QUESTION ANSWERING, GROUNDING, AND GENERATION FOR VISION AND LANGUAGE

Licheng Yu

A thesis submitted to the faculty at the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2019

Approved by:

Tamara L. Berg

Mohit Bansal

Alexander C. Berg

Ron Alterovitz

Dhruv Batra

ABSTRACT

Licheng Yu: Question Answering, Grounding, and Generation for Vision and Language
(Under the direction of Tamara L. Berg)

One ultimate goal of AI is to develop an artificial intelligent (AI) system that can communicate with people in a natural way. Such communication includes but is not limited to asking we humans questions, answering our questions, conducting dialogue with human beings, and performing some actions to better serve people. Imagine in the future where the service robot is everywhere, and we could ask our home robot to "grab me the red cup on the table." To perform this command, the AI system needs to understand this spoken English sentence, perceive the visual world, navigate to the right place "table", recognize the right object "the red cup", then grab it and finally return it back to the commander. Just for this single command, it already involves many techniques, such as speech recognition, language understanding, scene understanding, embodied navigation, object recognition, pose estimation, robot manipulation, etc. Each of these techniques are not well solved yet, but we are on a rapid way toward the success. This thesis is in advancing our knowledge to explore various connections between vision, language and even beyond to push forward this ultimate goal. We study 3 popular vision and language tasks, including visual question answering, language grounding, and image-to-text language generation. Inside each, we will introduce our proposed novel task, accompanied with high-quality dataset and well-performing data-driven approaches.

Specifically, we first introduce Visual Madlibs for image-based and region-based question answering. Then we introduce referring expressions, where we study both referring expression comprehension and generation, covering both language grounding and generation. Next, we study album summarization, which not only selects the key photos inside an album but also generates a natural language story describing the whole album. Last but not least, we describe multi-target

embodied question answering, a task that is even closer to our ultimate goal that requires both language understanding and navigation ability from the AI system.

ACKNOWLEDGEMENTS

I always talk with my families and friends how lucky I am as a PhD student, because I met a great professor who is my advisor - Tamara L. Berg. I really want to thank Tamara for giving me the chance working with her. She has so many brilliant ideas, teaches me with patience, and encourages me when I get stuck in research, for many years. I feel proud to be her student.

I also want to thank Mohit Bansal, who contributed in shaping me as a better research. We have been working together for 3 years and I learned a lot from him within this period. Mohit also helps me building academia connections with other researchers and gives me career advice. I really appreciate all kinds of help from him.

Also many thanks to the professors and teachers during my PhD study. Special thanks to Alex Berg who collaborated with me in the very beginning of my PhD study. I quite like Alex's jokes which might not be easy to understand in the beginning but would be very interesting after thinking a while. Chatting with him is always fun and helpful to me. I also want to thank Vladimir Jojic for so well-prepared Machine Learning class. Thanks to Marc Niethammer for the useful optimization course. Thanks to Shahriar Nirjon for the hands-on teaching of mobile computing. I coded my first AI-based Go-Bang Android App in his course, which is very cool. Thanks to Ron Alterovitz for teaching me in the Robotics class. I learned a lot from this class and made a cool project in this class - Speech-driven Manipulation Robot, which I feel quite proud of and always demonstrated in my talk. Also thanks Ron for being my committee member. Thanks to Jan-Michael Frahm for teaching me 3D vision, broadening me the horizon of understanding computer vision.

I also want to thank all my outside mentors from industry. I am thankful to my collaborators at eBay Research Labs, especially to Robinson Piramuthu and Hadi Kiapour. This is my first summer (in 2016) internship and thanks to them for letting me touch the industry-scale data and

real problem. I also want to thank Zhe Lin, Xiaohui Shen, and Jimei Yang for tutoring me at Adobe Research in 2017 summer. We worked together to make a breakthrough on the referential segmentation project with a huge performance boost. I also want to thank my Facebook AI Research collaborators - Dhruv Batra, Xinlei Chen and Georgia Gkioxari. Dhruv helped me define my summer project in a very clear way and tutored me working toward the final goal without detouring. Xinlei discussed with me delving into each detail of my project. Georgia helped me a lot on the paper writing.

Thanks also to all my labmates at UNC, who have made my life easier in my PhD study, including: Jie Lei, Hao Tan, Qiuyu Xiao, Hongkun Ge, Dong Nie, Zhen Wei, Yipin Zhou, True Price, Wei Liu, Xufeng Han, Patrick Poirson, Phil Ammirato, Eunbyung Park, Sirion Vittayakorn, Cheng-Yang Fu, Hadi Kiapour, Dinghuang Ji, Jared Heinly, Ke Wang, Akash Bapat, Yixin Nie, Mengyu Fu, Haonan Chen and Hao Jiang. Special thanks to Shan Yang, who has always been supporting me in every aspect of my life. I could not get through all difficulties without her accompany during my PhD study.

I also want to thank my external friends: Xin Wang, Ronghang Hu and Bichen Wu from UC Berkeley; Zhe Gan and Hongteng Xu from Duke; Zhuoyuan Chen, Kan Chen, and Yuxin Wu from Facebook; Yuting Zhang from Amazon; Chen Sun, Fan Yang, and Lu Jiang from Google; Yuke Zhu from Stanford; and my Shanghai Jiaotong University alumni - Yin Li, Yi Xu, Chao Ma, Xiaokang Yang, Dian Li, Feiya Chen, Eryue Chen, Ruotian Luo, Zhaowen Wang, Bo Xiao, Liwei Wang, Junchi Yan, Jinghui Zhang; and my all-time childhood friends - Minyu Liu, Chen Shen, Zheming Jin, Zhichun Xiong, and Yiqin Gong who teaches me “Learn to Fail or Fail to Learn.”

Thanks to my parents and my grandparents. I am here getting my PhD degree due to their constant strong support. Especially, I want to thank my grandfather, who was the first-generation computer science professional in China, back to 1970s. He advised me to be a good researcher, as well as a good person.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

One ultimate goal of artificial intelligence (AI) is to build an AI agent that can listen, see, talk, and even act, to better serve us. Perhaps one bright future application is home service robot. With such robot, we could ask it to perform a job that is distant or repetitive. For example, we could ask such robot to "grab me the green bottle on the table." Ideally, the robot follows this command and perform this task in a fully automatic manner. Such application could be potentially very helpful for those impaired or old people. However, just for this single command, it already involves quite a few cutting-edge techniques, including speech recognition to convert this command into readable text, language understanding so that the robot can do semantic role labeling and know the required action is "grab", the source "bottle" and the target is "me", embodied navigation to move to the table, object recognition to find "the green bottle", pose estimation to get the bottle's 3-D position, motion planning to grab it, and finally return it back to "me". Each of the above sub-tasks is challenging and has not been well addressed yet. In this thesis, we focus on the initialization stage of this example, which is the vision and language understanding for human-robot interaction.

The early vision and language problem was proposed in Farhadi et al. (2010) and Kulkarni et al. (2013) for image captioning task back to 2010. These datasets are typically of fairly small size Farhadi et al. (2010) and the sentences being generated are hand-crafted with predefined templates. Since AlexNet Krizhevsky et al. (2012) was introduced in 2012, we are witnessing a breakthrough in the field of both computer vision and natural language processing. Such success is mainly due to three aspects - bigger data, stronger computation power, and deep learning algorithms. Since then, we have seen the rise of multi-modal tasks combining vision and language. For example, Vinyals et al. (2015b) generates human-like captions from images, Anderson et al.

(2018a) automatically answers free-form questions based on images, Yu et al. (2018) localizes objects mentioned by a human description, etc. We also extend the study to understanding video and language, e.g., Krishna et al. (2017) detects dense events and generates captions for each of them from a video, Lei et al. (2018) does question-answering based on video clips and subtitles, Hendricks et al. (2017) localizes key moments based on natural descriptions, etc.

My PhD starts from 2014, which is roughly the start of applying deep learning onto vision and language, and thus most of my work lies in large-scale multi-modality tasks that are addressed with deep learning approaches. Specifically, my collaborators and I studied visual question answering, language grounding, image-to-text generation, and embodied AI. We proposed novel task in each of these research topics and achieved state-of-the-art performance on them.

In this thesis, we first introduce visual question answering. We collected a dataset called Visual Madlibs Yu et al. (2015). The Visual Madlibs consists of 360,001 focused natural language descriptions for 10,738 images. It is collected using automatically produced fill-in-the-blank templates designed to gather targeted descriptions about: people and objects, their appearances, activities, and interactions, as well as inferences about the general scene or its broader context. We provide several analyses of our Visual Madlibs dataset and demonstrate its applicability to two new description generation tasks: focused description generation, and multiple-choice question-answering for images. Respectively, we use joint-embedding method for the multiple-choice task and deep-learning based method for image description generation.

We then study referring expressions. Referring expressions are natural language constructions used to identify particular objects within a scene. We collected two datasets based on MS COCO Lin et al. (2014) images - RefCOCO and RefCOCO+ Yu et al. (2016b). Based on the datasets, we explore generating and comprehending natural language referring expressions for objects in images, where the generation task is to generate a sentence describing a target object and the comprehension task is to localize the object given a natural description. Then we proposed three works to address the two tasks. The first work Yu et al. (2016b) models visual and language context for referring expression comprehension and generation, the second work Yu

et al. (2017b) unifies the role of speaker (for generation) and listener (for comprehension) by a joint learning, and the third work Yu et al. (2018) proposed a neural module network to structure the input sentence and do modular comprehension for better performance.

Besides generating natural description from single image or single region, we also explore the album summarization and storytelling Yu et al. (2017a). This is a more challenging problem, requiring us to identify the most representative photos and then generate stories from those summary photos. For this task, we make use of the Visual Storytelling dataset Huang et al. (2016) and a model composed of three hierarchically-attentive Recurrent Neural Nets (RNNs) to: encode the album photos, select representative (summary) photos, and compose the story. Automatic and human evaluations show our model achieves better performance on selection, generation, and retrieval than baselines.

Last but not least, we seek to apply vision and language into a more realistic task - embodied AI, where an agent is required to explore and navigate inside a house based on its first-person view. To study an overall performance of understanding language and doing navigation, we propose a novel task called multi-target embodied question answering Yu et al. (2019). we study the questions that have multiple targets in them, such as "Is the dresser in the bedroom bigger than the oven in the kitchen?", where the agent has to navigate to multiple locations ("dresser in bedroom", "oven in kitchen") and perform comparative reasoning ("dresser" bigger than "oven") before it can answer a question. Such questions require the development of entirely new modules or components in the agent. We propose a modular architecture composed of a program generator, a controller, a navigator, and a VQA module to address this challenging task.

With all the above mentioned abilities together, including visual question answering, image(s)-to-text generation, object localization from natural description, and embodied navigation, we could easily initialize a human-computer interaction task, by talking to a robot and letting it to localize the target object and move toward it.

CHAPTER 2: VISUAL MADLIBS

2.1 Introduction

Much of everyday language and discourse concerns the visual world around us, making understanding the relationship between the physical world and language describing that world an important challenge problem for AI. Understanding this complex and subtle relationship will have broad applicability toward inferring human-like understanding for images, producing natural human robot interactions, and for tasks like natural language grounding in NLP. In computer vision, along with improvements in deep learning based visual recognition, there has been an explosion of recent interest in methods to automatically generate natural language descriptions for images Chen and Lawrence Zitnick (2015); Fang et al. (2015); Karpathy and Fei-Fei (2015); Vinyals et al. (2015b); Kiros et al. (2014); Lebret et al. (2015) or videos Venugopalan et al. (2014); Donahue et al. (2015). However, most of these methods and existing datasets have focused on only one type of description, a generic description for the entire image.

In this work, we collect a new dataset of focused, targeted, descriptions, the *Visual Madlibs dataset* Yu et al. (2015), as illustrated in Figure 2.1. To collect this dataset, we introduce automatically produced fill-in-the-blank templates designed to collect a range of different descriptions for visual content in an image. For example, a user might be presented with an image and a fill-in-the-blank template such as "The frisbee is [blank]" and asked to fill in the [blank] with a description of the appearance of frisbee. Alternatively, they could be asked to fill in the [blank] with a description of what the person is doing with the frisbee. Fill-in-the-blank questions can be targeted to collect descriptions about people and objects, their appearances, activities, and interactions, as well as descriptions of the general scene or the broader emotional, spatial, or temporal

1. This place is a park.
2. When I look at this picture, I feel competitive.
3. The most interesting aspect of this picture is the guys playing shirtless.
4. One or two seconds before this picture was taken, the person caught the frisbee.
5. One or two seconds after this picture was taken, the guy will throw the frisbee.
6. Person A is wearing blue shorts.
7. Person A is in front of person B.
8. Person A is blocking person B.
9. Person B is a young man wearing an orange hat.
10. Person B is on a grassy field.
11. Person B is holding a frisbee.
12. The frisbee is white and round.
13. The frisbee is in the hand of the man with the orange cap.
14. People could throw the frisbee.
15. The people are playing with the frisbee.

Figure 2.1: An example from the Visual Madlibs Dataset. This dataset collects targeted descriptions for people and objects, denoting their appearances, affordances, activities, and interactions. It also provides descriptions of broader emotional, spatial and temporal context for an image.

context of an image. Using these templates, we collect a large collection of 360,001 targeted descriptions for 10,738 images.

With this new dataset, we can develop methods to generate more focused descriptions. Instead of asking an algorithm to "describe the image" we can now ask for more focused descriptions such as "describe the person", "describe what the person is doing," or "describe the relationship between the person and the frisbee." We can also ask questions about aspects of an image that are somewhat beyond the scope of the directly depicted content. For example, "describe what might have happened just before this picture was taken." or "describe how this image makes you feel." These types of descriptions reach toward high-level goals of producing human-like visual interpretations for images.

In addition to focused description generation, we also introduce a multiple-choice question-answering task for images. In this task, the computer is provided with an image and a partial description such as "The person is [blank]". A set of possible answers is also provided, one answer that was written about the image in question, and several additional answers written about other images. The computer is evaluated on how well it can select the correct choice. In this way, we can evaluate performance of description generation on a concrete task, making evaluation more straightforward. Varying the difficulty of the negative answers—adjusting how similar they are to the correct answer—provides a nuanced measurement of performance.

For both the generation and question-answering tasks, we study and evaluate a recent state of the art approach for image description generation Vinyals et al. (2015b), as well as a simple joint-embedding method learned on deep representations. The evaluation also includes extensive analysis of the Visual Madlibs dataset and comparisons to the existing MS COCO dataset of natural language descriptions for images.

In summary, our contributions are:

1) A new description collection strategy, *Visual Madlibs*, for constructing fill-in-the-blank templates to collect targeted natural language descriptions.

2) A new Visual Madlibs Dataset consisting of 360,001 targeted descriptions, spanning 12 different types of templates, for 10,738 images, as well as analysis of the dataset and comparisons to existing MS COCO descriptions.

3) Evaluation of a generation method and a simple joint embedding method for targeted description generation.

4) Definition and evaluation of generation and joint-embedding methods on a new task, *multiple-choice fill-in-the-blank question answering for images*.

2.2 Related Work

Description Generation: Recently, there has been an explosion of interest in methods for producing natural language descriptions for images or video. Early work in this area generally

explored two complementary directions. The first type of approach focused on detecting content elements such as objects, attributes, activities, or spatial relationships and then composing captions for images Kulkarni et al. (2011); Yang et al. (2011); Mitchell et al. (2012); Farhadi et al. (2010) or videos Krishnamoorthy et al. (2013) using linguistically inspired templates. The second type of approach explored methods to make use of existing text either directly associated with an image Feng and Lapata (2010); Aker and Gaizauskas (2010) or retrieved from visually similar images Ordonez et al. (2011); Kuznetsova et al. (2012); Mason (2013).

