCO (SCST training)	16(x4)	1e-6(x4)	30	4x Titan Xp	3hr

Table 8.2: Model hyper-parameters and training specifications. LR indicates learning rate and T/E indicates time per epoch.

8.4 Experiments and Results

Data preparation. We conduct pre-training on the Conceptual Captions (CC) dataset [156] which has around 3 million web-accessible images with associated captions. The datasets for downstream tasks include COCO Captions [114], VQA 2.0 [104] and Flickr30k [115] (described in Sec. 3.3). For all the dataset, we trim long sentences and pad short sentences to 20 words and all the words are tokenized and numericalized as in BERT [27].

Implementation details. Our Transformer backbone is the same as BERT-base [27]. The input of the network consists of image (regions) and the associated/target caption. We represent each input image as 100 object regions extracted from a variant of Faster R-CNN [47] with ResNeXt-101 FPN backbone [150] pre-trained on Visual Genome [147, 73, 158]. We take the model output from fc6 layer as the region feature (R_i) and fine-tune the fc7 layer. The class likelihood on the 1600 object categories as region object labels (C_i). Note that if not specified, the weights in our BERT model are initialized from UniLM [86] pre-trained on text corpora only. For caption inference, we use greedy search on the validation set and beam search with beam size 5 on the test set. The same training optimizer including learning rate scheduler is used as in BERT [27]. We perform light model/training hyper-parameter search with the configurations presented in Tab. 8.2. The SCST training on COCO is performed after the VLP pre-training and COCO fine-tuning. λ is set to 0.75 for CC pre-training from light model validation (out of $\{0.25, 0.5, 0.75\}$),

	COCO				VQA 2.0 (Test-Standard)				Flickr30k			
Method	B@4	M	C	S	Overall	Yes/No	Number	Other	B@4	M	C	S
BUTD [73]	36.2	27.0	113.5	20.3	65.7	-	-	-	27.3	21.7	56.6	16.0
NBT (with BBox) [74]	34.7	27.1	107.2	20.1	-	-	-	-	27.1	21.7	57.5	15.6
GCN-LSTM (spa) [76]	**36.5**	27.8	115.6	20.8	-	-	-	-	-	-	-	-
GCN-LSTM (sem)	**36.8**	27.9	116.3	20.9	-	-	-	-	-	-	-	-
GVD [26]	-	-	-	-	-	-	-	-	26.9	22.1	60.1	16.1
GVD (with BBox)	-	-	-	-	-	-	-	-	27.3	22.5	62.3	16.5
BAN [101]	-	-	-	-	70.4	85.8	**53.7**	**60.7**	-	-	-	-
DFAF [102]	-	-	-	-	70.3	-	-	-	-	-	-	-
AoANet* [159]	37.2	28.4	119.8	21.3	-	-	-	-	-	-	-	-
ViLBERT* [30]	-	-	-	-	70.9	-	-	-	-	-	-	-
LXMERT* [88]	-	-	-	-	72.5	88.2	54.2	63.1	-	-	-	-
Ours												
w/o VLP (baseline)	35.5	28.2	114.3	21.0	70.0	86.3	52.2	59.9	27.6	20.9	56.8	15.3
seq2seq PT only	**36.5**	**28.4**	**117.7**	**21.3**	70.2	86.7	52.7	59.9	**31.1**	**23.0**	**68.5**	**17.2**
bidirectional PT only	36.1	28.3	116.5	21.2	**71.3**	**87.6**	**53.5**	**61.2**	**30.5**	22.6	63.3	16.9
Unified VLP	**36.5**	**28.4**	**116.9**	**21.2**	**70.7**	**87.4**	52.1	60.5	30.1	**23.0**	**67.4**	**17.0**

Table 8.3: Results on COCO Captions test set (with cross-entropy optimization only, all single models), VQA 2.0 Test-Standard set and Flickr30k test set. * indicates unpublished works by Sept. 2019. PT indicates pre-training, B@4 represents for BLEU@4, M for METEOR, C for CIDEr, and S for SPICE. Results on previous works are obtained from the original papers. Top two results on each metric are in bold. The improvement of Unified VLP over the baseline method (w/o VLP) on Flickr30k is statistically significant (p-value$<$0.02).

and set to 1 for image captioning (*i.e.*, full seq2seq) and 0 for VQA (*i.e.*, full bidirectional).

Model variants and metrics. To demonstrate the effectiveness of our vision-language pre-training, we first include a baseline model without this pre-training. We then include two extreme settings of our model with $\lambda = 1$ (seq2seq pre-training only) and $\lambda = 0$ (bidirectional pre-training only) to study how each objective individually works with different downstream tasks. Our full model conducts joint training on the two objectives. The fine-tuning procedure is performed the same regardless of the pre-training configurations. Regarding evaluation metrics, we use standard language metrics for image captioning, including Bleu@4, METEOR, CIDEr, and SPICE and the official measurement on accuracy for VQA, over different answer types including Yes/No, Number, and Other.

Comparisons against SotAs. Results comparing our methods and SotA methods on the test set are in Tab. 8.3. We include state-of-the-art published works (upper part of Tab. 8.3), unpublished works that are currently in submission (middle part), and our methods (lower

	COCO (w/ CIDEr optimization)			
Method	B@4	M	C	S
BUTD	36.3	27.7	120.1	21.4
GCN-LSTM (spa)	38.2	28.5	127.6	22.0
SGAE [77]	38.4	28.4	127.8	22.1
AoANet*	38.9	29.2	129.8	22.4
Ours (Unified VLP)	**39.5**	**29.3**	**129.3**	**23.2**

Table 8.4: Results on COCO Captions test set (with CIDEr optimization, all single models). *** indicates unpublished works**. Top one result on each metric is in bold.

part). All the image captioning methods are single models, with cross-entropy optimization only for a fair comparison. Our full model (Unified VLP) outperforms SotA methods on three out of four metrics on COCO, overall accuracy on VQA 2.0, and all four metrics on Flickr30k. The improvements are particularly sound on Flickr30k, where we get *5.1% absolute gain* on CIDEr metric and *2.8%* on BLEU@4.

We further perform CIDEr optimization on COCO Captions through Self-Critical Sequence Training (SCST) [160], as in most of the recent image captioning literatures. The results are in Tab. 8.4 where our full model sets new SotA on all the metrics.

Boost from pre-training. Our full model leads our baseline model by a large margin on most of the metrics thanks to our pre-training. Some noticeable improvements include over *10% absolute gain* on CIDEr metric on Flickr30k, and over *2% gain* on CIDEr on COCO and B@4, METEOR on Flickr30k. Small datasets (*i.e.*, Flickr30k) benefit the most as vision-language pre-training alleviates overfitting issues. Our model variants under the two extreme settings work well as expected on their "favorable" tasks, *i.e.*, seq2seq pre-training alone improves downstream captioning tasks significantly and bidirectional pre-training benefits understanding tasks (*i.e.*, VQA), but not the opposite. They set new SotAs on all metrics except the "Number" accuracy on VQA 2.0. The joint training organically combines the representations learned from the two rather different objectives and yields slightly compromised but decent accuracy on all the downstream tasks. That said, from an engi-

	COCO				VQA 2.0				Flickr30k			
Method	B@4	M	C	S	Overall	Yes/No	Number	Other	B@4	M	C	S
From scratch	34.5	28.1	114.2	21.1	63.4	80.2	46.4	55.2	26.9	20.8	52.1	14.4
Init from BERT	34.6	**28.4**	114.8	21.4	65.1	82.9	48.0	56.1	27.5	21.9	58.4	15.5
Init from UniLM												
w/o VLP (baseline)	34.5	28.1	113.9	21.3	66.1	83.8	49.7	56.9	27.5	21.5	58.3	15.3
seq2seq PT only	**35.3**	**28.4**	**116.7**	**21.5**	66.4	84.6	**50.1**	56.9	28.9	**23.6**	67.0	**17.2**
bidirectional PT only	**35.3**	28.3	116.1	21.4	**68.2**	**85.6**	**51.9**	**59.3**	**29.6**	23.2	**67.2**	16.8
Unified VLP	**35.5**	**28.5**	**118.0**	**21.6**	**67.4**	**85.4**	**50.1**	**58.3**	**29.7**	**23.8**	**69.1**	**17.6**

Table 8.5: Results on COCO Captions, VQA 2.0, and Flickr30k validation set. PT indicates pre-training, B@4 represents for BLEU@4, M for METEOR, C for CIDEr, and S for SPICE. Top two results on each metric are in bold.

	COCO				VQA 2.0 (Test-Dev)				Flickr30k			
Method	B@4	M	C	S	Overall	Yes/No	Number	Other	B@4	M	C	S
From scratch	35.2	27.9	112.5	20.6	67.7	83.5	50.7	58.1	28.4	20.8	53.5	15.2
Init from BERT	34.8	28.1	112.6	20.7	68.6	85.2	50.9	58.3	29.1	21.7	60.4	15.9
Init from UniLM	35.5	28.2	114.3	21.0	69.6	86.1	**52.4**	59.4	27.6	20.9	56.8	15.3
Unified VLP	**36.5**	**28.4**	**116.9**	**21.2**	**70.5**	**87.2**	52.1	**60.3**	**30.1**	**23.0**	**67.4**	**17.0**

Table 8.6: Impact of different levels of pre-training on downstream tasks. All results are on the test set (Test-Dev for VQA 2.0). Top one result on each metric is in bold.

neering perspective, if we can afford having separate pre-training models for generation task or understanding task, we will get the optimal model performance. If we value model architecture and parameter sharing, the joint model is a good trade-off. Note that we also include the corresponding results on the val set in Tab. 8.5.

Impact of pre-training types. Depending on how the base model Transformer is initialized, we define four "degrees" of pre-training from weakest to strongest as i) without any pre-training, *i.e.*, base model is trained from scratch, ii) bidirectional language pre-training, *i.e.*, base model is initialized from BERT weights [27], iii) seq2seq and bidirectional language pre-training, *i.e.*, base model is initialized from UniLM weights [86] which is our baseline setting, and iv) our full Vision-Language Pre-training. The corresponding fine-tuning results on downstream tasks are presented in Fig. 8.1 and Tab. 8.5 on the val set and Tab. 8.6 on the test set. As shown from the figure, our vision-language pre-training significantly accelerates the learning process of downstream tasks and contributes to better overall accuracy. It is worth noting that the learning process of VQA is greatly shortened despite that the hidden states associated with tokens [CLS] and [SEP] are not learned dur-

Method	B@4	M	C	S
From scratch	5.5	9.4	63.8	14.9
Init from BERT	5.7	9.7	66.7	15.3
Init from UniLM	5.8	9.7	67.0	15.5

Table 8.7: Impact of model weight initializations on pre-training. Results are on Conceptual Captions val set on caption generation.

Method	B@4	M	C	S
Region label as pretext	5.4	9.4	62.2	14.5
Region label probability as input	5.8	9.7	67.0	15.5

Table 8.8: Comparison between having region class prediction pretext and feeding in class probabilities as a part of the model input. Results are on Conceptual Captions val set.

ing the pre-training. This indicates that the contextualized vision-language representations can generalize to unseen domains and work reasonable well as a warm-start for new tasks. Note that for VQA 2.0, all the methods here are only trained on the training set while for the results reported on the test set (Tab. 8.3 and Tab. 8.6), all the models are trained on both training set and validation set following the practice from early works.

We also study how the pre-training types 1-3 influence our vision-language pre-training in terms of caption generation. The results on Conceptual Captions val set at epoch 20 are shown in Tab. 8.7. All the models are trained based on the unified VLP objective ($\lambda = 0.75$) for a fair comparison. We observe that initializing base model with weights transferred from pure language pre-training benefits vision-language pre-training. The training objectives of UniLM are closer to our seq2seq and bidirectional objectives than the ones in BERT and hence we hypothesize that this counts for the slightly larger improvement. Note that our intention here is to demonstrate how different weight initializations can influence pre-training performance rather than pursuing possibly high quantitative scores (with full seq2seq training, CIDEr could climb to 77.2 after training for 30 epochs).

Region object labels as pretext. Existing works [26, 74] regard region object labels (probabilities) (C_i) as an important auxiliary to enrich image region features and here we follow a similar design. We can also instead use these labels for a masked region classification pretext as in [88]. Here we have a comparison over the two design choices. “region label probability as input” is equivalent to our full model Unified VLP and “region label as pretext” is the implementation from [88]. As shown in Tab. 8.8, predicting class labels as a pretext has a negative impact on the pre-training, in terms of captioning performance. We hypothesize that this is because the class labels from the off-the-shelf object detector might be noisy which compromises the learned feature representation. In contrast, our model refines the visual representation through a more reliable masked language modeling and could correct the errors exist in the class labels.

Qualitative results and analyses. Qualitative examples on COCO Captions and VQA 2.0 are shown in Fig. 8.9. In the first two examples, our full model with vision-language pre-training captures more details in the image, such as “umbrellas” and “a blue wall” than the baseline methods. It also answers questions correctly. In the third example, all the methods dis-identify the gondola as a train due to their visual similarity. When it comes to the question answering, our methods all give correct answers while the GT answer is incorrect (note that there is a person in the gondola). In the fourth example, all the models mistakenly classify the activity as “surfing” while the correct one is “kayaking/boating”. This is consistent across both the caption model and the VQA model, which implies that the feature representations are indeed shared across tasks.

8.5 Discussion

This chapter presents a unified Vision-Language Pre-training (VLP) model that can be fine-tuned for both vision-language generation and understanding tasks. The model is pre-trained on large amounts of image-text pairs based on two objectives: bidirectional and seq2seq vision-language prediction. The two disparate objectives are fulfilled under the same architecture with parameter sharing, avoiding the necessity of having separate pre-trained models for different types of downstream tasks (*i.e.*, generation-based or understanding-based). In our comprehensive experiments on image captioning and VQA tasks, we demonstrate that the large-scale unsupervised pre-training can significantly speed up the learning on downstream tasks and improve model accuracy. Besides, compared to having separate pre-trained models, our unified model combines the representations learned from different objectives and yields slightly compromised but decent (SotA) accuracy on all the downstream tasks. Finally, we emphasize that VLP is self-supervised, which means that the training of VLP requires no human annotation. VLP is applicable to various input data formats and the learned vision-language representation (grounding) is generic to various scenarios.

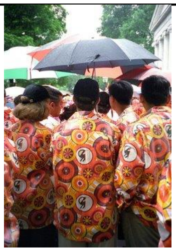
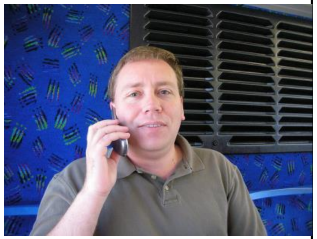
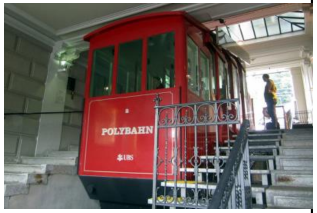
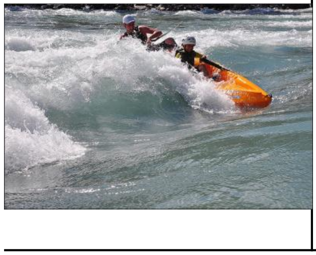
	**GT sentences:** People in matching shirts standing under umbrellas in the sun People in the same colorful shirts have umbrellas. A large group of people with an umbrella outside. A group of men standing next to a lot of umbrellas A group of people that are under one umbrella	**Unified VLP** (159.8): A group of people standing under umbrellas in the rain. **Init from UniLM** (59.0): A group of people standing around each other. **Init from BERT** (59.0): A group of people standing around each other.	**Question**: Are they dressed the same? **Correct answer**: Yes **Unified VLP**: Yes **Init from UniLM**: No **Init from BERT**: No
	**GT sentences:** A man standing in front of a blue wall A man talks on a phone in a room with blue wallpaper A man holding a cell phone standing in front of blue wallpaper with designs and a large wall vent A man on a cell phone by a bright blue wall A man holding a phone to his ear	**Unified VLP** (180.6): A man talking on a cell phone in front of a blue wall. **Init from UniLM** (126.9): A man talking on a cell phone while standing next to a blue wall. **Init from BERT** (59.6): A man talking on a cell phone while wearing a gray shirt.	**Question**: Is the man taking his own picture? **Correct answer**: No **Unified VLP**: No **Init from UniLM**: Yes **Init from BERT**: Yes
	**GT sentences:** A man standing by a large air gondola that is docked in a station A train is parked as a man at the top of the stairs waits along side it. Small tram bus parked between two stair cases A man standing next to cable car and a flight of stairs A man getting ready to board the trolley car	**Unified VLP** (28.0): A red train is parked in a station. **Init from UniLM** (36.9): A red train with a man standing on the top of it. **Init from BERT** (21.3): A red train car sitting inside of a train station.	**Question**: How many people are here? **Correct answer**: 1 **Unified VLP**: 2 **Init from UniLM**: 2 **Init from BERT**: 2
	**GT sentences:** Two boaters are white water rafting through rough currents. Two people in a small boat in a body of water There are people on a boat tube in the water Two people riding a raft through some waves Two people in a canoe in some rapids	**Unified VLP** (7.5): A man riding a surfboard on top of a wave. **Init from UniLM** (7.6): A man and a boy are riding a surfboard on a wave. **Init from BERT** (5.4): A man riding a paddle board on top of a wave.	**Question**: What is the person doing? **Correct answer**: kayaking/boating **Unified VLP**: surfing **Init from UniLM**: surfing **Init from BERT**: surfing

Table 8.9: Qualitative examples on COCO Captions and VQA 2.0. The first column indicates images from the COCO validation set. The second column shows the five human-annotated ground-truth (GT) captions. The third column indicates captions generated by three of our methods and the corresponding CIDEr scores, where only Unified VLP has vision-language pre-training. The last column shows VQA questions and correct answers associated with the image and answers generated by our models. The top two are successful cases and the bottom two are failed cases. See text for details.