With the advancement of deep learning for content estimation, there have been many exciting recent attempts to generate image descriptions using neural network based approaches. Some methods first detect words or phrases using Convolutional Neural Network (CNN) features, then generate and re-rank candidate sentences Fang et al. (2015); Lebret et al. (2015). Other approaches take a more end-to-end approach to generate output descriptions directly from images. Kiros *et al.* Kiros et al. (2014) learn a joint image-sentence embedding using visual CNNs and Long Short Term Memory (LSTM) networks. Similarly, several other methods have made use of CNN features and LSTM or recurrent neural networks (RNN) for generation with a variety of different architectures Vinyals et al. (2015b); Karpathy and Fei-Fei (2015); Chen and Lawrence Zitnick (2015). These new methods have shown great promise for image description generation under some measures (e.g. BLEU-1) achieving near-human performance levels. We look at related, but more focused description generation tasks.

Description Datasets: Along with the development of image captioning algorithms there have been a number of datasets collected for this task. One of the first datasets collected for this problem was the UIUC Pascal Sentence data set Farhadi et al. (2010) which contains 1,000 images with 5 sentences per image written by workers on Amazon Mechanical Turk. As the description problem gained popularity larger and richer datasets were collected, including the Flickr8K Rashtchian et al. (2010) and Flickr30K Young et al. (2014) datasets, containing 8,000 and 30,000 images respectively. In an alternative approach, the SBU Captioned photo dataset Ordonez et al. (2011) contains 1 million images with existing captions collected from Flickr. This

dataset is larger, but the text tends to contain more contextual information since captions were written by the photo owners. Most recently, Microsoft released the MS COCO Lin et al. (2014) dataset. MS COCO contains 120,000 images depicting 80 common object classes, with object segmentations and 5 turker written descriptions per image. These datasets have been one of the driving forces in improving methods for description generation, but are currently limited to a single description about the general content of an image. We make use of MS COCO data, extending the types of descriptions associated with images.

Question-answering: Natural language question-answering has been a long standing goal of NLP, with commercial companies like Ask-Jeeves or Google playing a significant role in developing effective methods. Recently, embedding and deep learning methods have shown great promise for question-answering Sukhbaatar et al. (2015); Bordes et al. (2014a,b). Lin *et al.* Lin and Parikh (2015) take an interesting multi-modal approach to question-answering. A multiple-choice text-based question is first constructed from 3 sentences written about an image; 2 of the sentences are used as the question, and 1 is used as the positive answer, mixed with several negative answers from sentences written about other images. The authors develop ranking methods to answer these questions and show that generating abstract images for each potential answer can improve results. Note, here the algorithms are not provided with an image as part of the question. Some recent work has started to look at the problem of question-answering for images. Malinowski *et al.* Malinowski and Fritz (2014) combine computer vision and NLP in a Bayesian framework, but restrict their method to scene based questions. Geman *et al.* Geman et al. (2015) design a visual Turing test to test image understanding using a series of binary questions about image content. We design more general question-answering tasks that allow us to ask a variety of different types of natural language questions about images.

2.3 Designing Visual Madlibs

The goal of Visual Madlibs is to study targeted natural language descriptions of image content that go beyond describing which objects are in the image, and beyond generic descriptions

of the whole image. The madlibs begin with a dataset of images where the presence of some objects have already been labeled. The prompts for the Madlibs-style fill-in-the-blank questions are automatically generated based on image content, in a manner designed to elicit more detailed descriptions of the objects, their interactions, and the broader context of the scene shown in each image.

Visual Madlibs – Image+Instruction+Prompts+Blank: A single fill-in-the-blank question consists of a prompt and a blank, e.g., *Person A is [blank] the car.* The implicit question is, "What goes in the blank?". This is presented to a person along with an image and instructions, e.g., *Describe the relationship between the indicated person and object.* The same image and prompt may be used with different instructions to collect a variety of description types.

Type	Instruction	Prompt	#words
1. image's scene	Describe the type of scene/place shown in this picture.	The place is a(n) _____ .	4+1.45
2. image's emotion	Describe the emotional content of this picture.	When I look at this picture, I feel _____ .	8+1.14
3. image's interesting	Describe the most interesting or unusual aspect of this picture.	The most interesting aspect of this picture is _____ .	8+3.14
4. image's past	Describe what happened immediately before this picture was taken.	One or two seconds before this picture was taken, _____ .	9+5.45
5. image's future	Describe what happened immediately after this picture was taken.	One or two seconds after this picture was taken, _____ .	9+5.04
6. object's attribute	Describe the appearance of the indicated object.	The object(s) is/are _____ .	3.20+1.62
7. object's affordance	Describe the function of the indicated object.	People could _____ the object(s).	4.20+1.74
8. object's position	Describe the position of the indicated object.	The object(s) is/are _____ .	3.20+3.35
9. person's attribute	Describe the appearance of the indicated person/people.	The person/people is/are _____ .	3+2.52
10. person's activity	Describe the activity of the indicated person/people.	The person/people is/are _____ .	3+2.47
11. person's location	Describe the location of the indicated person/people.	The person/people is/are _____ .	3.20+3.04
12. pair's relationship	Describe the relationship between the indicated person and object.	The person/people is/are _____ the object(s).	5.20+1.65

Table 2.1: All 12 types of Madlibs instructions and prompts. Right-most column shows the average number of words for each description (#words for prompt + #words for answer).

Instantiating Questions: While the general form of the questions for the Visual Madlibs were chosen by hand, see Table 2.1, most of the questions are instantiated depending on a subset of the objects present in an image. For instance, if an image contained two people and a dog, questions about each person (question types 9-11 in Table 2.1), the dog (types 6-8), relationships between the two people and the dog (type 12), could be instantiated. For each possible instantiation, the wording of the questions might alter slightly to maintain grammatical consistency. In addition to these types of questions that depend on the objects present in the image, other questions (types 1-5) can be instantiated for an image regardless of the objects present.

Notice in particular the questions about the temporal context – what might have happened before or what might happen after the image was taken. People can make inferences beyond the

specific content depicted in an image. Sometimes these inferences will be consistent between people (e.g., when what will happen next is obvious), and other times these descriptions may be less consistent. We can use the variability of returned responses to select images for which these inferences are reliable.

Asking questions about every object and all pairs of objects quickly becomes unwieldy as the number of objects increases. To combat this, we choose a subset of objects present to use in instantiating questions. Such selection could be driven by a number of factors. The experiments in this work consider comparisons to existing, general, descriptions of images, so we instantiate questions about the objects mentioned in those existing natural language descriptions. Whether an object is mentioned in an image description can be viewed as an indication of the object's importance Berg et al. (2012).

Labeled instances:
person, person, frisbee, ~~ear, ear, ear~~

MS COCO descriptions:
1. Two guys are playing Frisbee in the park.
2. Two young men playing a shirtless game of frisbee.
3. Two men in a grassy field playing with a frisbee.
4. Two men are playing with a frisbee together.
5. Two shirtless men playing frisbee in a field.

Figure 2.2: COCO instance annotation and descriptions for the image of Fig. 2.1. We show how we map labeled instances to the mentioned person and object in the sentence.

2.3.1 Data Collection

To collect the Visual Madlibs Dataset we use a subset of 10,738 human-centric images from MS COCO, that make up about a quarter of the validation data Lin et al. (2014), and instantiate fill-in-the-blank templates as described above. The MS COCO images are annotated with a list of objects present in the images, segmentations for the locations of those objects, and 5 general natural language descriptions of the image. To select the subset of images for collecting Madlibs, we start with the 19,338 images with a person labeled. We then look at the five descriptions for each and perform a dependency parse De Marneffe et al. (2006), only keeping those images where a word referring to a person (woman, man, etc. E.g., in Fig. 2.2, guys, men) is the head noun for

part of the parse. This leaves 14,150 images. We then filter out the images whose descriptions do not include a synonym for any of the 79 non-person object categories labeled in the MS COCO dataset. This leaves 10,738 human-centric images with at least one other object from the MS COCO data set mentioned in the general image descriptions.

Before final instantiation of the fill-in-the blank templates, we need to resolve a potential ambiguity regarding which objects are referred to in the descriptions. There could be several different people or different instances of an object type labeled in an image. It is not immediately obvious which ones are described in the sentences. To address this assignment problem, we estimate the quantity of each described person/object in the sentence by parsing the determinant (two men and a frisbee in Fig. 2.2), the conjunction (a man and a woman), and the singular/plural form (dog, dogs). We compare this number with the number of annotated instances for each category, and consider two possible cases: 1) there are fewer annotated instances than the sentences describe, 2) there are more annotated instances than the sentences describe. It is easy to address the first case, just construct templates for all of the labeled instances. For the second case, we sort the area of each segmented instance, and pick the largest ones up to the parsed number for instantiation. Using this procedure, we obtain 26,148 labeled object or person instances in the 10,738 images.

Each Visual Madlib is answered by 3 workers on Amazon's Mechanical Turk. To date, we have collected 360,001 answers to Madlib questions. Some example Madlibs answers are shown in Fig. 2.3.

2.4 Tasks: Multiple-choice question answering and targeted generation

We design two tasks to evaluate targeted natural language description for images. The first task is to automatically generate natural language descriptions of images to fill in the blank for one of the Madlibs questions. This allows for producing targeted descriptions such as: a description specifically focused on the appearance of an object, or a description about the relationship between two objects. The input to this task is an image, instructions, and a Madlibs prompt. As

Figure 2.3: Madlibs description. The first row corresponds to question types 1-5, the second row corresponds to question types 9-11, and the third row is to question types 6-8 and question type 12. All question types are listed in Table 2.1.

has been discussed at length in the community working on description generation for images, it can be difficult to evaluate free form generation. Our second task tries to address this issue by developing a new targeted multiple-choice question answering task for images. Here the input is again an image, instruction, and a prompt, but instead of a free form text answer, there are a fixed set of multiple-choice answers to fill in the blank. The possible multiple-choice answers are sampled from the Madlibs responses, one that was written for the particular image/instruction/prompt as the correct answer, and distractors chosen from either similar images or random images depending on the level of difficulty desired. This ability to choose distractors to adjust the difficulty of the question as well as the relative ease of evaluating multiple choice answers are attractive aspects of this new task.

In our experiments we randomly select 20% of the 10,738 images to use as our test set for evaluating these tasks. For the multiple-choice questions we form two sets of answers for each, with one set designed to be more difficult than the other. We first establish the easy task distractor answers by randomly choosing three descriptions (of the same question type) from other images Lin and Parikh (2015). The hard task is designed more delicately. Instead of randomly choosing from the other images, we now only look for those containing the same objects as our question image, and then arbitrarily pick three of their descriptions. Sometimes, the descriptions sampled from "similar" images could also be good answers for our questions (later we experiment with using Turkers to select less ambiguous multiple-choice questions from this set). For the targeted generation task, for question types 1-5, algorithms generate descriptions given the image, instructions, and prompt. For the other question types whose prompts are related to some specific person or object, we additionally provide the algorithm with the location of each person-/object mentioned in the prompt. We also experiment with estimating these locations using object detectors.

2.5 Analyzing the Visual Madlibs Dataset

We begin by conducting quantitative analyses of the responses collected in the Visual Madlibs Dataset in Sec. 2.5.1. A main goal is understanding what additional information is provided by the targeted descriptions in the Visual Madlibs Dataset vs general image descriptions. The MS COCO dataset Lin et al. (2014) collects general image descriptions following a similar methodology to previous efforts for collecting general image descriptions, e.g. Rashtchian et al. (2010); Young et al. (2014). So, we provide further analyses comparing the Visual Madlibs to the MS COCO descriptions collected for the same images in Sec. 2.5.2.

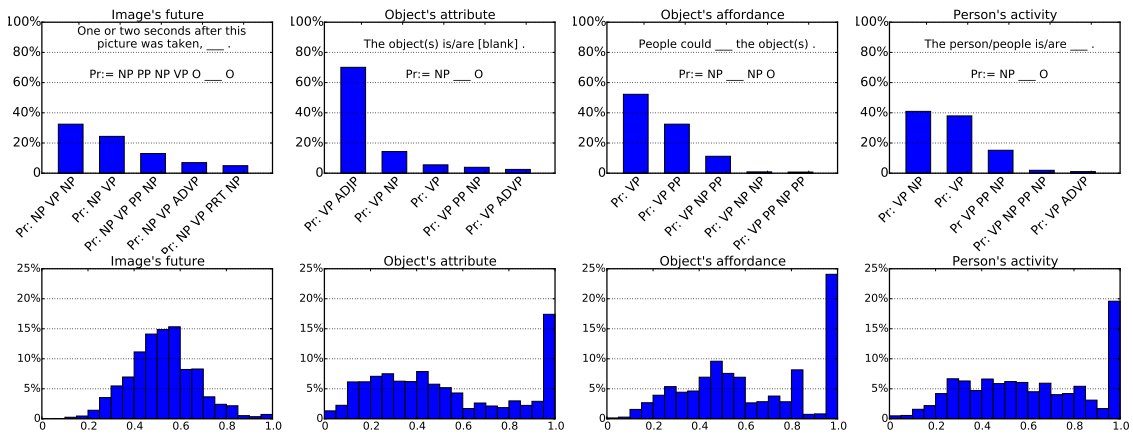


Figure 2.4: First row shows top-5 most frequent phrase templates for image's future, object's attribute, object's affordance and person's activity. Second row shows the histograms of similarity between answers.

2.5.1 Quantifying Visual Madlibs responses

We analyze the length, structure, and consistency of the Visual Madlibs responses. First, the average length of each type of description is shown in the far right column of Table 2.1. Note that descriptions of people tend to be longer than descriptions of other objects in the dataset[1].

Second, we use the phrase chunking Collobert et al. (2011) to analyze which phrasal structures are commonly used to fill in the blanks for different questions. Fig. 2.4, top row, shows relative frequencies for the top-5 most frequent templates used for several question types. Object attributes are usually described briefly with a simple adjectival phrase. On the other hand, people use more words and a wider variety of structure to describe possible future events. Except for future and past descriptions, the distribution of structures is generally concentrated on a few likely choices for each question type.

Third, we analyze how consistent the Mechanical Turk workers' answers are for each type of question. To compute a measure of similarity between a pair of responses we use the cosine similarity between representations of each response. A response is represented by the mean of the Word2Vec Mikolov et al. (2013) vectors for each word in the response, following Lin and Parikh

[1] Also note that the length of the prompts varies slightly depending on the object names used to instantiate the Madlib, hence the fractional values in the mean length of the prompts shown in gray.

a) person's attribute

The person is an old man in red coat.

- refer name: man + person
- general attribute: old
- affiliate attribute: red
- affiliate name: coat

b) object's attribute

The dog is large.

- object attribute: large

c) person's activity/pair's relationship

The person is running.

The person is waiting on the boat.

- verb: running, waiting

	MSCOCO	Mad Libs
Refer	95.1%	46.5 % (100%)
General att	15.5%	37.3%
Affiliate att	1.8%	16.3%
Affiliate obj	7.4%	29.6%
Verb	79.0%	95.4%
Object att	18.7%	86.8%

Figure 2.5: Template used for parsing person's attributes, activity and interaction with object, and object's attribute. The percentages below compares Madlibs and MSCOCO on how frequent these templates are used for description.

(2015); Lin et al. (2014). Word2Vec is a 300 dimensional embedding representation for words that encodes the distributional context of words learned over very large word corpora. This measure takes into account the actual words used in a response, as opposed to the previous analyses of parse structure. Each Visual Madlibs question is answered by three workers, providing 3 pairs for which similarity is computed. Fig. 2.4, bottom row, shows a histogram of all pairwise similarities for several question types. Generally the similarities have a normal-like distribution with an extra peak around 1 indicating the fraction of responses that agree almost perfectly. Once again, descriptions of the future and past are least likely to be (near) identical, while object attributes and affordances are often very consistent.

2.5.2 Visual Madlibs vs general descriptions

We compare the targeted descriptions in the Visual Madlibs Dataset to the general image descriptions in MS COCO. First, we analyze the words used in Visual Madlibs compared to MS COCO descriptions of the same images. For each image, we extract the unique set of words from all descriptions of that image from both datasets, and compute the coverage of each set with respect to the other. We find that on average (across images) 22.45% of the Madlibs's words are also present in MSCOCO descriptions, while 52.38% of the COCO words are also present in Madlibs.