CHAPTER IX

Conclusion

In this dissertation, we first propose the largest-of-its-kind video-language benchmark YouCook2 dataset and ActivityNet-Entities dataset in Chap. III. The rest of the chapters circle around two main problems: video description and video grounding. For video description, we first address the problem of decomposing a long video into compact and self-contained event segments in Chap. IV. Based on these individual event segments, we propose a non-recurrent approach (*i.e.*, Transformer) for video description generation in Chap. V as opposed to prior RNN-based methods. In Chap. VI, we introduce a grounded video description framework, transitioning our focus from end-to-end systems to visually-grounded systems which yield better model interpretability. The next two chapters study visual grounding in an annotation-efficient fashion and demonstrate its positive impact on downstream tasks. Throughout this dissertation, we elaborated how language plays a significant role in video understanding and how it delivers a holistic view of the video content through compact descriptions.

9.1 Takeaways and Lessons Learned

Here are some takeaways and lessons we have learned so far on video description, grounding, and related areas throughout the dissertation.

The power of data. The majority of this dissertation is on supervised learning where the target output is guided by ground-truth human annotations during training. The effectiveness of supervised learning has been demonstrated in almost every aspect of machine learning applications, such as vision [127, 97], language [62, 82], speech [161, 162] and has made no exception in our study. Pre-trained models on large-scale data corpus, either vision-based [123], language-based [27], or cross-model [38], have led to significant performance boost on various downstream tasks. In computer vision, supervised pre-training is pervasive and has been largely taken granted. In computational linguistics, this sort of transfer learning emerged recently and has dominated major research areas, particularly those that require language understanding. Interestingly, there has been heated debates in computer vision on whether supervision in pre-training is necessary [163] or pre-training at all [164]. Before more evidence on the effectiveness of the recent challengers, supervised pre-training is believed to continue dominating for the foresee future.

Transformer and BERT. Transformer and its variant BERT language model have been intensively used in this dissertation. Our discussion here mainly focuses on BERT, as it represents a wider range of concurrent models on contextualized representation learning. Due to the complexity presents in the model, BERT remains a "black-box" and the internal mechanism behind why multi-head self-attention benefits representation learning is still unclear. Some studies have shown that no single attention head in BERT has the complete syntactic tree information as such human would define, while only preserves partial knowledge of syntax [165, 166]. This makes it challenging to probe/visualize what BERT has learned and opens opportunities on potential future works (discussed later in Sec. 9.2). Overall, we have seen in our work the capability of Transformer on learning semantic knowledge, both visual and linguistic concepts. Rogers et al. [167] summarizes probing studies on BERT into a primer in "BERTology", stating that BERT also preserves syntactic knowledge and world knowledge. How much of the knowledge and what knowledge could be transferred to downstream tasks are still among the many open questions. For example, we mentioned in Chap. VIII, the fourth qualitative example in Fig. 8.9, our models consistently misclassify the activity as “surfing” across all the downstream tasks while the correct answer is “kayaking/boating”. This implies that the model has transferred generic semantic knowledge from pre-training. How to identify and quantify the transferred knowledge is an interesting future direction.

Blessing and curse of inductive bias. Inductive bias is ubiquitous in machine learning. It essentially leads to a simpler and more generic target function (relative to overfitting) that helps the model to generalize beyond the training data. As far as we’ve seen in this dissertation, most of the inductive bias learned through model training is beneficial in the sense that it facilitates the learning process. For instance, in grounding, the models learns to attend to the center of the scene as this is where the object often appears; in procedure segmentation, the model tends to distribute segments uniformly across the video as recipe proceeds along with the video timeline; in description generation, the model frequently predicts confident common words (*e.g.*, man, woman, standing, talking) in the outcome and leads to high automatic evaluation metrics. However, there are downsides. First, the model might shortcut the difficult in representation learning and make predictions based on trivial visual clues (*e.g.*, center bias, color bias). Second, it leads to low vocabulary diversity in description generation. How to mitigate the negative impact of inductive bias remains an open question. The following will shed some light on this.

Reconstruction *vs.* contrastive learning. Self-supervised representation learning has become a hot spot in machine learning community recently. We are familiar with one of its famous application in language understanding — BERT, which is mainly based on (corrupted) input reconstruction. Also, we have talked a lot about vision-language pre-training, which is inspired by BERT and learns the joint embedding space in a self-supervised fashion. In pure computer vision, there has not had any dominating methods. Early methods such as Jigsaw [168] and Colorization [169] are reconstruction-based. However, performing a pretext on the same image sample permits model to learn trivial visual shortcut (*e.g.*, local color and textual bias in Jigsaw), which compromises the feature learning. Therefore, later methods instead adopt contrastive learning to ensure the source sample and the target sample differ in trivial clues (*e.g.*, histograms of pixel intensities [170]). Now, switch the gear to the language domain. We notice that in BERT, besides the reconstruction-based pretext (*i.e.*, Masked Language Modeling, or MLM), it also has a contrastive objective called Next Sentence Prediction, or NSP. However, Liu et al. [157] demonstrate that NSP is rather optional compared to the main MLM objective. So, contrastive learning had a major setback in the language domain. Would it eventually replace reconstruction-based objective? More study needs to be done.

Deep learning "alchemy". We cannot emphasize more on the engineering aspect of the dissertation, including but not limited to efficient data loading and storage, model architecture design, and hyper-parameter search. Andrej Karpathy had an awesome blog on tricks, pitfalls, and caveats on training Neural Networks[1] and is a good supplementary read. Among all the lessons we learned over time, learning rate tuning is always the most basic and one of the most important routines in model training (*e.g.*, apply a coarse-to-fine search). Besides, tricks like model regularization (*i.e.*, weight decay), auxiliary losses (*i.e.*, attention/grounding loss in GVD), and dropout are generally helpful, but the impact is sometimes limited.

9.2 Perspectives on Future Work

Quo vadis, instructional video understanding? Instructional video understanding has become one of the major focuses in the video-language research community and is the center of this dissertation. This nascent field demonstrates strong potential in bridging vision modality and language modality in a *annotation-free* fashion (self-supervised through video and its auto-generated ASR transcripts). The learned multi-modal representation has shown to generalize well to unseen domains and tasks in preliminary studies [36, 37, 24, 16]. As self-supervised learning increasingly appears to be the next breakthrough in vision-

[1] http://karpathy.github.io/2019/04/25/recipe/

language research, the keen questions to ask include i) what data can further benefit representation learning and ii) what model can digest the enormous data we have or soon to come.

Regarding the first question, we obviously are eager for larger datasets, but we reckon that the data quality is even more important. The current description data (ASR transcripts) present the following issues:

- missing or incorrect punctuation, which leads to imprecise alignment between video clips and sentences,
- missing sentences, sometimes even a whole conversation block,
- grammar mistakes, such as "a" *vs.* "the", singular *vs.* plural,
- other speech recognition-related errors.

For a cleaner language supervision, one way is to substitute ASR transcripts with user-uploaded ("GT") transcripts. These transcripts usually have high quality and accurate punctuations so the video clip can align well with the text. We sampled a small portion of the 1.22M videos from HowTo100M dataset [24] and observed around 10% videos have user-uploaded speech transcripts (snippet.trackKind=default).

Model-wise, the following problems remain to be further explored. First of all, the multi-head and multi-layer natural makes it cumbersome to visualize the region/temporal attention in BERT-inspired vision-language models. There are some existing attempts [32, 171] but more efforts are required in demystifying the specific functionality of different heads/layers in the model. Secondly, as self-attention resembles the "complete" graph version of Graph Attention Networks [172], how to sparsify the connections and improve computation efficiency is an open question. For instance, when we model the intra-sentence relationship, we can simply keep the graph connections determined by a parsing tree (*i.e.*, dependency parsing) and study what impact could inductive biases have in model training speed. This could potentially make the model more data efficient. Finally, under the context of self-supervised learning, we need to rethink problems from improving model interpretability [173], overcoming data bias [174], to detecting novel objects [175, 176].

Beyond representation learning, there are other challenges. First, how to improve perception accuracy to better identify object-of-target, actions, and attributes in the cluttered scene. Second, the appearance of an object might change dramatically in instructional videos (*e.g.*, raw meat *vs.* cooked meat, whole tomato *vs.* tomato slices), and therefore how to model an object in an action-conditioned and attribute-aware fashion. Third, despite that action/activity grounding in the temporal dimension [177, 178, 179] is relative well-studied in the video community compared to object grounding, a higher-granularity spatial-temporal action grounding lacks enough attention. Four, transfer learning from different domains and different views. For example, most of the online videos are shot in a third-person view but in robotics applications, robots usually require a first-person view.

Despite the primitiveness of the field, there are potential real-world applications. For instance, Microsoft has used Hololens and AR to teach factory workers new skills, with visually-grounded instructions and guidance.[2] The same technique could be applied for surgeries and better human-machine interactions. Also, with the recent progress on learning from demonstration [180], learning dynamics from video [181], and a lot of others on combining video understanding and robotics, sci-fi-like self-taught robot chefs might emerge in not-so-far future.

[2] https://news.microsoft.com/en-gb/features/the-power-of-mixed-reality-in-the-modern-workplace/

APPENDICES

APPENDIX A

Preliminary: Transformer Networks

In this Appendix, we introduce background on Transformer [82], which is the building block of our models in Chap. V, VII, and VIII.

Scaled Dot-Product Attention

We start by introducing the *scaled dot-product attention*, which is the foundation of transformer. Given a query $q_i \in \mathbb{R}^d$ from all T' queries, a set of keys $k_t \in \mathbb{R}^d$ and values $v_t \in \mathbb{R}^d$ where $t = 1, 2, ..., T$, the scaled dot-product attention outputs a weighted sum of values v_t, where the weights are determined by the dot-products of query q and keys k_t. In practice, we pack k_t and v_t into matricies $K = (k_1, ..., k_T)$ and $V = (v_1, ..., v_T)$, respectively. The attention output on query q is:

$$A(q_i, K, V) = V \frac{\exp\left\{K^T q_i / \sqrt{d}\right\}}{\sum_{t=1}^{T} \exp\{k_t^T q_i / \sqrt{d}\}} \tag{A.1}$$

The *multi-head attention* consists of H paralleled scaled dot-product attention layers called “head”, where each “head” is an independent dot-product attention. The attention output from multi-head attention is as below:

$$\text{MA}(q_i, K, V) = W^O \begin{pmatrix} \text{head}_1 \\ \dots \\ \text{head}_H \end{pmatrix} \tag{A.2}$$

$$\text{head}_j = A(W_j^q q_i, W_j^K K, W_j^V V) \tag{A.3}$$

where $W_j^q, W_j^K, W_j^V \in \mathbb{R}^{\frac{d}{H} \times d}$ are the independent head projection matrices, $j = 1, 2, ..., H$, and $W^O \in \mathbb{R}^{d \times d}$.

This formulation of attention is quite general, for example when the query is the hidden states from the decoder, and both the keys and values are all the encoder hidden states, it represents the common cross-module attention. *Self-attention* [82] is another case of multi-head attention where the queries, keys and values are all from the same hidden layer (see also in Fig. A.1). A walk-through example on self-attention could be found in this blog (section "Self-Attention in Detail").[1]

Transformer Networks

Now we are ready to introduce Transformer model, which is an encoder-decoder based model that is originally proposed for machine translation [82]. The building block for Transformer is multi-head attention and a pointwise feed-forward layer. The pointwise feed-forward layer takes the input from multi-head attention layer, and further transforms it through two linear projections with ReLU activation. The feed-forward layer can also be viewed as two convolution layers with kernel size one. The *encoder* and *decoder* of Transformer is composed by multiple such building blocks, and they have the same number of layers. The decoder from each layer takes input from the encoder of the same layer as well as the lower layer decoder output. Self-attention is applied to both encoder and

[1] http://jalammar.github.io/illustrated-transformer/

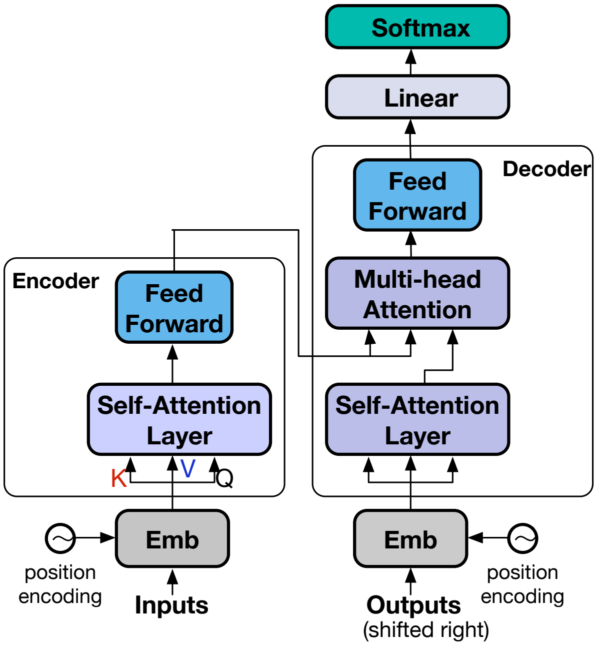


Figure A.1: Transformer with 1-layer encoder and 1-layer decoder.

decoder. Cross-module attention between encoder and decoder is also applied. Note that the (masked) self-attention layer in the decoder can only attend to the current and previous positions to preserve the auto-regressive property. The feed-forward block indicates a two-layer perceptron and the linear block indicates a linear layer. Residual connection [123] is applied to all input and output layers. Additionally, layer normalization [182] (LayerNorm) is applied to all layers. Fig. A.1 shows a one layered transformer.

Note that Transformers are a special form of Graph Neural Networks,[2] where the nodes are fully-connected (Transformer encoder) or mostly-connected (Transformer encoder). Transformer encoder is also the backbone of concurrent bidirectional language models, such as BERT [27]. Transformer decoder has been used intensively for language generation [155, 86].

Other learning resources. The blogs on “The Illustrated Transformer”[3] and “How Transformers Work”[4] are excellent supplementary read to this section.

[2] https://graphdeeplearning.github.io/post/transformers-are-gnns/

[3] http://jalammar.github.io/illustrated-transformer/

[4] https://towardsdatascience.com/transformers-141e32e69591

Implementations. Transformer comes with the official PyTorch library.[5] Other implementations include Hugging Face's Transformers [183].

[5] https://pytorch.org/docs/master/nn.html?highlight=transformertorch.nn.Transformer

APPENDIX B

Grounded Video Description

This Appendix provides additional details, evaluations, and implementation details on Chap. VI. In Sec. B, we provide more details on our dataset including the annotation interface and examples of our dataset, which are shown in Figs. B.1, B.2. In Sec. B, we provide additional results on our ActivityNet-Entities dataset. In Sec. B, we provide additional results on the Flickr30kEntities dataset. Finally in Sec. B, we provide more implementation details (*e.g.*, training details).

Dataset

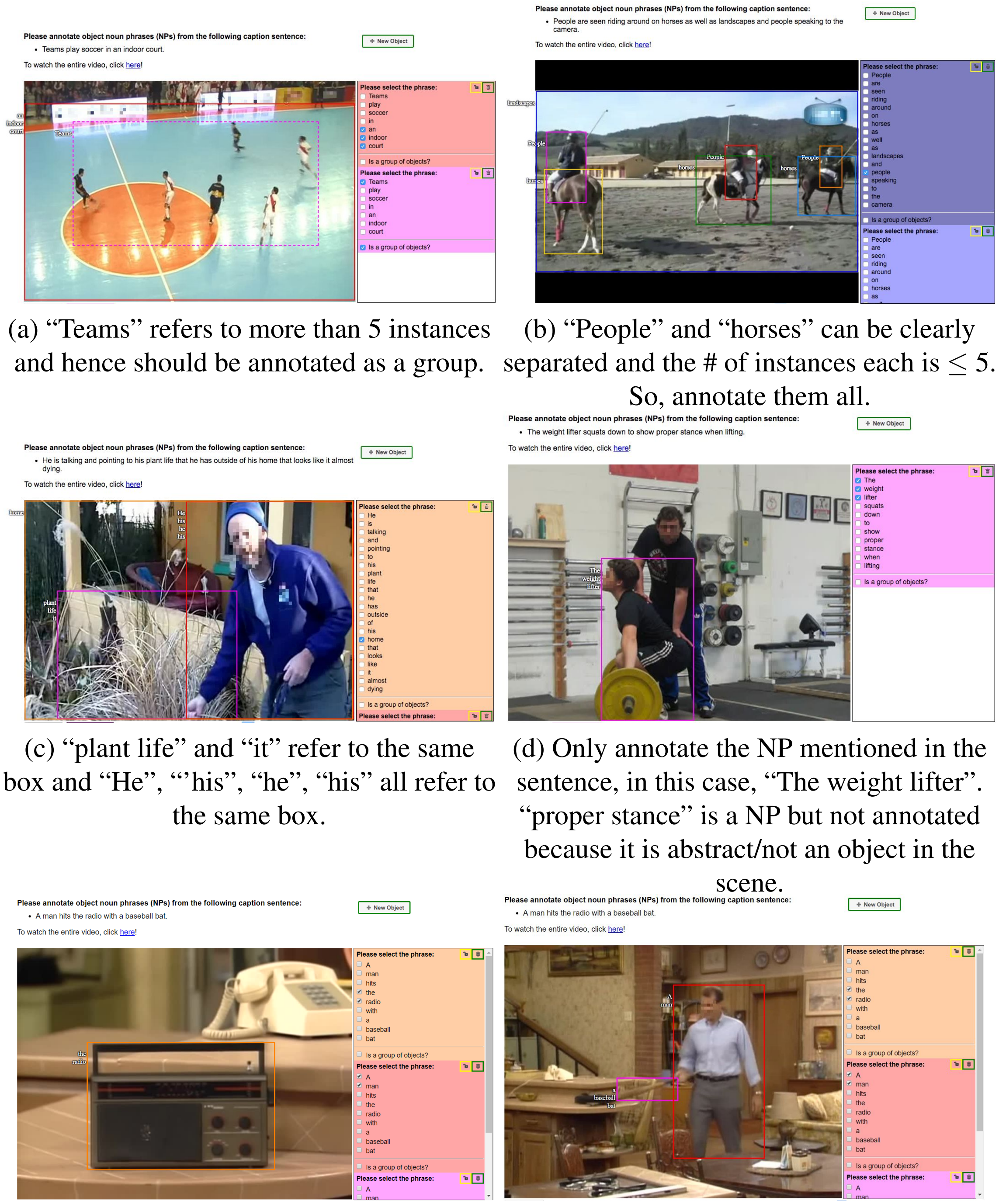
Definition of a noun phrase. Following the convention from Flickr30k Entities dataset [89], we define noun phrase as:

- short (avg. 2.23 words), non-recursive phrases (*e.g.*, the complex NP “the man in a white shirt with a heart” is split into three: “the man”, “a white shirt”, and “a heart”)
- refer to a specific region in the image so as to be annotated as a bounding box.
- could be
  - a single instance (*e.g.*, a cat),

  - multiple distinct instances (*e.g.*two men),
  - a group of instances (*e.g.*, a group of people),
  - a region or scene (*e.g.*, grass/field/kitchen/town),
  - a pronoun, *e.g.*, it, him, they.
- could include
  - adjectives (*e.g.*, a *white* shirt),
  - determiners (*e.g.*, *A* piece of exercise equipment),
  - prepositions (*e.g.*the woman *on the right*)
  - other noun phrases, if they refer to the identical bounding concept & bounding box (*e.g.*, a group of people, a shirt of red color)

Annotator instructions. Further instructions include:

- Each word from the caption can appear in at most one NP. “A man in a white shirt” and “a white shirt” should not be annotated at the same time.
- Annotate multiple boxes for the same NP if the NP refers to multiple instances.
  - If there are more than 5 instances/boxes (*e.g.*, six cats or many young children), mark all instances as a single box and mark as “a group of objects”.
  - Annotate 5 or fewer instances with a single box if the instances are difficult to separate, *e.g.*if they are strongly occluding each other.
- We don’t annotate a NP if it’s abstract or not presented in the scene (*e.g.*, “the camera” in “A man is speaking to the camera”)
- One box can correspond to multiple NPs in the sentence (*e.g.*, “the man” and “him”), *i.e.*, we annotate co-references within one sentence.



Please annotate object noun phrases (NPs) from the following caption sentence:

- Teams play soccer in an indoor court.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☐ Teams
☐ play
☐ soccer
☐ in
☑ an
☑ indoor
☑ court

☐ Is a group of objects?

Please select the phrase:
☑ Teams
☐ play
☐ soccer
☐ in
☐ an
☐ indoor
☐ court

☑ Is a group of objects?

(a) “Teams” refers to more than 5 instances and hence should be annotated as a group.

Please annotate object noun phrases (NPs) from the following caption sentence:

- People are seen riding around on horses as well as landscapes and people speaking to the camera.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☐ People
☐ are
☐ seen
☐ riding
☐ around
☐ on
☐ horses
☐ as
☐ well
☐ as
☐ landscapes
☐ and
☑ people
☐ speaking
☐ to
☐ the
☐ camera

☐ Is a group of objects?

Please select the phrase:
☐ People
☐ are
☐ seen
☐ riding
☐ around
☐ on
☐ horses
☐ as

(b) “People” and “horses” can be clearly separated and the # of instances each is ≤ 5. So, annotate them all.

Please annotate object noun phrases (NPs) from the following caption sentence:

- He is talking and pointing to his plant life that he has outside of his home that looks like it almost dying.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☐ He
☐ is
☐ talking
☐ and
☐ pointing
☐ to
☐ his
☐ plant
☐ life
☐ that
☐ he
☐ has
☐ outside
☐ of
☐ his
☑ home
☐ that
☐ looks
☐ like
☐ it
☐ almost
☐ dying

☐ Is a group of objects?

Please select the phrase:

(c) “plant life” and “it” refer to the same box and “He”, “’his”, “he”, “his” all refer to the same box.

Please annotate object noun phrases (NPs) from the following caption sentence:

- The weight lifter squats down to show proper stance when lifting.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☑ The
☑ weight
☑ lifter
☐ squats
☐ down
☐ to
☐ show
☐ proper
☐ stance
☐ when
☐ lifting

☐ Is a group of objects?

(d) Only annotate the NP mentioned in the sentence, in this case, “The weight lifter”. “proper stance” is a NP but not annotated because it is abstract/not an object in the scene.

Please annotate object noun phrases (NPs) from the following caption sentence:

- A man hits the radio with a baseball bat.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☐ A
☐ man
☐ hits
☑ the
☑ radio
☐ with
☐ a
☐ baseball
☐ bat

☐ Is a group of objects?

Please select the phrase:
☑ A
☑ man
☐ hits
☐ the
☐ radio
☐ with
☐ a
☐ baseball
☐ bat

☐ Is a group of objects?

Please select the phrase:
☐ A
☐ man

(e) Note that (e) and (f) refer to the same video segment. See the caption of (f) for more details.

Please annotate object noun phrases (NPs) from the following caption sentence:

- A man hits the radio with a baseball bat.

To watch the entire video, click here!

+ New Object

Please select the phrase:
☐ A
☐ man
☐ hits
☑ the
☑ radio
☐ with
☐ a
☐ baseball
☐ bat

☐ Is a group of objects?

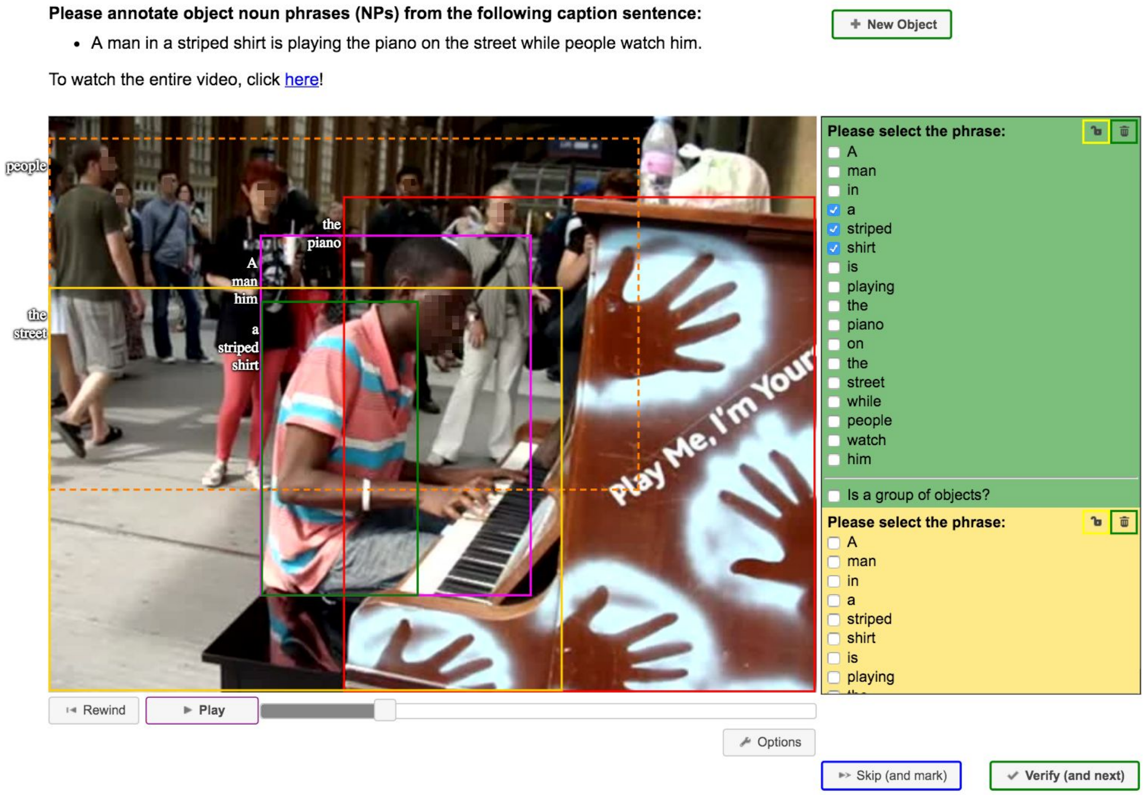
Please select the phrase:
☑ A
☑ man
☐ hits
☐ the
☐ radio
☐ with
☐ a
☐ baseball
☐ bat

☐ Is a group of objects?