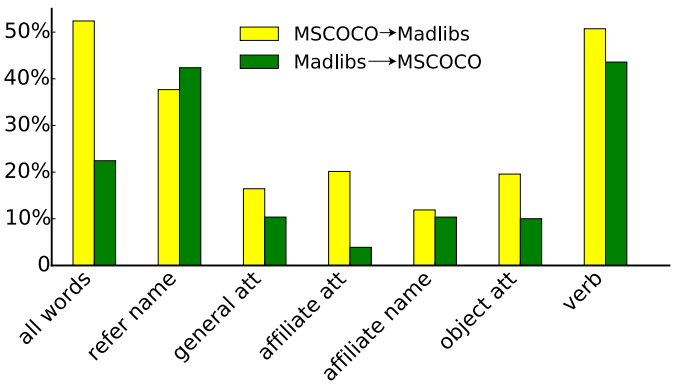


Figure 2.6: Frequency that a word in a position in the people and object parsing template in one dataset is in the same position for the other dataset.

Second, we compare how Madlibs and MS COCO answers describe the people and objects in images. We observe that the Madlibs questions types, Table 2.1, cover much of the information in MS COCO descriptions Lin et al. (2014). As one way to see this, we run the StanfordNLP parser[2] on both datasets. For attributes of people, we use the parsing template shown in Fig. 2.5(a) to analyze the structures being used. The *refer name* indicates whether the person was mentioned in the description. Note that the Madlibs descriptions always have one reference to a person in the prompt (The person is [blank].). Therefore, for Madlibs, we report the presence of additional references to the person (e.g., the person is a *man*). The *general attribute* directly describes the appearance of the person or object (e.g., old or small); the *affiliate object* indicates whether additional objects are used to describe the targeted person (e.g. with a bag, coat, or glasses) and the *affiliate attribute* are appearance characteristics of those secondary objects (e.g., red coat). The templates for *object's attribute* and *verbs* are more straightforward as shown in Fig. 2.5(b)(c). The table in Fig. 2.5 shows the frequency of each parse component. Overall, more of the potential descriptive elements in these constructions are used in response to the Madlibs prompts than in the general descriptions found in MS COCO.

We also break down the overlap between Visual Madlibs and MS COCO descriptions over different parsing templates for descriptions about people and object (Fig. 2.6). Yellow bars show how often words for each parse type in MSCOCO descriptions were also found in the same parse

[2] `http://nlp.stanford.edu/software/lex-parser.shtml`

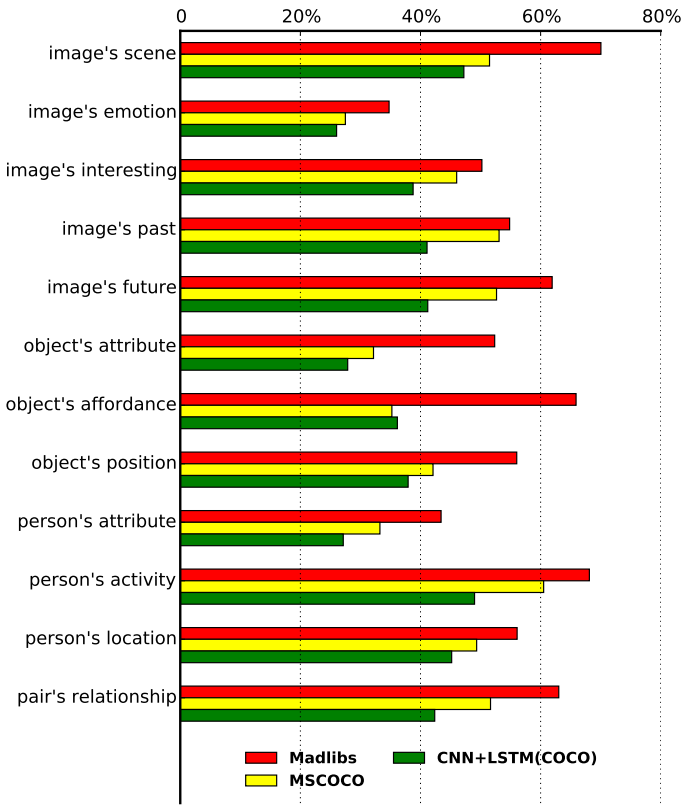


Figure 2.7: The accuracy of Madlibs, MS COCO and CNN+LSTM Vinyals et al. (2015b)(trained on MS COCO) used as references to answer the Madlibs hard multiple-choice questions.

type in the Visual Madlibs answers, and green bars measure the reverse direction. Observations indicate that Madlibs provides more coverage in its descriptions than MS COCO for all templates except for person's refer name. One possible reason is that the prompts already indicates "the person" or "people" explicitly, so workers need not add an additional reference to the person in their descriptions.

Extrinsic comparison of Visual Madlibs Data and general descriptions: Here we provide an extrinsic analysis of the information available in the general descriptions compared to Visual Madlibs. We perform this analysis by using either: a) the MS COCO descriptions for an image, or b) Visual Madlibs responses from other Turkers for an image, to select answers for our multiple-choice evaluation task. Specifically, we use one of the human provided descriptions, either from Madlibs or from MS COCO, and select the multiple-choice answer that is most sim-

ilar to that description. Similarity is measured as cosine similarity between the mean Word2Vec vectors for the words a description compared to the Word2Vec vectors of the multiple-choice answers. In addition to comparing how well the Madlibs or MS COCO descriptions can select the correct multiple-choice answer, we also use the descriptions automatically produced by a recent natural language generation system (CNN+LSTM Vinyals et al. (2015b), implementation from Karpathy and Fei-Fei (2015)) trained on MS COCO dataset. This allows us to make one possible measurement of how close current automatically generated image descriptions are to our Madlibs descriptions. Fig. 2.7 shows the accuracies resulting from using Madlibs, MSCOCO, or CNN+LSTM Vinyals et al. (2015b) to select the correct multiple-choice answer.

Although this approach is quite simple, it allows us we make two interesting observations. First, Madlibs outperforms MS COCO on all types of multiple-choice questions. If Madlibs and MS COCO descriptions provided the same information, we would expect their performance to be comparable. Presumably the performance increase for Madlibs is due to the coverage of targeted descriptions compared to MS COCO's sentences that describe the overall image content more generally. Second, the automatically generated descriptions from the pre-trained CNN+LSTM perform much worse than the actual MS COCO descriptions, despite doing quite well on general image description generation (The BLEU-1 score of CNN+LSTM, 0.67, is near human agreement 0.69 on MS COCO Vinyals et al. (2015b)).

2.6 Experiments

We evaluate a series of methods on the Visual Madlibs Dataset for the targeted natural language generation and multiple-choice question answering tasks, introduced in Sec. 2.4. As methods, we evaluate simple joint-embedding methods – canonical correlation analysis (CCA) and normalized CCA (nCCA) Gong et al. (2014b) – as well as a recent deep-learning based method for image description generation – CNN+LSTM Vinyals et al. (2015b). We train these models on 80% of the images in the MadLibs collection and evaluate their performance on the remaining 20%.

Easy Task

	#Q	CCA	nCCA	nCCA (bbox)	nCCA (all)	CNN+LSTM (madlibs)
1. scene	6277	75.7%	86.8%	–	**87.6**%	71.1%
2. emotion	5138	41.3%	**49.2**%	–	42.4%	34.0%
3. past	4903	61.8%	77.5%	–	**80.3**%	35.8%
4. future	4658	61.2%	78.0%	–	**80.2**%	40.0%
5. interesting	5095	66.8%	76.5%	–	**78.9**%	39.8%
6. obj attr	7194	44.1%	47.5%	**54.7**%	50.9%	45.4%
7. obj aff	7326	59.8%	73.0%	72.2%	**76.7**%	–
8. obj pos	7290	53.0%	65.9%	58.9%	**69.7**%	50.9%
9. per attr	6651	40.4%	48.0%	**53.1**%	44.5%	37.3%
10. per act	6501	70.0%	80.7%	75.6%	**82.8**%	63.7%
11. per loc	6580	69.8%	82.7%	73.8%	**82.7**%	59.2%
12. pair rel	7595	54.3%	63.0%	64.2%	**67.2**%	–

Hard Task

	#Q	CCA	nCCA	nCCA (bbox)	nCCA (all)	CNN+LSTM (madlibs)
1. scene	6277	63.8%	**70.1**%	–	68.2%	60.5%
2. emotion	5138	33.9%	**37.2**%	–	33.2%	32.7%
3. past	4903	47.9%	52.8%	–	**54.0**%	32.0%
4. future	4658	47.5%	**54.3**%	–	53.3%	34.3%
5. interesting	5095	51.4%	53.7%	–	**55.1**%	33.3%
6. obj attr	7194	42.2%	43.6%	**49.8**%	39.3%	40.3%
7. obj aff	7326	54.5%	63.5%	**63.0**%	48.5%	–
8. obj pos	7290	49.0%	**55.7**%	50.7%	53.4%	44.9%
9. per attr	6651	33.9%	38.6%	**46.1**%	31.6%	36.1%
10. per act	6501	59.7%	65.4%	65.1%	**66.6**%	53.6%
11. per loc	6580	56.8%	**63.3**%	57.8%	62.6%	49.3%
12. pair rel	7595	49.4%	54.3%	**56.5**%	52.0%	–

Filtered questions from **Hard**

	#Q	CCA	nCCA	nCCA (bbox)	nCCA (all)	CNN+LSTM (madlibs)
1. scene	4940	70.4%	**77.6**%	–	76.3%	66.3%
2. emotion	2052	43.2%	**49.0**%	–	43.8%	34.0%
3. future	3820	51.4%	**59.2**%	–	58.3%	33.3%
4. past	3976	51.0%	57.4%	–	**59.4**%	31.1%
5. interesting	4159	56.1%	59.5%	–	**61.3**%	35.4%
6. obj attr	5436	45.3%	47.2%	**54.6**%	42.8%	43.2%
7. obj aff	4581	61.2%	**71.0**%	70.5%	57.6%	–
8. obj pos	5721	53.0%	**60.2**%	54.6%	57.7%	46.9%
9. per attr	4893	36.5%	42.4%	**52.1**%	34.4%	37.0%
10. per act	5813	62.0%	68.3%	67.9%	**69.6**%	54.8%
11. per loc	5096	63.1%	69.9%	62.6%	**70.0**%	51.2%
12. pair rel	5981	52.3%	57.6%	**60.0**%	56.5%	–

Table 2.2: Accuracies computed for different approaches on the easy and hard multiple-choice answering task, and the filtered hard question set. CCA, nCCA, and CNN+LSTM are trained on the whole image representation for each type of question. nCCA(box) is trained and evaluated on ground-truth bounding-boxes from COCO segmentations. nCCA(all) trains a single embedding using all question types.

		Easy Task			Hard Task		
	#Q	nCCA	nCCA (bbox)	nCCA (dbox)	nCCA	nCCA (bbox)	nCCA (dbox)
6. obj attr	2021	47.6%	53.6%	51.4%	43.9%	47.9%	45.2%
9. per attr	4206	50.2%	55.4%	51.2%	40.0%	47.0%	43.3%

Table 2.3: Multiple-choice answering using automatic detection for 42 object/person categories. "bbox" denotes ground-truth bounding box and "dbox" denotes detected bounding box.

In our experiments we extract image features using the VGG Convolutional Neural Network (CNN) Simonyan and Zisserman (2014). This model has been trained on the ILSVRC-2012 dataset to recognize images depicting 1000 object classes, and generates a 4,096 dimensional image representation. On the sentence side, we average the Word2Vec of all words in a sentence to obtain a 300 dimensional representation.

CCA is an approach for finding a joint embedding between two multi-dimensional variables, in our case image and text vector representations. In an attempt to increase the flexibility of the feature selection and for improving computational efficiency, Gong *et al.* Gong et al. (2014b) proposed a scalable approximation scheme of explicit kernel mapping followed by dimension reduction and linear CCA. In the projected latent space, the similarity is measured by the eigenvalue-weighted normalized correlation. This method, nCCA, provides high-quality retrieval results, improving over the original CCA performance significantly Gong et al. (2014b).

We train CCA and nCCA models for each question type separately using the training portion of the Visual Madlibs Dataset. These models allow us to map from an image representation, to the joint-embedding space, to vectors in the Word2Vec space, and vice versa. For targeted generation, we map an image to the joint-embedding space and then choose the answer from the training set text that is closest to this embedded point. In order to answer a multiple-choice question we embed each multiple choice answer, and then select the answer who's embedding is closest to image.

Following the recent "Show and Tell" description generation technique Vinyals et al. (2015b) (using an implementation from Karpathy and Fei-Fei (2015)), we train a CNN+LSTM model for each question type on the Visual Madlibs training set. This approach has demonstrated state of the art performance on generating general natural language descriptions for images. These

Figure 2.8: Some example question-answering results from nCCA. First row shows correct choices. Second row shows incorrect choices.

	BLEU-1			BLEU-2		
	nCCA	nCCA(bbox)	CNN+LSTM	nCCA	nCCA(box)	CNN+LSTM
1. scene	0.52	–	0.62	0.17	–	0.19
2. emotion	0.17	–	0.39	0	–	0
3. future	0.38	–	0.32	0.12	–	0.08
4. past	0.39	–	0.42	0.12	–	0.11
5. interesting	0.49	–	0.51	0.14	–	0.15
6. obj attr	0.28	0.36	0.45	0.02	0.02	0.01
7. obj aff	0.56	0.60	–	0.10	0.11	–
8. obj pos	0.53	0.55	0.71	0.24	0.25	0.50
9. per attr	0.26	0.29	0.55	0.06	0.07	0.25
10. per act	0.47	0.41	0.52	0.14	0.11	0.22
11. per loc	0.52	0.46	0.64	0.22	019	0.39
12. pair rel	0.46	0.48	–	0.07	0.08	–

Table 2.4: BLEU-1 and BLEU-2 computed on Madlibs testing dataset for different approaches.

models directly learn a mapping from an image to a sequence of words which we can use to evaluate the targeted generation task. Note that we input the words from the prompt, e.g., The chair is, and then let the CNN+LSTM system generate the remaining words of the description[3]. For the multiple choice task, we compute cosine similarity between Word2Vec representations of the generated description and each question answer and select the most similar answer.

[3] The missing entries for questions 7 and 12 are due to this priming failing for a fraction of the questions.

2.6.1 Discussion of results

Table 2.2 shows accuracies of each algorithm on the easy and hard versions of the multiple-choice task. Fig. 2.8, shows example correct and wrong answer choices. There are several interesting observations we can make. First, training nCCA on all types of question together, labeled as nCCA(all), is helpful for the easy variant of the task, however it is less useful on the "fine-grained" hard version of the task. Second, extracting visual features from the bounding box of the relevant person/object yields higher accuracy for predicting attributes, but not for other questions. Based on this finding, we try answering the attribute question using automatic detection methods. The detectors are trained on ImageNet using R-CNN Girshick et al. (2014), covering 42 MS COCO categories. We observe similar performance between ground-truth and detected bounding boxes in Table 2.3.

As an additional experiment we ask humans to answer the multiple choice task, with 5 Turkers answering each question. We use their results to filter out a subset of the hard multiple-choice questions where at least 3 Turkers choose the correct answer. Results of the methods on this subset are shown in Table 2.2 bottom set of rows. These results show the same pattern as on the unfiltered set, with slightly higher accuracy.

Table 2.4 shows BLEU-1 and BLEU-2 scores for targeted generation. Although the CNN+LSTM models we trained on Madlibs were not quite as accurate as nCCA for selecting the correct multiple-choice answer, they did result in better, sometimes much better, accuracy (as measured by BLEU scores) for targeted generation.

CHAPTER 3: REFERRING EXPRESSION GENERATION AND COMPREHENSION

While image description strives to construct broad descriptions of image content, referring expressions are a more focused form of language, used to identify a particular object or person in an image. People use such expressions all the time, especially in dialogue to indicate a particular object or event to a co-observer, e.g. *the woman in the blue shirt*, or *the green cup on the table*. Computational models that generate and comprehend such expressions have broad applicability to human-computer interaction, especially for agents such as robots, interacting with people in the real world. Successful models need to connect visual interpretations of objects in the world to natural language that discriminatively describes an object.