Please select the phrase:
☐ A
☐ man

(f) “The radio” is annotated in a different frame as “a man” and “a baseball bat”, since it cannot be clearly observed in the same frame.

Figure B.1: Examples of our ActivityNet-Entities annotations in the annotation interface.



Please annotate object noun phrases (NPs) from the following caption sentence:

- A man in a striped shirt is playing the piano on the street while people watch him.

To watch the entire video, click here!

Please select the phrase:

□ A
□ man
□ in
☑ a
☑ striped
☑ shirt
□ is
□ playing
□ the
□ piano
□ on
□ the
□ street
□ while
□ people
□ watch
□ him

□ Is a group of objects?

Please select the phrase:

□ A
□ man
□ in
□ a
□ striped
□ shirt
□ is
□ playing

Figure B.2: A screen shot of our annotation interface. The "verify (and next)" button indicates the annotation is under the verification mode, where the initial annotation is loaded and could be revised.

See Fig. B.1 for more examples.

Annotation interface. We show a screen shot of the interface in Fig. B.2.

Validation process. We deployed a rigid quality control process during annotations. We were in daily contact with the annotators, encouraged them to flag all examples that were unclear and inspected a sample of the annotations daily, providing them with feedback on possible spotted annotation errors or guideline violations. We also had a post-annotation verification process where all the annotations are verified by human annotators.

List of objects. Tab. B.5 lists all the 432 object classes which we use in our approach. We threshold at 50 occurrences. Note that the annotations in ActivityNet-Entities also contain the full noun phrases w/o thresholds.

	F1$_{all}$		F1$_{loc}$	
Method	Precision	Recall	Precision	Recall
Unsup. (w/o SelfAttn)	3.76	3.63	12.6	12.9
Unsup.	0.28	0.27	1.13	1.13
Sup. Attn.	6.71	6.73	22.6	22.8
Sup. Grd.	6.25	5.84	21.2	21.2
Sup. Cls.	0.40	0.32	1.39	1.47
Sup. Attn.+Grd.	7.07	6.54	23.0	23.0
Sup. Attn.+Cls.	7.29	6.94	24.0	24.1
Sup. Grd. +Cls.	4.94	4.64	17.7	17.6
Sup. Attn.+Grd.+Cls.	7.42	6.81	23.7	23.9

Table B.1: Attention precision and recall on generated sentences on ANet-Entities val set. All values are in %.

	F1$_{all}$		F1$_{loc}$	
Method	Precision	Recall	Precision	Recall
Unsup. (w/o SelfAttn)	3.62	3.85	11.7	11.8
Sup. Attn.+Cls.	7.64	7.55	25.1	24.8

Table B.2: Attention precision and recall on generated sentences on ANet-Entities test set. All values are in %.

Results on ActivityNet-Entities

We include here the precision and recall associated with $F1_{all}$ and $F1_{loc}$ (see Tabs. B.1, B.2).

Results on Flickr30k Entities

We include here the precision and recall associated with $F1_{all}$ and $F1_{loc}$ (see Tabs. B.3, B.4).

Implementation Details

Region proposal and feature. We uniformly sample 10 frames per video segment (an event in ANet-Entities) and extract region features. For each frame, we use a Faster RCNN

	F1_{all}		F1_{loc}	
Method	Precision	Recall	Precision	Recall
Unsup. (w/o SelfAttn)	4.08	4.89	12.8	12.8
Unsup.	0.75	0.87	2.08	2.10
Sup. Attn.	7.46	8.83	22.4	22.5
Sup. Grd.	6.90	8.43	21.0	21.0
Sup. Cls. (w/o SelfAttn)	3.70	4.66	11.4	11.5
Sup. Attn.+Grd.	7.93	9.45	23.7	23.6
Sup. Attn.+Cls.	7.61	9.25	23.2	23.1
Sup. Grd. +Cls.	4.70	5.83	13.7	13.7
Sup. Attn.+Grd.+Cls.	7.56	9.20	23.2	23.2

Table B.3: Attention precision and recall on generated sentences on Flickr30k Entities val set. All values are in %.

	F1_{all}		F1_{loc}	
Method	Precision	Recall	Precision	Recall
BUTD [73]	4.07	5.13	13.1	13.0
Our Unsup. (w/o SelfAttn)	3.44	4.47	11.6	11.8
Our Sup. Attn.+Grd.+Cls.	6.91	8.33	22.2	22.2

Table B.4: Attention precision and recall on generated sentences on Flickr30k Entities test set. All values are in %.

model [47] with a ResNeXt-101 FPN backbone [150] for region proposal and feature extraction. The Faster RCNN model is pretrained on the Visual Genonme dataset [147]. We use the same train-val-test split pre-processed by Anderson *et al.* [73] for joint object detection (1600 classes) and attribute classification. In order for a proposal to be considered valid, its confident score has to be greater than 0.2. And we limit the number of regions per image to a fixed 100 [158]. We take the output of the fc6 layer as the feature representation for each region, and fine-tune the fc7 layer and object classifiers with 0.1× learning rate during model training.

Training details. We optimize the training with Adam (params: 0.9, 0.999). The learning rate is set to 5e-4 in general and to 5e-5 for fine-tuning, *i.e.*, fc7 layer and object classifiers, decayed by 0.8 every 3 epochs. The batch size is 240 for all the methods. We implement the model in PyTorch based on NBT[1] and train on 8x V100 GPUs. The training is limited to 40 epochs and the model with the best validation CIDEr score is selected for testing.

[1] https://github.com/jiasenlu/NeuralBabyTalk

__background__	egg	nail	kid	snowboard	hoop	roller	pasta
bagpipe	stilt	metal	butter	cheerleader	puck	kitchen	stage
coach	paper	dog	surfboard	landscape	scene	guitar	trophy
bull	dough	tooth	object	eye	scissors	grass	stone
rod	costume	pipe	ocean	sweater	ring	drum	swimmer
disc	oven	shop	person	camera	city	accordion	stand
dish	braid	shot	edge	vehicle	horse	ramp	road
chair	pinata	kite	bottle	raft	basketball	bridge	swimming
carpet	bunch	text	camel	themselves	monkey	wall	image
animal	group	barbell	photo	calf	top	soap	playground
gymnast	harmonica	biker	polish	teen	paint	pot	brush
mower	platform	shoe	cup	door	leash	pole	female
bike	window	ground	sky	plant	store	dancer	log
curler	soccer	tire	lake	glass	beard	table	area
ingredient	coffee	title	bench	flag	gear	boat	tennis
woman	someone	winner	color	adult	shorts	bathroom	lot
string	sword	bush	pile	baby	gym	teammate	suit
wave	food	wood	location	hole	wax	instrument	opponent
gun	material	tape	ski	circle	park	blower	head
item	number	hockey	skier	word	part	beer	himself
sand	band	piano	couple	room	herself	stadium	t-shirt
saxophone	they	goalie	dart	car	chef	board	cloth
team	foot	pumpkin	sumo	athlete	target	website	line
sidewalk	silver	hip	game	blade	instruction	arena	ear
razor	bread	plate	dryer	roof	tree	referee	he
clothes	name	cube	background	cat	bed	fire	hair
bicycle	slide	beam	vacuum	wrestler	friend	worker	slope
fence	arrow	hedge	judge	closing	iron	child	potato
sign	rock	bat	lady	male	coat	bmx	bucket
jump	side	bar	furniture	dress	scuba	instructor	cake
street	everyone	artist	shoulder	court	rag	tank	piece
video	weight	bag	towel	goal	clip	hat	pin
paddle	series	she	gift	clothing	runner	rope	intro
uniform	fish	river	javelin	machine	mountain	balance	home
supplies	gymnasium	view	glove	rubik	microphone	canoe	ax
net	logo	set	rider	tile	angle	it	face
exercise	girl	frame	audience	toddler	snow	surface	pit
body	living	individual	crowd	beach	couch	player	cream
trampoline	flower	parking	people	product	equipment	cone	lemon
leg	container	racket	back	sandwich	chest	violin	floor
surfer	house	close	sponge	mat	contact	helmet	fencing
water	hill	arm	mirror	tattoo	lip	shirt	field
studio	wallpaper	reporter	diving	ladder	tool	paw	other
sink	dirt	its	slice	bumper	spectator	bowl	oar
path	toy	score	leaf	end	track	member	picture
box	cookie	finger	bottom	baton	flute	belly	frisbee
boy	guy	teens	tube	man	cigarette	vegetable	lens
stair	card	pants	ice	tomato	mouth	pan	pool
bow	yard	opening	skateboarder	neck	letter	wheel	building
credit	skateboard	screen	christmas	liquid	darts	ball	lane
smoke	thing	outfit	knife	light	pair	drink	phone
trainer	swing	toothbrush	hose	counter	knee	hand	mask
shovel	castle	news	bowling	volleyball	class	fruit	jacket
kayak	cheese	tub	diver	truck	lawn	student	stick

Table B.5: List of objects in ActivityNet-Entities, including the “__background__” class.

APPENDIX C

Open Source, Workshops, and Challenges

In this Appendix, we listed our open-sourced projects, organized major workshops and challenges.

Open Source

- YouCook2 and YouCook2-BoundingBox dataset (Chap. III):
  `http://youcook2.eecs.umich.edu/`
- ActivityNet-Entities dataset (Chap. III):
  `https://github.com/facebookresearch/ActivityNet-Entities`
- PyTorch Implementation of Vision-Language Pre-training (Chap. VIII):
  `https://github.com/LuoweiZhou/VLP`
- PyTorch Implementation of Grounded Video Description (Chap. VI):
  `https://github.com/facebookresearch/grounded-video-description`
- PyTorch Implementation of Dense Video Description (Chap. V):
  `https://github.com/salesforce/densecap`

- PyTorch Implementation of Weakly-Supervised Object Grounding (Chap. VII):

  `https://github.com/MichiganCOG/Video-Grounding-from-Text`

- Torch Implementation of ProcNets (Chap. IV):

  `https://github.com/LuoweiZhou/ProcNets-YouCook2`

Workshops and Challenges

- Co-organizer, CVPR 2018 Workshop on Fine-grained Instructional Video undERstanding (FIVER):

  `http://fiver.eecs.umich.edu/`

- Co-organizer, Challenge on ActivityNet-Entities Object Localization (Grounding):

  `http://activity-net.org/challenges/2020/tasks/guest_anet_eol.html`, a part of the International Challenge on Activity Recognition (ActivityNet) at CVPR 2020.

- Program Committee, CVPR 2020 Workshop on Learning from Instructional Videos (WLIV):

  `https://sites.google.com/view/wliv20/home`

- Program Committee, ECCV 2018 Workshop on Shortcomings in Vision and Language (SiVL):

  `https://sites.google.com/view/sivl`

- Program Committee, NAACL 2018 Workshop on Storytelling:

  `http://www.visionandlanguage.net/workshop2018/index.html`

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 706–715, 2017.

[2] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8739–8748, 2018.

[3] Masataka Yamaguchi, Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Spatio-temporal person retrieval via natural language queries. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1453–1462, 2017.

[4] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–970, 2015.

[5] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph based video segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[6] Chenliang Xu, Caiming Xiong, and Jason J Corso. Streaming hierarchical video segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 626–639. Springer, 2012.

[7] Brent A Griffin and Jason J Corso. Bubblenets: Learning to select the guidance frame in video object segmentation by deep sorting frames. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8914–8923, 2019.

[8] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. 1991.

[9] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2544–2550. IEEE, 2010.

[10] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4293–4302, 2016.

[11] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4282–4291, 2019.

[12] Ivan Laptev. On space-time interest points. *International Journal of Computer Vision (IJCV)*, 64(2-3):107–123, 2005.

[13] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2556–2563. IEEE, 2011.

[14] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497, 2015.

[15] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733. IEEE, 2017.

[16] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] Brent Griffin, Victoria Florence, and Jason J Corso. Video object segmentation-based visual servo control and object depth estimation on a mobile robot. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1647–1657, 2020.

[18] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[19] Atousa Torabi, Christopher Pal, Hugo Larochelle, and Aaron Courville. Using descriptive video services to create a large data source for video annotation research. *arXiv preprint arXiv:1503.01070*, 2015.

[20] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5288–5296, 2016.

[21] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2018.

[22] Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1207–1216, 2019.

[23] Ramon Sanabria, Ozan Caglayan, Shruti Palaskar, Desmond Elliott, Loïc Barrault, Lucia Specia, and Florian Metze. How2: a large-scale dataset for multimodal language understanding. *arXiv preprint arXiv:1811.00347*, 2018.

[24] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2630–2640, 2019.

[25] Luowei Zhou, Nathan Louis, and Jason J Corso. Weakly-supervised video object grounding from text by loss weighting and object interaction. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[26] Luowei Zhou, Yannis Kalantidis, Xinlei Chen, Jason J Corso, and Marcus Rohrbach. Grounded video description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6578–6587, 2019.

[27] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[28] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 780–787, 2014.

[29] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Josef Sivic, Ivan Laptev, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4575–4583, 2016.

[30] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13–23, 2019.

[31] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*, 2019.

[32] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[33] Chris Alberti, Jeffrey Ling, Michael Collins, and David Reitter. Fusion of detected objects in text for visual question answering. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[34] Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2020.

[35] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[36] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[37] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Contrastive bidirectional transformer for temporal representation learning. *arXiv preprint arXiv:1906.05743*, 2019.

[38] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2020.

[39] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nick Johnston, Andrew Rabinovich, and Kevin Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2015.

[40] Piotr Bojanowski, Rémi Lajugie, Edouard Grave, Francis Bach, Ivan Laptev, Jean Ponce, and Cordelia Schmid. Weakly-supervised alignment of video with text. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4462–4470, 2015.

[41] Ozan Sener, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4480–4488, 2015.

[42] Hilde Kuehne, Juergen Gall, and Thomas Serre. An end-to-end generative framework for video segmentation and recognition. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8, 2016.

[43] Hilde Kuehne, Alexander Richard, and Juergen Gall. Weakly supervised learning of actions from transcripts. *Computer Vision and Image Understanding (CVIU)*, 2017.

[44] De-An Huang, Li Fei-Fei, and Juan Carlos Niebles. Connectionist temporal modeling for weakly supervised action labeling. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 137–153. Springer, 2016.

[45] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 628–643. Springer, 2014.

[46] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1049–1058, 2016.

[47] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015.

[48] Jiyang Gao, Zhenheng Yang, Chen Sun, Kan Chen, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[49] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 768–784. Springer, 2016.

[50] Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2911–2920, 2017.

[51] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *European conference on computer vision*, pages 15–29. Springer, 2010.

[52] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(12):2891–2903, 2013.

[53] Muhammad Usman Ghani Khan, Lei Zhang, and Yoshihiko Gotoh. Towards coherent natural language description of video streams. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 664–671. IEEE, 2011.

[54] Andrei Barbu, Alexander Bridge, Zachary Burchill, Dan Coroian, Sven Dickinson, Sanja Fidler, Aaron Michaux, Sam Mussman, Siddharth Narayanaswamy, Dhaval Salvi, et al. Video in sentences out. *arXiv preprint arXiv:1204.2742*, 2012.

[55] Niveda Krishnamoorthy, Girish Malkarnenkar, Raymond Mooney, Kate Saenko, and Sergio Guadarrama. Generating natural-language video descriptions using text-mined knowledge. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[56] Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2634–2641, 2013.

[57] Haonan Yu and Jeffrey Mark Siskind. Grounded language learning from video described with sentences. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 53–63, 2013.

[58] Shaohua Yang, Qiaozi Gao, Changsong Liu, Caiming Xiong, Song-Chun Zhu, and Joyce Chai. Grounded semantic role labeling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 149–159, 2016.

[59] Ran Xu, Caiming Xiong, Wei Chen, and Jason J Corso. Jointly modeling deep video and compositional text to bridge vision and language in a unified framework. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[60] Ted Pedersen, Siddharth Patwardhan, Jason Michelizzi, et al. Wordnet:: Similarity-measuring the relatedness of concepts. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, volume 4, pages 25–29, 2004.

[61] Margaret Mitchell, Xufeng Han, Jesse Dodge, Alyssa Mensch, Amit Goyal, Alex Berg, Kota Yamaguchi, Tamara Berg, Karl Stratos, and Hal Daumé III. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 747–756. Association for Computational Linguistics, 2012.

[62] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[63] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4507–4515, 2015.

[64] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2048–2057, 2015.

[65] Yehao Li, Ting Yao, Yingwei Pan, Hongyang Chao, and Tao Mei. Jointly localizing and describing events for dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7492–7500, 2018.

[66] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 22–29, 2017.

[67] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4651–4659, 2016.

[68] Luowei Zhou, Chenliang Xu, Parker Koch, and Jason J Corso. Watch what you just said: Image captioning with text-conditional attention. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pages 305–313. ACM, 2017.