In this chapter, we first define the two tasks, then discuss about how we collect the referring expression dataset, and finally introduce 3 deep learning based approaches to address the two tasks.

3.1 Two Tasks

For Referring Expressions (RE), there is a pragmatic interaction between agents that involves two main tasks: a) a speaker task where one must generate a natural language expression given a target and its surrounding world context; (b) a listener task where one must interpret and comprehend the expression and map it to the correct target. We refer to these two tasks as referring expression generation and comprehension, respectively.

The comprehension task requires a system to select the region being described by a given referring expression. To address this problem, Mao et al. (2016); Yu et al. (2016b); Nagaraja et al. (2016); Hu et al. (2016b) model $P(r|o)$ and looks for the object o maximizing the probability. People also try modeling $P(o, r)$ directly using embedding model Rohrbach et al. (2016a); Wang

et al. (2016), which learns to minimize the distance between paired object and sentence in the embedding space.

The generation task asks a system to compose an expression for a specified object within an image. While some previous work used rule-based approaches to generate expressions with fixed grammar pattern Mitchell et al. (2013b); FitzGerald et al. (2013); Kazemzadeh et al. (2014), recent work has followed the CNN-LSTM structure to generate expressions Mao et al. (2016); Yu et al. (2016b).

3.2 Referring Expression Datasets

Some initial datasets in referring expression generation (REG) used graphics engines to produce images of objects van Deemter et al. (2006); Viethen and Dale (2008a) with corresponding shared evaluation challenges Gatt and Belz (2009). Recently more realistic datasets have been introduced, consisting of craft objects like pipecleaners, and ribbons Mitchell et al. (2010), or everyday home and office objects such as staplers or combs Mitchell et al. (2013a), arrayed on a simple background. These datasets helped move REG research into the domain of real world objects.

In the past few years, datasets have become even larger and more realistic and expanded to include video REs. The ReferIt Dataset Kazemzadeh et al. (2014) was perhaps the first large-scale RE dataset to be based on complex real world scenes. The images used to construct this dataset were originally sampled from the ImageCLEF IAPR image retrieval dataset Grubinger et al. (2006), a large collection of scene images with associated object segmentations. The ReferIt dataset was collected via a simple two-player online game (the ReferItGame) to crowdsource REs. In this game, Player 1 is shown an image with a highlighted target object and asked to write a natural language expression referring to the target. Player 2 is shown only the image and RE and asked to click on the corresponding object. If the players do their job correctly, they receive points and the expression is added to the dataset. This allows both data collection and verification within the game.

Based on this game, we further collected the RefCOCO and RefCOCO+ datasets, building on the MS COCO image collection Lin et al. (2014). In the RefCOCO dataset, no restrictions are placed on the type of language used in the REs, while in the RefCOCO+ dataset players are stopped from using location words in their REs by adding 'taboo' words to the ReferItGame. Thus, RefCOCO+ tends to focus more on appearance based descriptions. Another dataset based on MS COCO images has also been collected, called the RefCOCOg dataset Mao et al. (2016). During collection of this dataset, one set of workers on Mechanical Turk were asked to write REs for objects. Another set of workers were asked to click on the indicated object given an RE. In Table 3.1, we show the statistics of each of the above-mentioned 4 datasets. REs in RefCOCO and RefCOCO+ tend to contain fewer words than those in Refexp since the competitive and time-based nature of games encourages players to write only the amount of information necessary to convey the correct object to the other player. Refexp contains more caption-like REs with many details about each referred object since labelers were encouraged to do so. Fig. 3.1 shows example images and expressions. In this chapter, we evaluate our approaches on RefCOCO, RefCOCO+, and RefCOCOg[1], shows the advantages of our methods for both referring expression generation and comprehension.

Besides, there are two types of splits of the data into train/test sets for RefCOCO and RefCOCO+: a per-object split and a people-vs-objects split. The first type is **per-object split**. In this split, the dataset is divided by randomly partitioning objects into training and testing sets. This means that each object will only appear either in training or testing set, but that one object from an image may appear in the training set while another object from the same image may appear in the test set. We use this split for RefCOCOg since same division was used in the previous state-of-the-art approach Mao et al. (2016). The second type is **people-vs-objects splits**. One thing we observe from analyzing the datasets is that about half of the referred objects are people. Therefore, we create a split for RefCOCO and RefCOCO+ datasets that evaluates images containing multiple people (testA) vs images containing multiple instances of all other objects (testB). In

[1] Datasets and toolbox can be downloaded from https://github.com/lichengunc/refer

Figure 3.1: Example images and referring expressions from RE datasets.

this split all objects from an image will appear either in the training or testing sets, but not both. This split creates a more meaningfully separated division between training and testing, allowing us to evaluate the usefulness of context more fairly.

Table 3.1: 4 referring expression datasets that use realistic images.

Dataset	#images	#expressions	collection way	expression style
Referit	19.894	130,525	Referit Game	Free style
RefCOCO	19,994	142,210	Referit Game	Free style
RefCOCO+	19,992	141,564	Referit Game	Abs. Loc forbidden
RefCOCOg	104,560	26,711	Two rounds	COCO-caption style

3.3 Modeling Context in Referring Expressions

In this section, we focus on incorporating better measures of visual context into referring expression models and find that visual comparison to other objects within an image helps improve performance significantly. We also develop methods to tie the language generation process together, so that we generate expressions for all objects of a particular category jointly.

To investigate the context, we firstly implement several model variations for referring expression generation and comprehension. The baseline models are recent state of the art deep learning approaches from Mao et al Mao et al. (2016). We use these as our baselines (Sec 3.3.1). Next, we investigate incorporating better visual context features into the models (Sec 3.3.2). Finally, we explore methods to jointly produce an entire set of referring expressions for all depicted objects of the same category (Sec 3.3.3).

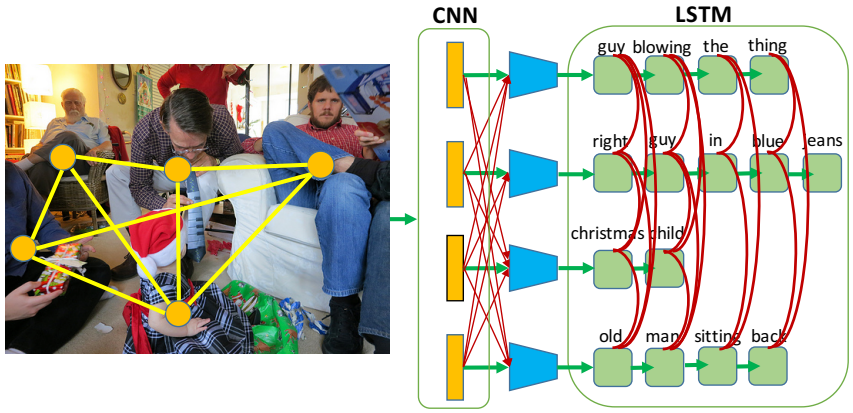


Figure 3.2: Framework: We extract VGG-fc7 and location features for each object of the same type, then compute visual differences. These features and differences are then fed into LSTM. For sentence generation, the LSTMs are tied together, incorporating the hidden output difference as additional information for predicting words.

3.3.1 Baselines

For comparison, we implement both the baseline and strong model of Mao et al Mao et al. (2016). Both models utilize a pre-trained CNN network to model the target object and its context within the image, and then use a LSTM for generation. In particular, object and context are modeled as features from a CNN trained to recognize 1,000 object categories Simonyan and Zisserman (2014) from ImageNet Russakovsky et al. (2015). Specifically, the visual representation is composed of:

- Target object representation, o_i. The object is modeled as features extracted from the VGG-fc7 layer by forwarding its bounding box through the network.

- Global context representation, g_i. Context is modeled as features extracted from the VGG-fc7 layer for the entire image.

- Location/size representation, l_i, for the target object. Location and size are modeled as a 5-d vector encoding the x and y locations of the top left and bottom right corners of the target object bounding box, as well as the bounding box size with respect to the image, i.e., $l_i = [\frac{x_{tl}}{W}, \frac{y_{tl}}{H}, \frac{x_{br}}{W}, \frac{y_{br}}{H}, \frac{w \cdot h}{W \cdot H}]$.

Language generation is handled by a long short-term memory network (LSTM) Hochreiter and Schmidhuber (1997) where inputs are the above visual features and the network is trained to generate natural language referring expressions. In Mao et al's baseline Mao et al. (2016), the model uses maximum likelihood training and outputs the most likely referring expression given the target object, context, and location/size features. In addition, they also propose a stronger model that uses maximum mutual information (MMI) training to consider whether a listener would interpret a referring expression unambiguously. They impose this by penalizing the model if a generated referring expression could also be generated by some other object within the image. We implement both their original model and MMI model in our experiments. We subsequently refer to these two models as Baseline and MMI, respectively.

3.3.2 Visual Comparison

Previous works Brown-Schmidt and Tanenhaus (2006); Mitchell et al. (2013b) have shown that objects in an image, of the same type as the target object, are most important for influencing what attributes people use to describe the target. One drawback of considering a general feature over the entire image to encode context (as in the baseline models) is that it may not specifically focus on visual comparisons to the most relevant objects – the other objects of the same object category within the image.

In this work, we propose a more explicit encoding of the visual difference between objects of the same category within an image. This helps for generating referring expressions which best discriminate the target object from the surrounding objects. For example, in an image with three cars, two blue and one red, visual appearance comparisons could help generate "the red car" as an expression for the latter object.

Given the referred object and its surrounding objects, we compute two types of features for visual comparison. The first type encodes the similarities and differences in *visual appearance* between the target object and other objects of the same cateogry depicted in the image. Inspired by Sadeghi et al Sadeghi et al. (2015), we compute the difference in visual CNN features as

our representation of relative appearance. Because there may be many surrounding objects of the same type in the image, and not every object will provide useful information about how to describe the target object, we need to first select which objects to compare and aggregate their visual differences. In Section 3.3.4, we experiment with selecting different subsets of comparison objects: objects of the same category, objects of different category, or all other depicted objects. For each selected comparison object, we compute the appearance difference as the subtraction of the target object and comparison object CNN representations. We experiment with three different strategies for computing an aggregate vector to represent the visual difference between the target object and the surrounding objects: minimum, maximum, and average over each feature dimension. In our experiments, pooling the average difference between the target object and surrounding objects seems to work best. Therefore, we use this pooling in all experiments.

- Visual appearance difference representation, $\delta v_i = \frac{1}{n}\sum_{j \neq i} \frac{o_i - o_j}{\|o_i - o_j\|}$, where n is the number of objects chosen for comparisons and we use average pooling to aggregate the differences.

The second type of comparison feature encodes the *relative location and size* differences between the target object and surrounding objects of the same object category. People often use comparative size or location terms in referring expressions, e.g. "the second giraffe from the left" or "the smaller monkey" Viethen and Dale (2008b). To address the dynamic number of nearby objects, we choose up to five comparison objects of the same category as the target object, sorted by distance to the target. When fewer than five objects of the same category are depicted, this 25-d vector (5-d x 5 surrounding objects) is padded with zeros.

- Location difference representation, δl_i, where each 5-d difference is computed as $\delta l_{ij} = [\frac{[\triangle x_{tl}]_{ij}}{w_i}, \frac{[\triangle y_{tl}]_{ij}}{h_i}, \frac{[\triangle x_{br}]_{ij}}{w_i}, \frac{[\triangle y_{br}]_{ij}}{h_i}, \frac{w_j h_j}{w_i h_i}]$.

In summary, our final visual representation for a target object is:

$$v_i = W_m[o_i, g_i, l_i, \delta v_i, \delta l_i] + b_m \tag{3.1}$$

where o_i, g_i, l_i are the target object, global context, and location/size features from the baseline model, δv_i and δl_i encodes visual appearance difference and location difference. W_m and b_m project the concatenation of the five types of features to be the final representation.

3.3.3 Joint Language Generation

For the referring expression generation task, rather than generating sentences for each object in an image separately Johnson et al. (2016); Mao et al. (2016), we consider tying the generation process together into a single task to jointly generate expressions for all objects of the same object category depicted in an image. This makes sense intuitively – when a person attempts to generate a referring expression for an object in an image they inherently compose that expression while keeping in mind expressions for the other objects in the picture. This can be observed in the fact that the expressions people generate for objects in an image tend to share similar patterns of expression. If you say "the man on the left" for one object then you tend to say "the man on the right" for the other object. We would like our algorithms to mimic these behaviors. Additionally, the algorithm should also be able to push generated expressions away from each other to create less ambiguous references. For example, if we use the word "red" to describe one object, then we probably shouldn't use the same word to describe another object.

To model this joint generation process, we model generation using an LSTM model where in addition to the usual connections between time steps within an expression we also add connections between expressions for different objects. This architecture is illustrated in Fig 3.2.

Specifically, we use LSTM to generate multiple referring expressions, $\{r_i\}$, given depicted objects of the same type, $\{o_j\}$.

$$\begin{aligned} P(R|O) &= \prod_i P(r_i|o_i, \{o_{j\neq i}\}, \{r_{j\neq i}\}), \\ &= \prod_i \prod_t P(r_{i_t}|r_{i_{t-1}}, ..., r_{i_1}, v_i, \{h_{j_t, j\neq i}\}) \end{aligned} \tag{3.2}$$

where r_{i_t} are words at time t, v_i visual representations, and h_{j_t} is the hidden output of j-th object at time step t that encodes the visual and sentence information for the j-th object. As visual comparison, we aggregate the difference of hidden outputs to push away ambiguous information. $h_{dif_{i_t}} = \frac{1}{n}\sum_{j \neq i} \frac{h_{i_t} - h_{j_t}}{\|h_{i_t} - h_{j_t}\|}$. There, n is the the number of other objects of the same type. The hidden difference is jointly embedded with the target object's hidden output, and forwarded to the softmax layer for predicting the word.

$$P(r_{i_t}|r_{i_{t-1}}, ..., r_{i_1}, v_i, \{h_{j_t, j \neq i}\}) = \text{softmax}(W_h[h_{i_t}, h_{dif_{i_t}}] + b_h) \tag{3.3}$$

3.3.4 Experiments

We first perform some experiments to analyze the use of context in referring expressions (Sec 3.3.4.1). Given these findings, we then perform experiments evaluating the usefulness of our proposed visual and language innovations on the comprehension (Sec 3.3.4.2) and generation tasks (Sec 3.3.4.3).

In experiments for the referring expression comprehension task, we use the same evaluation as Mao et al Mao et al. (2016), namely we first predict the region referred by the given expression, then we compute the intersection over union (IOU) ratio between the true and predicted bounding box. If the IOU is larger than 0.5 we count it as a true positive. Otherwise, we count it as a false positive. We average this score over all images. For the referring expression generation task we use automatic evaluation metrics, BLEU, ROUGE, and METEOR developed for evaluat-

	RefCOCO		RefCOCO+	
	Test A	Test B	Test A	Test B
no context	63.91%	66.31%	50.09%	45.05%
global context	63.15%	64.21%	48.73%	42.13%
scale 2	65.57%	67.13%	50.38%	44.89%
scale 3	66.14%	68.07%	50.25%	45.40%
scale 4	**66.68%**	**68.56%**	**50.34%**	**45.48%**

Table 3.2: Expression Comprehension accuracies on RefCOCO and RefCOCO+ of the Baseline model with differenct context source. Scale n indicates the size of the cropped window centered by the target object.

ing machine translation results, commonly used to evaluate language generation results Xu et al. (2015); Karpathy and Fei-Fei (2015); Fang et al. (2015); Mao et al. (2015); Vinyals et al. (2015b); Kulkarni et al. (2013). We further perform human evaluations, and propose a new metric evaluating the duplicate rate of generated expressions. For both tasks, we compare our models with "Baseline" and "MMI" Mao et al. (2016). Specifically, we denote "visdif" as our visual comparison model, and "tie" as the LSTM tying model. We also perform an ablation study, evaluating the combinations.