[69] Yingwei Pan, Ting Yao, Houqiang Li, and Tao Mei. Video captioning with transferred semantic attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 3, 2017.

[70] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988. IEEE, 2017.

[71] Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. Attend and interact: Higher-order object interactions for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6790–6800, 2018.

[72] Anna Rohrbach, Marcus Rohrbach, Siyu Tang, Seong Joon Oh, and Bernt Schiele. Generating descriptions with grounded and co-referenced people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[73] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086, 2018.

[74] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Neural baby talk. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7219–7228, 2018.

[75] Mihai Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Spatio-temporal attention models for grounded video captioning. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 104–119, 2016.

[76] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 684–699, 2018.

[77] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10685–10694, 2019.

[78] Abhishek Das, Harsh Agrawal, Larry Zitnick, Devi Parikh, and Dhruv Batra. Human attention in visual question answering: Do humans and deep networks look at the same regions? *Computer Vision and Image Understanding (CVIU)*, 163:90–100, 2017.

[79] Yundong Zhang, Juan Carlos Niebles, and Alvaro Soto. Interpretable visual question answering by visual grounding from attention supervision mining. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 349–357. IEEE, 2019.

[80] Chenxi Liu, Junhua Mao, Fei Sha, and Alan L Yuille. Attention correctness in neural image captioning. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 4176–4182, 2017.

[81] Youngjae Yu, Jongwook Choi, Yeonhwa Kim, Kyung Yoo, Sang-Hun Lee, and Gunhee Kim. Supervising neural attention models for video captioning by human gaze data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2680–29, 2017.

[82] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[83] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4565–4574, 2016.

[84] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4584–4593, 2016.

[85] Fabian Caba Heilbron, Wayner Barrios, Victor Escorcia, and Bernard Ghanem. Scc: Semantic context cascade for efficient action detection.

[86] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13042–13054, 2019.

[87] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[88] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[89] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2641–2649, 2015.

[90] Bryan A Plummer, Paige Kordas, M Hadi Kiapour, Shuai Zheng, Robinson Piramuthu, and Svetlana Lazebnik. Conditional image-text embedding networks. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[91] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1307–1315, 2018.

[92] Zhenyang Li, Ran Tao, Efstratios Gavves, Cees GM Snoek, and Arnold WM Smeulders. Tracking by natural language specification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6495–6503, 2017.

[93] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with language referring expressions. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 123–141. Springer, 2018.

[94] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 817–834. Springer, 2016.

[95] Fanyi Xiao, Leonid Sigal, and Yong Jae Lee. Weakly-supervised visual grounding of phrases with linguistic structures. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[96] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1889–1897, 2014.

[97] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137, 2015.

[98] De-An Huang, Shyamal Buch, Lucio Dery, Animesh Garg, Li Fei-Fei, and Juan Carlos Niebles. Finding “it”: Weakly-supervised reference-aware visual grounding in instructional video. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[99] Haonan Yu and Jeffrey Mark Siskind. Sentence directed video object codiscovery. *International Journal of Computer Vision (IJCV)*, 124(3):312–334, 2017.

[100] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

[101] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1564–1574, 2018.

[102] Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven CH Hoi, Xiaogang Wang, and Hongsheng Li. Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6639–6648, 2019.

[103] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and Yang: Balancing and answering binary visual questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[104] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6904–6913, 2017.

[105] Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don’t just assume; look and answer: Overcoming priors for visual question answering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4971–4980, 2017.

[106] Mike Lewis and Angela Fan. Generative question answering: Learning to answer the whole question. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[107] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1194–1201, 2012.

[108] Sebastian Stein and Stephen J McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 729–738. ACM, 2013.

[109] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[110] Christopher Baldassano, Janice Chen, Asieh Zadbood, Jonathan W Pillow, Uri Hasson, and Kenneth A Norman. Discovering event structure in continuous narrative perception and memory. *Neuron*, 95(3):709–721, 2017.

[111] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision (IJCV)*, 101(1):184–204, 2013.

[112] Chunhui Gu, Chen Sun, Sudheendra Vijayanarasimhan, Caroline Pantofaru, David A Ross, George Toderici, Yeqing Li, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2018.

[113] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[114] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[115] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2:67–78, 2014.

[116] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2625–2634, 2015.

[117] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 20–36. Springer, 2016.

[118] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1961–1970, 2016.

[119] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2678–2687, 2016.

[120] Wei Chen, Caiming Xiong, Ran Xu, and Jason J Corso. Actionness ranking with lattice conditional ordinal random fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 748–755, 2014.

[121] Muhannad Al-Omari, Paul Duckworth, David C Hogg, and Anthony G Cohn. Natural language acquisition and grounding for embodied robotic systems. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 4349–4356, 2017.

[122] Haonan Yu and Jeffrey Mark Siskind. Learning to describe video with weak supervision by exploiting negative sentential information. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 3855–3863, 2015.

[123] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[124] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

[125] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 766–782. Springer, 2016.

[126] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.

[127] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[128] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[129] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[130] Bernard Ghanem, Juan Carlos Niebles, Cees Snoek, Fabian Caba Heilbron, Humam Alwassel, Ranjay Khrisna, Victor Escorcia, Kenji Hata, and Shyamal Buch. Activitynet challenge 2017 summary. *arXiv preprint arXiv:1710.08011*, 2017.

[131] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*.

[132] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[133] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[134] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[135] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[136] Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1019–1027, 2016.

[137] Yuanjun Xiong, Limin Wang, Zhe Wang, Bowen Zhang, Hang Song, Wei Li, Dahua Lin, Yu Qiao, Luc Van Gool, and Xiaoou Tang. Cuhk & ethz & siat submission to activitynet challenge 2016. *arXiv preprint arXiv:1608.00797*, 2016.

[138] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223. Springer, 2007.

[139] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2015.

[140] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence-video to text. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4534–4542, 2015.

[141] Haonan Yu, N Siddharth, Andrei Barbu, and Jeffrey Mark Siskind. A compositional framework for grounding language inference, generation, and acquisition in video. *Journal of Artificial Intelligence Research (JAIR)*, 52:601–713, 2015.

[142] Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 771–787, 2018.

[143] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object hallucination in image captioning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4035–4045, 2018.

[144] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh. Vizwiz: Nearly real-time answers to visual questions. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 333–342, New York, NY, USA, 2010. ACM.

[145] Anna Rohrbach, Atousa Torabi, Marcus Rohrbach, Niket Tandon, Chris Pal, Hugo Larochelle, Aaron Courville, and Bernt Schiele. Movie description. *International Journal of Computer Vision (IJCV)*, 2017.

[146] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[147] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision (IJCV)*, 123(1):32–73, 2017.

[148] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[149] Yilei Xiong, Bo Dai, and Dahua Lin. Move forward and tell: A progressive generator of video descriptions. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[150] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995. IEEE, 2017.

[151] Alessandro Prest, Christian Leistner, Javier Civera, Cordelia Schmid, and Vittorio Ferrari. Learning object class detectors from weakly annotated video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3282–3289. IEEE, 2012.

[152] Suha Kwak, Minsu Cho, Ivan Laptev, Jean Ponce, and Cordelia Schmid. Unsupervised object discovery and tracking in video collections. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3173–3181. IEEE, 2015.

[153] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[154] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[155] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

[156] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018.

[157] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[158] Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0. 1: the winning entry to the vqa challenge 2018. *arXiv preprint arXiv:1807.09956*, 2018.

[159] Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. Attention on attention for image captioning. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[160] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7008–7024, 2017.

[161] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[162] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 173–182, 2016.

[163] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[164] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4918–4927, 2019.

[165] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[166] Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R Bowman. Do attention heads in bert track syntactic dependencies? *arXiv preprint arXiv:1911.12246*, 2019.

[167] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *arXiv preprint arXiv:2002.12327*, 2020.

[168] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 69–84. Springer, 2016.

[169] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 649–666. Springer, 2016.

[170] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[171] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. M ˆ2: Meshed-memory transformer for image captioning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[172] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[173] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 269–286. Springer, 2018.

[174] Lisa Anne Hendricks, Kaylee Burns, Kate Saenko, Trevor Darrell, and Anna Rohrbach. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 793–811. Springer, 2018.

[175] Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, and Trevor Darrell. Deep compositional captioning: Describing novel object categories without paired training data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–10, 2016.

[176] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. nocaps: novel object captioning at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8948–8957, 2019.

[177] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics (TACL)*, 1:25–36, 2013.

[178] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5803–5812, 2017.

[179] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with temporal language. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[180] De-An Huang, Suraj Nair, Danfei Xu, Yuke Zhu, Animesh Garg, Li Fei-Fei, Silvio Savarese, and Juan Carlos Niebles. Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8565–8574, 2019.

[181] Angjoo Kanazawa, Jason Y Zhang, Panna Felsen, and Jitendra Malik. Learning 3d human dynamics from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5614–5623, 2019.

[182] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[183] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.