3.3.4.1 Analysis Experiments

Context Representation: As previously discussed, we suggest that the approaches proposed in recent referring expression works Mao et al. (2016); Hu et al. (2016b) make use of relatively weak contextual information, by only considering a single global image context for all objects. To verify this intuition, we implemented both the baseline and strong MMI models from Mao et al Mao et al. (2016), and compare the results for referring expression comprehension task with and without global context on RefCOCO and Refcoco+ in Table 3.2. Surprisingly we find that the global context does not improve the performance of the model. In fact, adding context even decreases performance slightly. This may be due to the fact that the global context for each object in an image would be the same, introducing some ambiguity into the referring expression

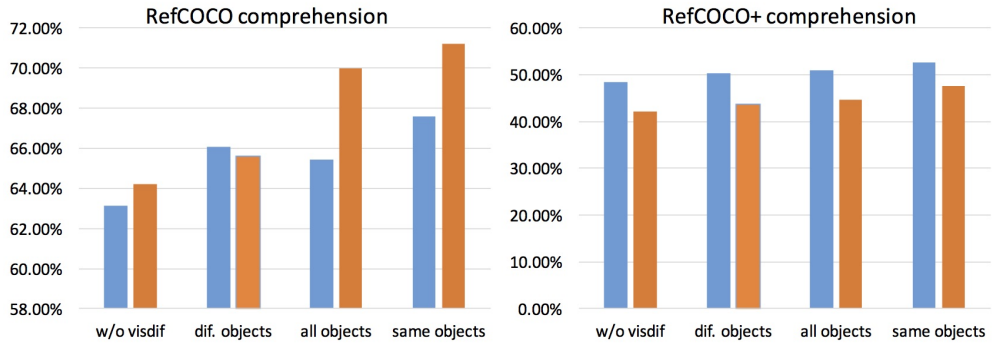


Figure 3.3: Comprehension accuracies on RefCOCO and RefCOCO+ datasets. We compare the performance of "visdif" model without visual comparison, and visual comparison between different-category objects, between all objects, and between same-type objects.

comprehension task. Given these findings, we implemented a simple modification to the global context, computing the same visual representation, but on a somewhat scaled window centered around the target object. We found this to improve performance, suggesting room for improving the visual context feature. This motivate our development of a better context feature.

Visual Comparison: For our visual comparison model, there could be several choices regarding which objects from the image should be compared to the target object. We experiment with three sets of reference objects on RefCOCO and RefCOCO+ datasets: a) objects of the same-category in the image, b) objects of different-category in the image, and c) all objects appeared in the image. We use our "visdif" model for this experiment. The results are shown in Figure 3.3. It is clear to see the visual comparisons to the same-category objects are most useful for referring expression comprehension task. This is more like mimicing how human refer object – we tend to point out the difference between the target object with the other same-category objects within the same image.

3.3.4.2 Referring Expression Comprehension

We evaluate performance on the referring expression comprehension task on RefCOCO, RefCOCO+ and RefCOCOg datasets. For RefCOCO and RefCOCO+, we evaluate on the two subsets of people (testA) and all other objects (testB). For RefCOCOg, we evaluate on the per-object split as previous work Mao et al. (2016). Since the authors haven't released their testing set, we show the performance on their validation set only, using the optimized hyper-parameters on RefCOCO. Table 3.3 shows the comprehension accuracies. We observe that our implementation of Mao et al Mao et al. (2016) achieves comparable performance to the numbers reported in their paper. We also find that adding visual comparison features to the Baseline model improves performance across all datasets and splits. Similar improvements are also observed on top of the MMI model.

In order to make a fully automatic referring system, we also train a Fast-RCNN Girshick (2015) detector and build our system on top of the detections. We train Fast-RCNN on the valida-

tion portion only as the RefCOCO and RefCOCO+ are collected using MSCOCO training data. For RefCOCOg, we use the detection results provided by Mao et al. (2016), which were trained uisng Multibox Erhan et al. (2014). Results on shown in the bottom half of Table 3.3. Although all comprehension accuracies drop due to imperfect detections, the improvements of our models over Baseline and MMI are still observed. One weakness of our automatic system is that it highly depends on detection performance, especially for general objects (testB). However, considering our detector was trained on MSCOCO validation only, we believe such weakness may be alleviated with more training data and stronger detection techniques, e.g., He et al. (2016)Liu et al. (2016)Ren et al. (2015)Bell et al. (2016), etc.

	RefCOCO		RefCOCO+		RefCOCOg
	Test A	Test B	Test A	Test B	Validation
BaselineMao et al. (2016)	63.15%	64.21%	48.73%	42.13%	55.16%
visdif	67.57%	71.19%	52.44%	47.51%	59.25%
MMIMao et al. (2016)	71.72%	71.09%	58.42%	51.23%	62.14%
visdif+MMI	**73.98%**	**76.59%**	**59.17%**	**55.62%**	**64.02%**
Baseline(det)Mao et al. (2016)	58.32%	48.48%	46.86%	34.04%	40.75%
visdif(det)	62.50%	50.80%	50.10%	37.48%	41.85%
MMI(det)Mao et al. (2016)	64.90%	54.51%	54.03%	42.81%	45.85%
visdif+MMI(det)	**67.64%**	**55.16%**	**55.81%**	**43.43%**	**46.86%**

Table 3.3: Referring Expression comprehension results on the RefCOCO, RefCOCO+, and RefCOCOg datasets. Rows of "method(det)" are the results of automatic system built on Fast-RCNN Girshick (2015) and Multibox Erhan et al. (2014) detections.

3.3.4.3 Referring Expression Generation

For the referring expression generation task, we evaluate the usefulness of our visual comparison features as well as our joint language generation model. These serve to tie the generation process together so that the model considers other objects of the same type both visually and linguistically during generation. On the visual side, comparisons are used to judge similarity of the target object to other objects of the same type in terms of appearance, size and location. On the language side, the joint LSTM model serves to both differentiate and mimic language patterns in the referring expressions for the entire set of depicted objects. Fig 3.4 shows some comparison between our model with other methods.

RefCOCO

	Test A				Test B			
	Bleu 1	Bleu 2	Rouge	Meteor	Bleu 1	Bleu 2	Rouge	Meteor
Baseline Mao et al. (2016)	0.477	0.290	0.413	0.173	0.553	0.343	0.499	0.228
MMI Mao et al. (2016)	0.478	0.295	0.418	0.175	0.547	0.341	0.497	0.228
visdif	0.505	**0.322**	0.441	0.184	0.583	0.382	0.530	0.245
visdif+MMI	0.494	0.307	0.441	0.185	0.578	0.375	0.531	0.247
Baseline+tie	0.490	0.308	0.431	0.181	0.561	0.352	0.505	0.234
visdif+tie	**0.510**	0.318	**0.446**	**0.189**	**0.593**	**0.386**	**0.533**	**0.249**
visdif+MMI+tie	0.506	0.312	0.445	0.188	0.579	0.370	0.525	0.246

RefCOCO+

	Test A				Test B			
	Bleu 1	Bleu 2	Rouge	Meteor	Bleu 1	Bleu 2	Rouge	Meteor
Baseline Mao et al. (2016)	0.391	0.218	0.356	0.140	0.331	0.174	0.322	0.135
MMI Mao et al. (2016)	0.370	0.203	0.346	0.136	0.324	0.167	0.320	0.133
visdif	0.407	0.235	0.363	0.145	0.339	0.177	0.325	0.145
visdif+MMI	0.386	0.221	0.360	0.142	0.327	0.172	0.325	0.135
Baseline+tie	0.392	0.219	0.361	0.143	0.336	0.177	0.325	0.140
visdif+tie	**0.409**	**0.232**	**0.372**	**0.150**	**0.340**	**0.178**	**0.328**	**0.143**
visdif+MMI+tie	0.393	0.220	0.360	0.142	0.327	0.175	0.321	0.137

RefCOCOg

	validation			
	Bleu 1	Bleu 2	Rouge	Meteor
Baseline Mao et al. (2016)	0.437	0.273	0.363	0.149
MMI Mao et al. (2016)	0.428	0.263	0.354	0.144
visdif	**0.442**	**0.277**	**0.370**	**0.151**
visdif+MMI	0.430	0.262	0.356	0.145

Table 3.4: Referring Expression Generation Results: Bleu, Rouge, Meteor evaluations for RefCOCO, RefCOCO+ and RefCOCOg.

	RefCOCO		RefCOCO+	
	Test A	Test B	Test A	Test B
Baseline Mao et al. (2016)	62.42%	64.99%	49.18%	42.03%
MMI	65.76%	68.25%	49.84%	45.38%
visdif	68.27%	74.92%	55.20%	43.65%
visdif+MMI	70.25%	75.47%	53.56%	47.58%
Baseline+tie	64.51%	68.34%	52.06%	43.53%
visdif+tie	**71.40%**	76.14%	**57.17%**	47.92%
visdif+MMI+tie	70.01%	**76.31%**	55.64%	**48.04%**

Table 3.5: Human Evaluations on referring expression generation.

Our full results are shown in Table 3.4. We find that incorporating our visual comparison features into the Baseline model improves generation quality (compare row "Baseline" to row "visdif"). It also improves the performance of MMI model (compare row "MMI" to row "visdif+MMI"). We also observe that tying the language generation together across all objects consistently improves the performance (compare the bottom three "+tie" rows with the above). Especially for method "visdif+tie", it achieves the highest score under almost every measurement. We do not perform language tying on RefCOCOg since here some objects from an image may appear in training while others may appear in testing.

	RefCOCO		RefCOCO+	
	Test A	Test B	Test A	Test B
Baseline Mao et al. (2016)	15.60%	16.40%	28.67%	46.27%
MMI	11.60%	11.73%	21.07%	26.40%
visdif	9.20%	8.80%	19.60%	31.07%
visdif+MMI	5.07%	6.13%	12.13%	16.00%
Baseline+tie	11.20%	14.93%	22.00%	32.13%
visdif+tie	**4.27%**	5.33%	11.73%	16.27%
visdif+MMI+tie	6.53%	**4.53%**	**10.13%**	**13.33%**

Table 3.6: Fraction of images for which the algorithm generates the same referring expression for multiple objects. Smaller is better.

Figure 3.4: Referring expression generation on RefCOCO by different methods.

We observe in Table 3.4 that models incorporating "+MMI" are worse than without "+MMI" under the automatic scoring metrics. To verify whether these metrics really reflect performance, we performed human evaluations on the expression generation task. Three Turkers were asked to click on the referred object given the image and the generated expression. If more than two clicked on the true target object, we consider this expression to be correct. Table 3.5 shows the human evaluation results, indicating that models with "+MMI" are consistently higher performance. We also find "+tie" methods perform the best, indicating that tying language together is able to produce less ambiguous referring expressions.

Finally, we introduce another evaluation metric which measures the fraction of images for which an algorithm produces the same generated referring expression for multiple objects within the image. Obviously, a good referring expression generator should never produce the same expressions for two objects within the same image. Thus we would like this number to be as small as possible. The evaluation results under such metric are shown in Table 3.6. We find "+MMI" produces smaller number of duplicated expressions on both RefCOCO and RefCOCO+, while "+tie" helps generating even more different expressions. Our combined model "visdif+MMI+tie" performs the best under this metric.

3.4 A Joint Speaker-Listener-Reinforcer Model for Referring Expressions

we propose a unified model that *jointly* learns both the CNN-LSTM speaker and embedding-based listener models, for both the generation and comprehension tasks. Additionally, we add a discriminative reward-based *reinforcer* to guide the sampling of more discriminative expressions and further improve our final system. Instead of working independently, we let the speaker, listener, and reinforcer interact with each other, resulting in improved performance on both generation and comprehension tasks. Results evaluated on three standard, large-scale datasets verify that our proposed listener-speaker-reinforcer model significantly outperforms the state-of-the-art on both the comprehension task and the generation task, and automatic metrics.

3.4.1 Model

Our model is composed of three modules: speaker, listener, and reinforcer. During training, the speaker and listener are trained jointly so that they can benefit from each other and from the reinforcer. Because the reward function for the reinforcer is not differentiable, it is incorporated during training using a reinforcement learning policy gradient algorithm.

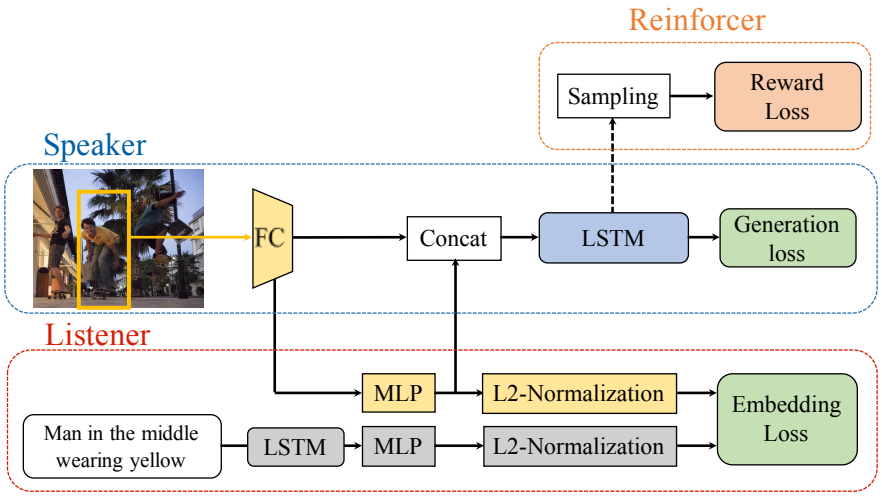


Figure 3.5: Framework: The Speaker is a CNN-LSTM model, which generates a referring expression for the target object. The Listener is a joint-embedding model learned to minimize the distance between paired object and expression representations. In addition, a Reinforcer module helps improve the speaker by sampling more discriminative (less ambiguous) expressions for training. The model is jointly trained with 3 loss functions – generation loss, embedding loss, and reward loss, thereby improving performance on both the comprehension and generation tasks.

3.4.1.1 Speaker

For our speaker module, we follow the previous state-of-the-art Mao et al. (2016); Yu et al. (2016b), and use a CNN-LSTM framework. Here, a pre-trained CNN model is used to define a visual representation for the target object and other visual context. Then, a Long-short term memory (LSTM) is used to generate the most likely expression given the visual representation.

According to Sec. 3.3.2, we extract the visual representation for the target object which is a concatenation of different features followed by a fully-connected layer fusing them together, $v_i = W_m[o_i, g_i, l_i, \delta v_i, \delta l_i] + b_m$. This joint feature is then fed into the LSTM for referring expression generation. During training we minimize the negative log-likelihood:

$$L_1^s(\theta) = -\sum_i \log P(r_i|o_i;\theta) \tag{3.4}$$

Note that the speaker can be modeled using any form of CNN-LSTM structure.

In Mao et al. (2016), Mao proposed to add a Maximum Mutual Information (MMI) constraint encouraging the generated expression to describe the target object better than the other objects

within the image (i.e., a ranking loss on objects). We generalize this idea to incorporate two triplet hinge losses composed of a positive match and two negative matches. Given a positive match (r_i, o_i), we sample the contrastive pair (r_j, o_i) where r_j is the expression describing some other object and pair (r_i, o_k) where o_k is some other object in the same image, then we optimize the following max-margin loss:

$$\begin{aligned} L_2^s(\theta) = \sum_i [&\lambda_1^s \max(0, M + \log P(r_i|o_k) - \log P(r_i|o_i)) \\ &+\lambda_2^s \max(0, M + \log P(r_j|o_i) - \log P(r_i|o_i))] \end{aligned} \tag{3.5}$$

The first term is from Mao et al. (2016), while the second term encourages that the target object to be better described by the true expression compared to expressions describing other objects in the image (i.e., a ranking loss on expressions).

3.4.1.2 Listener

We use a joint-embedding model to mimick the listener's behaviour. The purpose of this embedding model is to encode the visual information from the target object and semantic information from the referring expression into a joint embedding space that embeds vectors that are visually or semantically related closer together in the space. Here for referring expression comprehension task, given a referring expression representation, the listener embeds it into the joint space, then selects the closest object in the embedding space for the predicted target object.

As illustrated in Fig. 3.5, for our listener joint-embedding model (outlined by a red dashline), we use an LSTM to encode the input referring expression and the same visual representation as the speaker to encode the target object (thus connecting the speaker to the listener). We then add two MLPs (multi-layer perceptions) and two L2 normalization layers following each view, the object and the expression. Each MLP is composed of two fully connected layers with ReLU nonlinearities between them, serving to transform the object view and the expression view into a common embedding space. The inner-product of the two normalized representations is computed

as their similarity score $S(r, o)$ in the space. As a listener, we force the similarity on target object and referring expression pairs by applying a hinge loss over triplets, which consist of a positive match and two negative matches:

$$L^l(\theta) = \sum_i [\lambda_1^l \max(0, M + S(r_i, o_k) - S(r_i, o_i)) \\ + \lambda_2^l \max(0, M + S(r_j, o_i) - S(r_i, o_i))] \tag{3.6}$$

where the negative matches are randomly chosen from the other objects and expressions in the same image.

Note that the listener model is not limited to this particular triplet-based model. For example, Rohrbach et al. (2016a) computes a similarity score between every object for given referring expression, and minimizes the cross entropy of the SoftMax knowing the target object, which could also be applied here.

3.4.1.3 Reinforcer

Besides using the ground-truth pairs of target object and referring expression for training the speaker, we also use reinforcement learning to guide the speaker toward generating less ambiguous expressions (expressions that apply to the target object but not to other objects). This reinforcer module is composed of a discriminative reward function and performs a non-differentiable policy gradient update to the speaker.

Specifically, given the softmax output of the speaker's LSTM, we sample words according to the categorical distribution at each time step, resulting in a complete expression after sampling the <END> token. This sampling operation is non-differentiable as we do not know whether an expression is ambiguous or not until we feed it into a reward function. Therefore, we use policy gradient reinforcement learning to update the speaker's parameters. Here, the goal is to maximize the reward expectation $F(w_{1:T})$ under the distribution of $p(w_{1:T}; \theta)$ parameterized by the speaker,

i.e., $J = E_{p(w_{1:T})}[F]$. According to the policy gradient algorithm Williams (1992), we have

$$\nabla_\theta J = E_{p(w_{1:T})}[F(w_{1:T})\nabla_\theta \log p(w_{1:T}; \theta)], \tag{3.7}$$

Where $\log p(w_t)$ is defined by the softmax output. We then use this gradient to update our speaker model during training.

The only thing left is to choose a reward function that encourages the speaker to sample less ambiguous expressions. As illustrated in Fig. 3.5 (outlined in dashed orange), the reinforcer module learns a reward function using paired objects and expressions. We again use the same visual representation for the target object and use another LSTM to encode the expression representation. Rather than using two MLPs to encode each view as in the listener, here we concatenate the two views and feed them together into a MLP to learn a 1-d Logistic Regression score between 0 and 1. Trained with cross-entropy loss, the reward function computes a match score between an input object and expression. We use this score as the reward signal in Eqn. 3.7 for sampled expression and target object pairs. After training, the reward function is fixed to assist our joint speaker-listener system.

3.4.1.4 Joint Model

In this subsection, we describe some specifics of how our three modules (speaker, listener, reinforcer) are integrated into a joint framework (shown in Fig. 3.5). For the listener, we notice that the visual vector in the embedding space is learned to capture the neighbourhood vectors of referring expressions, thus making it aware of the listener's knowledge. Therefore, we take this MLP embedded vector as an additional input for the speaker, which encodes the listener based information. In Fig. 3.5, we use concatenation to jointly encode the standard visual representation of target object and this listener-aware representation and then feed them into speaker. Besides concatenation, the element-wise product or compact bilinear pooling can also be applied Fukui et al. (2016). During training, we sample the same triplets for both the speaker and listener, and

make the word embedding of the speaker and listener shared to reduce the number of parameters. For the reinforcer module, we do sentence sampling using the speaker's LSTM as shown in the top right of Fig. 3.5. Within each mini-batch, the sampled expressions for the target objects are fed into the reward function to obtain reward values.

The overall loss function is formulated as a multi-task learning problem:

$$\theta = \arg\min L_1^s(\theta) + L_2^s(\theta) + L^l(\theta) - \lambda^r J(\theta), \tag{3.8}$$

where λ^r is the weight on reward loss. The weights on the loss of speaker and listener are already included in Eqn. 3.5 and Eqn. 3.6.

3.4.1.5 Comprehension and Generation

For the comprehension task, at test time, we could use either the speaker or listener to select the target object given an input expression. Using the listener, we would embed the input expression into the learned embedding space and select the closest object as the predicted target. Using the speaker, we would generate expressions for each object within the image and then select the object whose generated expression best matches the input expression. Therefore, we utilize both modules by ensembling the speaker and listener predictions together to pick the most probable object given a referring expression.

$$\hat{o} = \arg\max_o P(r|o)S(o,r)^\lambda \tag{3.9}$$

Surprisingly, using the speaker alone (setting λ to 0) already achieves state-of-art results due to the joint training. Adding the listener further improves performance to more than 4% over previous state-of-art results.

For the generation task, we first let the speaker generate multiple expressions per object via beam search. We then use the listener to rerank these expressions and select the least ambiguous expression, which is similar to Andreas and Klein (2016). To fully utilize the listener's power in

generation, we propose to consider cross comprehension as well as the diversity of expressions by minimizing the potential:

$$
\begin{aligned}
E(r) &= \sum_i \theta_i(r_i) + \sum_{i,j} \theta_{i,j}(r_i, r_j) \\
\theta_i(r_i) &= -\log P(r_i|o_i) - \lambda_1 \log S(r_i, o_i) \\
&\quad + \lambda_2 \max_{j \neq i} \log S(r_i, o_j) \\
\theta_{i,j}(r_i, r_j) &= \lambda_3 I(r_i = r_j)
\end{aligned}
\tag{3.10}
$$

The first term and second term in unary potential measure how well the target object and generated expression match using the speaker and listener modules respectively, which was also used in Andreas and Klein (2016). The third term in unary potential measures the likelihood of the generated sentence for describing other objects in the same image. The pairwise potential penalize the same sentences being generated for different objects (encouraging diversity in generation). In this way, the expressions for every object in an image are jointly generated. Compared with the previous model that attempted to tie language generation of referring expressions together Yu et al. (2016b), the constraints in Eqn. 3.10 are more explicit and overall this works better to reduce ambiguity in the generated expressions.

3.4.2 Experiments

3.4.2.1 Comprehension Task

After training, we can either use the speaker or listener to perform the comprehension task. For the speaker that models $P(r|o)$, we feed every ground-truth object region within the given image to the speaker and select the most probable region for the expression as the comprehension result, i.e., $o^* = \text{argmax}_{o_i} p(r|o_i)$. For the listener, we directly compute the similarity score $S(r, o)$ between the proposal/object and expression and pick the object with the highest probability. For

		RefCOCO			RefCOCO+			RefCOCOg
		val	TestA	TestB	val	TestA	TestB	val
1	listener	77.48%	76.58%	78.94%	60.50%	61.39%	58.11%	71.12%
2	previous state-of-artYu et al. (2016b)	76.90%	75.60%	78.00%	58.94%	61.29%	56.24%	65.32%
3	baselineMao et al. (2016)	64.56%	63.20%	66.69%	47.78%	51.01%	44.24%	56.81%
4	**speaker**Yu et al. (2016b)	69.95%	68.59%	72.84%	52.63%	54.51%	50.02%	59.40%
5	**speaker**+listener	71.20%	69.98%	73.66%	54.23%	56.22%	52.46%	61.83%
6	**speaker**+reinforcer	71.88%	70.18%	73.01%	53.38%	56.50%	51.16%	61.91%
7	**speaker**+listener+reinforcer	72.46%	71.10%	74.01%	55.54%	57.46%	53.71%	64.07%
8	baseline+MMIMao et al. (2016)	72.28%	72.60%	73.39%	56.66%	60.01%	53.15%	63.31%
9	**speaker**+MMIYu et al. (2016b)	76.18%	74.39%	77.30%	58.94%	61.29%	56.24%	65.32%
10	**speaker**+listener+MMI	79.22%	77.78%	79.90%	61.72%	64.41%	58.62%	71.77%
11	**speaker**+reinforcer+MMI	78.38%	77.13%	79.53%	61.32%	63.99%	58.25%	67.06%
12	**speaker**+listener+reinforcer+MMI	**79.56%**	**78.95%**	**80.22%**	**62.26%**	**64.60%**	**59.62%**	**72.63%**
		RefCOCO (detected)			RefCOCO+ (detected)			RefCOCOg (detected)
		val	TestA	TestB	val	TestA	TestB	val
1	listener	-	71.63%	61.47%	-	57.33%	47.21%	56.18%
2	previous state-of-artYu et al. (2016b)	-	72.03%	63.08%	-	58.87%	47.70%	58.26%
3	baselineMao et al. (2016)	-	64.42%	56.75%	-	52.84%	42.68%	53.13%
4	**speaker**Yu et al. (2016b)	-	67.69%	60.16%	-	54.37%	45.00%	53.83%
5	**speaker**+listener	-	68.27%	61.00%	-	55.41%	45.65%	54.96%
6	**speaker**+reinforcer	-	69.12%	60.47%	-	55.45%	44.96%	55.64%
7	**speaker**+listener+reinforcer	-	69.15%	61.96%	-	55.97%	46.45%	57.03%
8	baseline+MMIMao et al. (2016)	-	68.73%	59.56%	-	58.15%	46.63%	57.23%
9	**speaker**+MMIYu et al. (2016b)	-	72.03%	63.08%	-	58.87%	47.70%	58.26%
10	**speaker**+listener+MMI	-	**72.95%**	63.10%	-	60.23%	48.11%	58.57%
11	**speaker**+reinforcer+MMI	-	72.34%	63.24%	-	59.36%	48.72%	58.70%
12	**speaker**+listener+reinforcer+MMI	-	72.88%	**63.43%**	-	**60.43%**	**48.74%**	**59.51%**

Table 3.7: Ablation study using the speaker module for the comprehension task (indicated in **bold**). Top half shows performance given ground truth bounding boxes for objects, bottom half performance using automatic object detectors to select potential objects. We find that adding listener and reinforcer modules to the speaker increases performance.

evaluation, we compute the intersection-over-union (IoU) of the comprehended region with the ground-truth object. If the IoU score of the predicted region is greater than 0.5, we consider this a correct comprehension.

To demonstrate the benefits of each module, we run ablation studies in Lines 3-12 of Table 3.7 (for speaker as comprehender) and in Lines 3-8 of Table 3.8 (for listener as comprehender) on all three datasets. Each row shows the results after adding a module during training. For some models that have both speaker and listener, we highlight the module being used for comprehension in bold. For example, "**speaker**+listener" means we use the speaker module of the joint model to do the comprehension task, while "speaker+**listener**" means we use the listener module for this task. Note our speaker module is implemented using the "visdif" model in Yu et al. (2016b) as mentioned in Section 3.4.1.1. Following previous work Mao et al. (2016)Yu et al.

		RefCOCO			RefCOCO+			RefCOCOg
		val	TestA	TestB	val	TestA	TestB	val
1	listener	77.48%	76.58%	78.94%	60.50%	61.39%	58.11%	71.12%
2	previous state-of-artYu et al. (2016b)	76.90%	75.60%	78.00%	58.94%	61.29%	56.24%	65.32%
3	speaker+**listener**	77.84%	77.50%	79.31%	60.97%	62.85%	58.58%	72.25%
4	speaker+**listener**+reinforcer	78.14%	76.91%	80.10%	61.34%	63.34%	58.42%	71.72%
5	**speaker+listener**+reinforcer (ensemble)	78.88%	78.01%	80.65%	61.90%	64.02%	59.19%	72.43%
6	speaker+**listener**+MMI	78.42%	78.45%	79.94%	61.48%	62.14%	58.91%	72.13%
7	speaker+**listener**+reinforcer+MMI	78.36%	77.97%	79.86%	61.33%	63.10%	58.19%	72.02%
8	**speaker+listener**+reinforcer+MMI (ensemble)	**80.36%**	**80.08%**	**81.73%**	**63.83%**	**65.40%**	**60.73%**	**74.19%**
		RefCOCO (detected)			RefCOCO+ (detected)			RefCOCOg (detected)
		val	TestA	TestB	val	TestA	TestB	val
1	listener	-	71.63%	61.47%	-	57.33%	47.21%	56.18%
2	previous state-of-artYu et al. (2016b)	-	72.03%	63.08%	-	58.87%	47.70%	58.26%
3	speaker+**listener**	-	72.23%	62.92%	-	59.61%	48.31%	57.38%
4	speaker+**listener**+reinforcer	-	72.65%	62.69%	-	58.68%	48.23%	58.32%
5	**speaker+listener**+reinforcer (ensemble)	-	72.78%	**64.38%**	-	59.80%	49.34%	**60.46%**
6	speaker+**listener**+MMI	-	72.95%	62.43%	-	58.68%	48.44%	57.34%
7	speaker+**listener**+reinforcer+MMI	-	72.94%	62.98%	-	58.68%	47.68%	57.72%
8	**speaker+listener**+reinforcer+MMI (ensemble)	-	**73.78%**	63.83%	-	**60.48%**	**49.36%**	59.84%

Table 3.8: Ablation study using listener or ensembled listener+speaker modules for the comprehension task (indicated in **bold**). Top half shows performance given ground truth bounding boxes for objects, bottom half performance using automatic object detectors to select potential objects. We find that jointly training with the speaker improves listener's performance and that adding the reinforcer module in an ensembled speaker+listener prediction performs the best.

(2016b)Nagaraja et al. (2016), we show the results trained with MMI and those trained w/o MMI on speakers. We compare our models with the "baseline" model Mao et al. (2016) (Line 3, 8 in Table 3.7), the pure listener model (Line 1), and previous state-of-art results (Line 2) achieved in Nagaraja et al. (2016)Yu et al. (2016b).

First, we evaluate the performance of the speaker on the comprehension task (Table 3.7). We observe all speaker models trained with MMI outperform w/o MMI. We also find that the speaker can be improved by joint training with the listener module and by incorporating the reinforcer module. With MMI ranking, the speaker learned with joint training (Line 12) is able to outperform the pure listener by around 2% on all three datasets, which already achieves state-of-art performance on the comprehension task.

Second, we show evaluations using variations of the listener module or ensembled listener+speaker modules (indicated in **bold**) for the comprehension task in Table 3.8. We note that the listener generally works better than speaker for the comprehension task, indicating that the deterministic joint-embedding model is more suitable for this task than the speaker model – similar results were observed in Rohrbach et al. (2016a). While the reinforcer module seems not to be as ef-

Figure 3.6: Example comprehension results based on detection. Green box shows the ground-truth region, blue box shows correct comprehension using our "speaker+listener+reinforcer+MMI" model, and red box shows incorrect comprehension. We use top two rows to show some correct comprehensions and bottom two rows to show some incorrect ones.

fective as the speaker, we still find that the joint training always brings additional discriminative benefits to the listener module, resulting in improved performance (compare Line 3-8 with Line 1 in Table 3.8). Ensembling the speaker and listener together achieves the best results overall.

Both of the above experiments analyze comprehension performance given ground truth bounding boxes for potential comprehension objects, where the algorithm must select which of these objects is being described. This provides an analysis of comprehension performance independent of any particular object detection method. Additionally, we also show results using an object detector to automatically select regions for consideration during comprehension in the bottom half of each table (Tables 3.7 and 3.8). As our detection algorithm, we use current state of the art detector in effectiveness and speed, SSD Liu et al. (2016), trained on a subset of the MS COCO train+val dataset, removing images that are in the test splits of RefCOCO or RefCOCO+

	RefCOCO				RefCOCO+				RefCOCOg	
	Test A		Test B		Test A		Test B		val	
	Meteor	CIDEr	Meteor	CIDEr	Meteor	CIDEr	Meteor	CIDEr	Meteor	CIDEr
speaker+tie Yu et al. (2016b)	0.283	0.681	0.320	1.273	0.204	0.499	0.196	0.683	-	-
baseline+MMI	0.243	0.615	0.300	1.227	0.199	0.462	0.189	0.679	0.149	0.585
speaker+MMI	0.260	0.679	0.319	1.276	0.202	0.475	0.196	0.683	0.147	0.573
speaker+listener+MMI	0.268	0.704	0.327	1.303	**0.208**	**0.496**	0.201	0.697	0.150	0.589
speaker+reinforcer+MMI	0.266	0.702	0.323	1.291	0.204	0.482	0.197	0.692	0.151	**0.602**
speaker+listener+reinforcer+MMI	**0.268**	0.697	**0.329**	**1.323**	0.204	0.494	**0.202**	**0.709**	**0.154**	0.592
baseline+MMI+rerank	0.280	0.729	0.329	1.285	0.204	0.484	0.205	0.730	**0.160**	0.654
speaker+MMI+rerank	0.287	0.745	0.334	1.295	0.208	0.490	0.213	0.712	0.156	0.653
speaker+listener+MMI+rerank	0.293	0.763	0.337	1.306	0.211	0.500	**0.221**	0.734	0.159	0.650
speaker+reinforcer+MMI+rerank	0.291	0.748	0.337	1.311	0.207	0.499	0.215	0.729	0.158	0.653
speaker+listener+reinforcer+MMI+rerank	**0.296**	**0.775**	**0.340**	**1.320**	**0.213**	**0.520**	0.215	**0.735**	0.159	**0.662**

Table 3.9: Ablation study for generation using automatic evaluation.

	RefCOCO		RefCOCO+	
	Test A	Test B	Test A	Test B
speaker+tie Yu et al. (2016b)	71.40%	76.14%	57.17%	47.92%
speaker+MMI Yu et al. (2016b)	68.82%	75.50%	53.57%	47.88%
speaker+listener+MMI	73.23%	76.08%	53.83%	49.19%
speaker+reinforcer+MMI	71.08%	76.09%	55.16%	48.50%
speaker+listener+reinforcer+MMI	74.08%	76.44%	56.92%	53.23%
speaker+listener+reinforcer+MMI+rerank	**76.95%**	**78.10%**	**58.85%**	**58.20%**

Table 3.10: Human Evaluations on generation.

or in the validation split of RefCOCOg. We empirically select 0.30 as the confidence threshold for detection output. While performance drops somewhat due to the strong dependence of "vis-dif" model on detection Yu et al. (2016b), the overall improvements brought by each module are consistent with using ground-truth objects, showin the robustness of our joint model. Fig. 3.6 shows some comprehension results using our full model.

3.4.2.2 Generation Task

For the generation task, we evaluate variations on the speaker module. Evaluating the generation is not as simple as comprehension. In image captioning, BLEU, ROUGE, METEOR and CIDEr are common automatic metrics and have been widely used as standard evaluations. We show automatic evaluation using the METEOR and CIDEr metrics for generation in Table 3.9 where "+rerank" denotes models incorporating the reranking mechanism and global optimization over all objects (Eqn. 3.10). To computer CIDEr robustly, we collect more expressions for

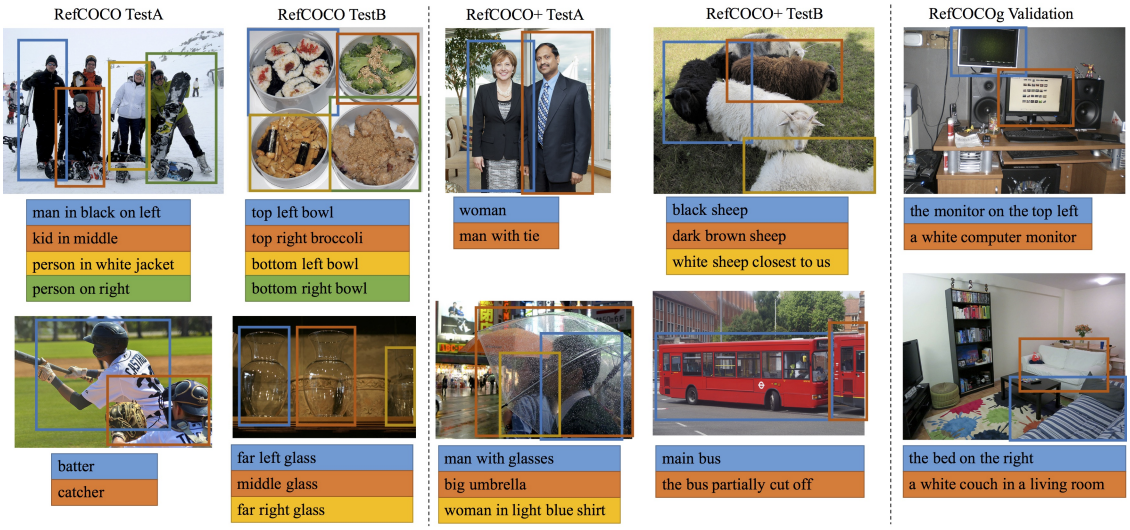


Figure 3.7: Joint generation examples using "speaker+listener+reinforcer+MMI+rerank". Each sentence shows the generated expression for one of the depicted objects (color coded to indicate correspondence)

objects in the test sets for RefCOCO and RefCOCO+, obtaining 10.1 and 9.4 expressions respectively on average per object. For RefCOCOg we use the original expressions released with the dataset which may be limited, but we still show its performance for completeness. We choose the "speaker+tie" model in Yu et al. (2016b) as reference, which learns to tie the expression generation together and achieves state-of-art performance. Generally we find that the speaker in jointly learned models achieves higher scores than the single speaker under both metrics across datasets. Such improvements are observed under both settings without "+rerank" or with "+rerank".

In addition, since previous work Yu et al. (2016b) has found that these metrics do not always agree well with human judgments for referring expressions, we also run a human evaluation on the same set of objects as Yu et al. (2016b) for RefCOCO and RefCOCO+. Here we ask Turkers to click on the referred object given a generated expression. These results are shown in Table. 3.10. Results indicate the ablated benefits brought by each module, and ultimately the "+rerank" of our joint model achieves the best performance.

We show the joint expression generation using our full model with "+rerank" in Fig. 3.7. As observed, the expressions of every target object are considered together, where each of them is meant to be relevant to the target object and irrelevant to the other objects.

3.5 Modular Attention Network for Referring Expression Comprehension

As mentioned above, most previous work uses a simple concatenation of all features (target object feature, location feature and context feature) as input and a single LSTM to encode/decode the whole expression, ignoring the variance among different types of referring expressions. Depending on what is distinctive about a target object, different kinds of information might be mentioned in its referring expression. For example, if the target object is a red ball among 10 black balls then the referring expression may simply say "the red ball". If that same red ball is placed among 3 other red balls then location-based information may become more important, e.g., "red ball on the right". Or, if there were 100 red balls in the scene then the ball's relationship to other objects might be the most distinguishing information, e.g., "red ball next to the cat". Therefore, it is natural and intuitive to think about the comprehension model as a modular network, where different visual processing modules are triggered based on what information is present in the referring expression.

Modular networks have been successfully applied to address other tasks such as (visual) question answering Andreas et al. (2016a,b), visual reasoning Hu et al. (2017a); Johnson et al. (2017b), relationship modeling Hu et al. (2017b), and multi-task reinforcement learning Andreas et al. (2017). To the best our knowledge, we present the first modular network for the general referring expression comprehension task. Moreover, these previous work typically relies on an off-the-shelf language parser Socher et al. (2013) to parse the query sentence/question into different components and dynamically assembles modules into a model addressing the task. However, the external parser could raise parsing errors and propagate them into model setup, adversely effecting performance.

Therefore, we propose a modular network for referring expression comprehension - Modular Attention Network (MAttNet) Yu et al. (2018) - that takes a natural language expression as input and softly decomposes it into three phrase embeddings. These embeddings are used to trigger three separate visual modules (for subject, location, and relationship comprehension, each with a different attention model) to compute matching scores, which are finally combined into an overall

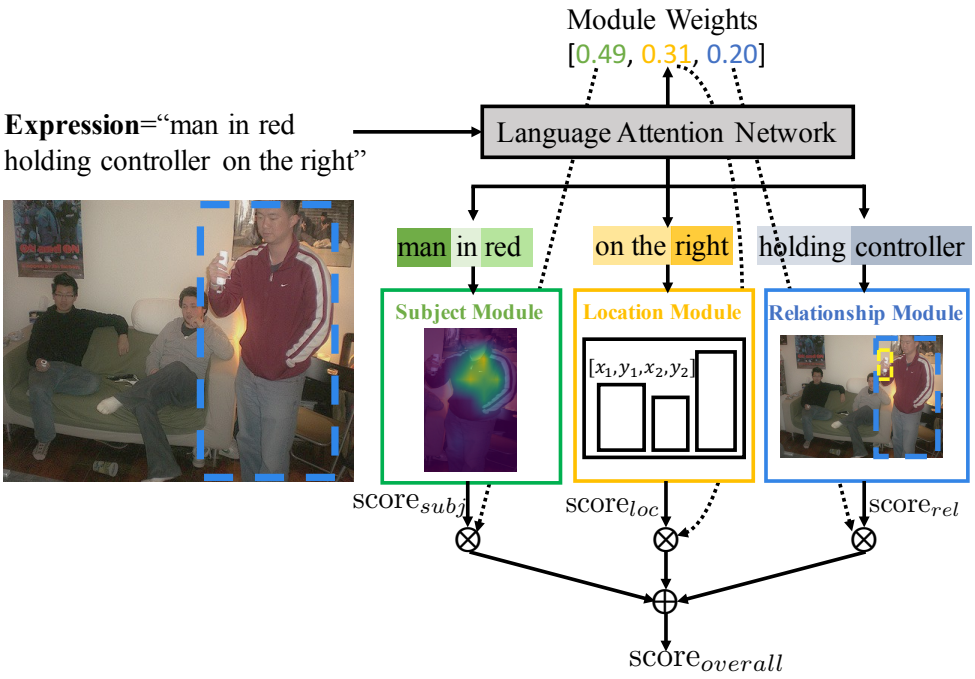


Figure 3.8: Modular Attention Network (MAttNet). Given an expression, we attentionally parse it into three phrase embeddings, which are input to three visual modules that process the described visual region in different ways and compute individual matching scores. An overall score is then computed as a weighted combination of the module scores.

region score based on the module weights. Our model is illustrated in Fig. 3.8. There are 3 main novelties in MAttNet.

First, MAttNet is designed for general referring expressions. It consists of 3 modules: subject, location and relationship. As in Kazemzadeh et al. (2014), a referring expression could be parsed into 7 attributes: category name, color, size, absolute location, relative location, relative object and generic attribute. MAttNet covers all of them. The subject module handles the category name, color and other attributes, the location module handles both absolute and (some) relative location, and the relationship module handles subject-object relations. Each module has a different structure and learns the parameters within its own modular space, without affecting the others.

Second, MAttNet learns to parse expressions automatically through a soft attention based mechanism, instead of relying on an external language parser Socher et al. (2013); Kazemzadeh et al. (2014). We show that our learned "parser" attends to the relevant words for each module

and outperforms an off-the-shelf parser by a large margin. Additionally, our model computes module weights which are adaptive to the input expression, measuring how much each module should contribute to the overall score. Expressions like "red cat" will have larger subject module weights and smaller location and relationship module weights, while expressions like "woman on left" will have larger subject and location module weights.

Third, we apply different visual attention techniques in the subject and relationship modules to allow relevant attention on the described image portions. In the subject module, soft attention attends to the parts of the object itself mentioned by an expression like "man in red shirt" or "man with yellow hat". We call this "in-box" attention. In contrast, in the relationship module, hard attention is used to attend to the relational objects mentioned by expressions like "cat on chair" or "girl holding frisbee". Here the attention focuses on "chair" and "frisbee" to pinpoint the target object "cat" and "girl". We call this "out-of-box" attention. We demonstrate both attentions play important roles in improving comprehension accuracy.

During training, the only supervision is object proposal, referring expression pairs, (o_i, r_i), and all of the above are automatically learned in an end-to-end unsupervised manner, including the word attention, module weights, soft spatial attention, and hard relative object attention.

We demonstrate MAttNet has significantly superior comprehension performance over all state-of-the-art methods, achieving $\sim 10\%$ improvements on bounding-box localization and almost doubling precision on pixel segmentation.

3.5.1 Model

MAttNet is composed of a language attention network plus visual subject, location, and relationship modules. Given a candidate object o_i and referring expression r, we first use the language attention network to compute a soft parse of the referring expression into three components (one for each visual module) and map each to a phrase embedding. Second, we use the three visual modules (with unique attention mechanisms) to compute matching scores for o_i to their

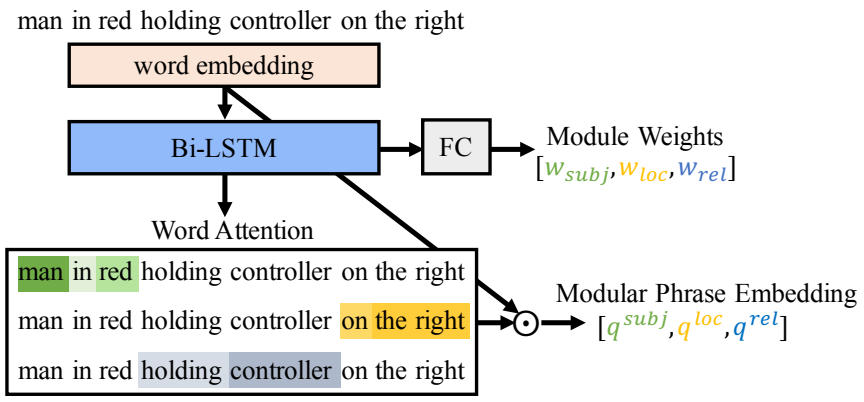


Figure 3.9: Language Attention Network

respective embeddings. Finally, we take a weighted combination of these scores to get an overall matching score, measuring the compatibility between o_i and r.

3.5.1.1 Language Attention Network

Instead of using an external language parser Socher et al. (2013); Andreas et al. (2016b,a) or pre-defined templates Kazemzadeh et al. (2014) to parse the expression, we propose to learn to attend to the relevant words automatically for each module, similar to Hu et al. (2017b). Our language attention network is shown in Fig. 3.9. For a given expression $r = \{u_t\}_{t=1}^{T}$, we use a bi-directional LSTM to encode the context for each word. We first embed each word u_t into a vector e_t using an one-hot word embedding, then a bi-directional LSTM-RNN is applied to encode the whole expression. The final hidden representation for each word is the concatenation of the hidden vectors in both directions:

$$e_t = \text{embedding}(u_t)$$

$$\vec{h}_t = \vec{\text{LSTM}}(e_t, \vec{h}_{t-1})$$

$$\overleftarrow{h}_t = \overleftarrow{\text{LSTM}}(e_t, \overleftarrow{h}_{t+1})$$

$$h_t = [\vec{h}_t, \overleftarrow{h}_t].$$

Given $H = \{h_t\}_{t=1}^{T}$, we apply three trainable vectors f_m where $m \in \{\text{subj}, \text{loc}, \text{rel}\}$, computing the attention on each word Yang et al. (2016) for each module:

$$a_{m,t} = \frac{\exp\left(f_m^T h_t\right)}{\sum_{k=1}^{T} \exp\left(f_m^T h_k\right)}$$

The weighted sum of word embeddings is used as the modular phrase embedding:

$$q^m = \sum_{t=1}^{T} a_{m,t} e_t$$

Different from relationship detection Hu et al. (2017b) where phrases are always decomposed as (Subject, Preposition/Verb, Object) triplets, referring expressions have no such well-posed structure. For example, expressions like "smiling boy" only contain language relevant to the subject module, while expressions like "man on left" are relevant to the subject and location modules, and "cat on the chair" are relevant to the subject and relationship modules. To handle this variance, we compute 3 module weights for the expression, weighting how much each module contributes to the expression-object score. We concatenate the first and last hidden vectors from H which memorizes both structure and semantics of the whole expression, then use another fully-connected (FC) layer to transform it into 3 module weights:

$$[w_{subj}, w_{loc}, w_{rel}] = \text{softmax}(W_m^T [h_0, h_T] + b_m)$$

3.5.1.2 Visual Modules

While most previous work Yu et al. (2016b, 2017b); Mao et al. (2016); Nagaraja et al. (2016) evaluates CNN features for each region proposal/candidate object, we use Faster R-CNN Ren et al. (2015) as the backbone net for a faster and more principled implementation. Additionally, we use ResNet He et al. (2016) as our main feature extractor, but also provide compar-

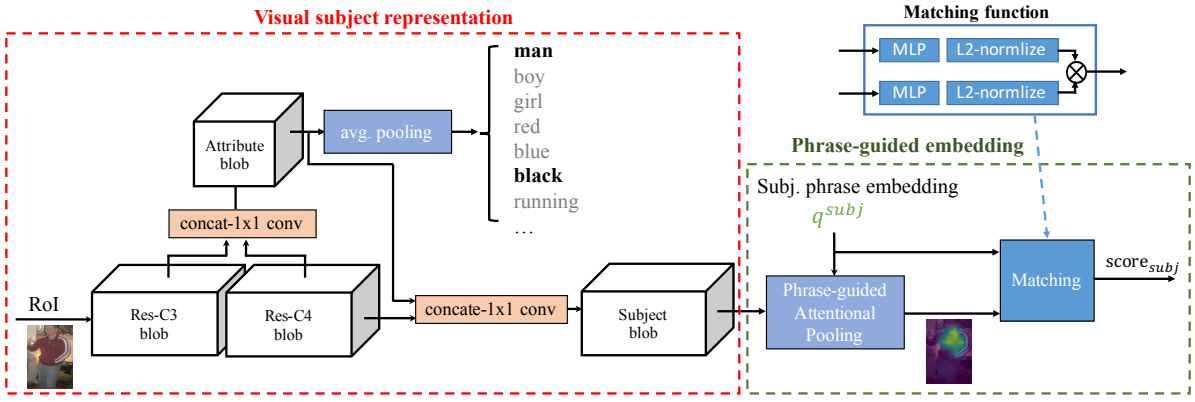


Figure 3.10: The subject module is composed of a visual subject representation and phrase-guided embedding. An attribute prediction branch is added after the ResNet-C4 stage and the 1x1 convolution output of attribute prediction and C4 is used as the subject visual representation. The subject phrase embedding attentively pools over the spatial region and feeds the pooled feature into the matching function.

isons to previous methods using the same VGGNet features Simonyan and Zisserman (2014) (in Sec. 3.5.2.1).

Given an image and a set of candidates o_i, we run Faster R-CNN to extract their region representations. Specifically, we forward the whole image into Faster R-CNN and crop the C3 feature (last convolutional output of 3rd-stage) for each o_i, following which we further compute the C4 feature (last convolutional output of 4th-stage). In Faster R-CNN, C4 typically contains higher-level visual cues for category prediction, while C3 contains relatively lower-level cues including colors and shapes for proposal judgment, making both useful for our purposes. In the end, we compute the matching score for each o_i given each modular phrase embedding, i.e., $S(o_i|q^{subj})$, $S(o_i|q^{loc})$ and $S(o_i|q^{rel})$.

Subject Module: Our subject module is illustrated in Fig. 3.10. Given the C3 and C4 features of a candidate o_i, we forward them to two tasks. The first is attribute prediction, helping produce a representation that can understand appearance characteristics of the candidate. The second is the phrase-guided attentional pooling to focus on relevant regions within object bounding boxes.

Attributes are frequently used in referring expressions to differentiate between objects of the same category, e.g. “woman in red” or “the fuzzy cat”. Inspired by previous work Yao et al.

(2016); Wu et al. (2017); You et al. (2016); Liu et al. (2017b); Su et al. (2017), we add an attribute prediction branch in our subject module. While preparing the attribute labels in the training set, we first run a template parser Kazemzadeh et al. (2014) to obtain color and generic attribute words, with low-frequency words removed. We combine both C3 and C4 for predicting attributes as both low and high-level visual cues are important. The concatenation of C3 and C4 is followed with a 1×1 convolution to produce an attribute feature blob. After average pooling, we get the attribute representation of the candidate region. A binary cross-entropy loss is used for multi-attribute classification:

$$L_{subj}^{attr} = \lambda_{attr} \sum_i \sum_j w_j^{attr} [\log(p_{ij}) + (1 - y_{ij}) \log(1 - p_{ij})]$$

where $w_j^{attr} = 1/\sqrt{\text{freq}_{attr}}$ weights the attribute labels, easing unbalanced data issues. During training, only expressions with attribute words go through this branch.

The subject description varies depending on what information is most salient about the object. Take people for example. Sometimes a person is described by their accessories, e.g., "girl in glasses"; or sometimes particular clothing items may be mentioned, e.g., "woman in white pants". Thus, we allow our subject module to localize relevant regions within a bounding box through "in-box" attention. To compute spatial attention, we first concatenate the attribute blob and C4, then use a 1×1 convolution to fuse them into a subject blob, which consists of spatial grid of features $V \in R^{d \times G}$, where $G = 14 \times 14$. Given the subject phrase embedding q^{subj}, we compute its attention on each grid location:

$$H_a = \tanh(W_v V + W_q q^{subj})$$
$$a^v = \text{softmax}(w_{h,a}^T H_a)$$

The weighted sum of V is the final subject visual representation for the candidate region o_i:

$$\widetilde{v}_i^{subj} = \sum_{i=1}^{G} a_i^v v_i$$

We measure the similarity between the subject representation $\widetilde{v}_i^{subj}$ and phrase embedding q_{subj} using a matching function, i.e, $S(o_i|q^{subj}) = F(\widetilde{v}_i^{subj}, q^{subj})$. As shown in top-right of Fig. 3.10, it consists of two MLPs (multi-layer perceptions) and two L2 normalization layers following each input. Each MLP is composed of two fully connected layers with ReLU activations, serving to transform the visual and phrase representation into a common embedding space. The inner-product of the two l2-normalized representations is computed as their similarity score. The same matching function is used to compute the location score $S(o_i|q^{loc})$, and relationship score $S(o_i|q^{rel})$.

Location Module: Our location module is shown in Fig. 3.11. Location is frequently used

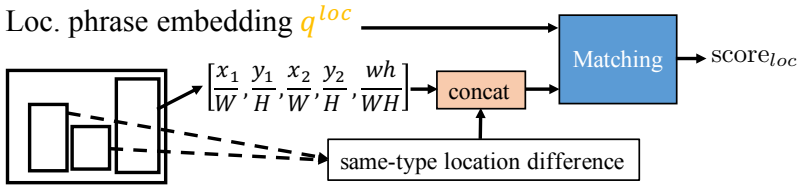


Figure 3.11: Location Module

in referring expressions with about 41% expressions from RefCOCO and 36% expressions from RefCOCOg containing absolute location words Kazemzadeh et al. (2014), e.g. "cat on the right" indicating the object location in the image. Following previous work Yu et al. (2016b, 2017b), we use a 5-d vector l_i to encode the top-left position, bottom-right position and relative area to the image for the candidate object, i.e., $l_i = [\frac{x_{tl}}{W}, \frac{y_{tl}}{H}, \frac{x_{br}}{W}, \frac{y_{br}}{H}, \frac{w \cdot h}{W \cdot H}]$.

Additionally, expressions like "dog in the middle" and "second left person" imply relative positioning among objects of the same category. We encode the relative location representation of a candidate object by choosing up to five surrounding objects of the same category and calculating their offsets and area ratio, i.e., $\delta l_{ij} = [\frac{[\triangle x_{tl}]_{ij}}{w_i}, \frac{[\triangle y_{tl}]_{ij}}{h_i}, \frac{[\triangle x_{br}]_{ij}}{w_i}, \frac{[\triangle y_{br}]_{ij}}{h_i}, \frac{w_j h_j}{w_i h_i}]$. The final location representation for the target object is:

$$\widetilde{l}_i^{loc} = W_l[l_i; \delta l_i] + b_l$$

and the location module matching score between o_i and q^{loc} is $S(o_i|q^{loc}) = F(\widetilde{l}_i^{loc}, q^{loc})$.

Relationship Module

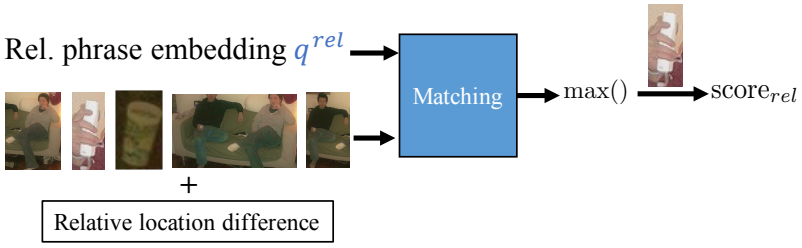


Figure 3.12: Relationship Module

While the subject module deals with "in-box" details about the target object, some other expressions may involve its relationship with other "out-of-box" objects, e.g., "cat on chaise lounge". The relationship module is used to address these cases. As in Fig. 3.12, given a candidate object o_i we first look for its surrounding (up-to-five) objects o_{ij} regardless of their categories. We use the average-pooled C4 feature as the appearance feature v_{ij} of each supporting object. Then, we encode their offsets to the candidate object via

$$\delta m_{ij} = [\frac{[\triangle x_{tl}]_{ij}}{w_i}, \frac{[\triangle y_{tl}]_{ij}}{h_i}, \frac{[\triangle x_{br}]_{ij}}{w_i}, \frac{[\triangle y_{br}]_{ij}}{h_i}, \frac{w_j h_j}{w_i h_i}].$$

The visual representation for each surrounding object is then:

$$\widetilde{v}_{ij}^{rel} = W_r[v_{ij}; \delta m_{ij}] + b_r$$

We compute the matching score for each of them with q^{rel} and pick the highest one as the relationship score, i.e.,

$$S(o_i|q^{rel}) = \max_{j \neq i} F(\widetilde{v}_{ij}^{rel}, q^{rel})$$

This can be regarded as weakly-supervised Multiple Instance Learning (MIL) which is similar to Hu et al. (2017b); Nagaraja et al. (2016).

			RefCOCO			RefCOCO+			RefCOCOg		
		feature	val	testA	testB	val	testA	testB	val*	val	test
1	Mao Mao et al. (2016)	vgg16	-	63.15	64.21	-	48.73	42.13	62.14	-	-
2	Varun Nagaraja et al. (2016)	vgg16	76.90	75.60	78.00	-	-	-	-	-	68.40
3	Luo Luo and Shakhnarovich (2017)	vgg16	-	74.04	73.43	-	60.26	55.03	65.36	-	-
4	CMN Hu et al. (2017b)	vgg16-frcn	-	-	-	-	-	-	69.30	-	-
5	Speaker/visdif Yu et al. (2016b)	vgg16	76.18	74.39	77.30	58.94	61.29	56.24	59.40	-	-
6	Listener Yu et al. (2017b)	vgg16	77.48	76.58	78.94	60.50	61.39	58.11	71.12	69.93	69.03
7	**Speaker**+Listener+Reinforcer	vgg16	79.56	78.95	80.22	62.26	64.60	59.62	72.63	71.65	71.92
8	Speaker+**Listener**+Reinforcer	vgg16	78.36	77.97	79.86	61.33	63.10	58.19	72.02	71.32	71.72
9	MAttN:subj(+attr)+loc(+dif)+rel	vgg16	80.94	79.99	82.30	63.07	65.04	61.77	73.08	73.04	72.79
10	MAttN:subj(+attr)+loc(+dif)+rel	res101-frcn	83.54	82.66	84.17	68.34	69.93	65.90	-	76.63	75.92
11	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-frcn	**85.65**	**85.26**	**84.57**	**71.01**	**75.13**	**66.17**	-	**78.10**	**78.12**

Table 3.11: Comparison with state-of-the-art approaches on ground-truth MS COCO regions.

3.5.1.3 Loss Function

The overall weighted matching score for candidate object o_i and expression r is:

$$S(o_i|r) = w_{subj}S(o_i|q^{subj}) + w_{loc}S(o_i|q^{loc}) + w_{rel}S(o_i|q^{rel}). \tag{3.11}$$

During training, for each given positive pair of (o_i, r_i), we randomly sample two negative pairs (o_i, r_j) and (o_k, r_i), where r_j is the expression describing some other object and o_k is some other object in the same image, to calculate a combined hinge loss,

$$L_{rank} = \sum_i [\lambda_1 \text{max}(0, \Delta + S(o_i|r_j) - S(o_i|r_i)) + \lambda_2 \text{max}(0, \Delta + S(o_k|r_i) - S(o_i|r_i))].$$

The overall loss incorporates both attributes and ranking loss: $L = L_{subj}^{attr} + L_{rank}$.

3.5.2 Experiments

3.5.2.1 Results: Referring Expression Comprehension

Given a test image, I, with a set of proposals/objects, $O = \{o_i\}_{i=1}^N$, we use Eqn. 3.11 to compute the matching score $S(o_i|r)$ for each proposal/object given the input expression r, and pick the one with the highest score. For evaluation, we compute the intersection-over-union (IoU) of the selected region with the ground-truth bounding box, considering IoU > 0.5 a correct comprehension.

		RefCOCO			RefCOCO+			RefCOCOg	
		val	testA	testB	val	testA	testB	val	test
1	Matching:subj+loc	79.14	79.42	80.42	62.17	63.53	59.87	70.45	70.92
2	MAttN:subj+loc	79.68	80.20	81.49	62.71	64.20	60.65	72.12	72.62
3	MAttN:subj+loc(+dif)	82.06	81.28	83.20	64.84	65.77	64.55	75.33	74.46
4	MAttN:subj+loc(+dif)+rel	82.54	81.58	83.34	65.84	66.59	65.08	75.96	74.56
5	MAttN:subj(+attr)+loc(+dif)+rel	83.54	82.66	84.17	68.34	69.93	65.90	76.63	75.92
6	MAttN:subj(+attr+attn)+loc(+dif)+rel	**85.65**	**85.26**	**84.57**	**71.01**	**75.13**	**66.17**	**78.10**	**78.12**
7	parser+MAttN:subj(+attr+attn)+loc(+dif)+rel	80.20	79.10	81.22	66.08	68.30	62.94	73.82	73.72

Table 3.12: Ablation study of MAttNet using different combination of modules. The feature used here is res101-frcn.

			RefCOCO			RefCOCO+			RefCOCOg	
		detector	val	testA	testB	val	testA	testB	val	test
1	**Speaker**+Listener+Reinforcer	res101-frcn	69.48	73.71	64.96	55.71	60.74	48.80	60.21	59.63
2	Speaker+**Listener**+Reinforcer	res101-frcn	68.95	73.10	64.85	54.89	60.04	49.56	59.33	59.21
3	Matching:subj+loc	res101-frcn	72.28	75.43	67.87	58.42	61.46	52.73	64.15	63.25
4	MAttN:subj+loc	res101-frcn	72.72	76.17	68.18	58.70	61.65	53.41	64.40	63.74
5	MAttN:subj+loc(+dif)	res101-frcn	72.96	76.61	68.20	58.91	63.06	55.19	64.66	63.88
6	MAttN:subj+loc(+dif)+rel	res101-frcn	73.25	76.77	68.44	59.45	63.31	55.68	64.87	64.01
7	MAttN:subj(+attr)+loc(+dif)+rel	res101-frcn	74.51	77.81	68.39	62.13	66.33	55.75	65.33	65.19
8	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-frcn	76.40	80.43	69.28	64.93	70.26	56.00	**66.67**	67.01
9	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-mrcn	**76.65**	**81.14**	**69.99**	**65.33**	**71.62**	**56.02**	66.58	**67.27**

Table 3.13: Ablation study of MAttNet on fully-automatic comprehension task using different combination of modules. The features used here are res101-frcn, except the last row using res101-mrcn.

First, we compare our model with previous methods using COCO's ground-truth object bounding boxes as proposals. Results are shown in Table. 3.11. As all of the previous methods (Line 1-8) used a 16-layer VGGNet (vgg16) as the feature extractor, we run our experiments using the same feature for fair comparison. Note the flat fc7 is a single 4096-dimensional feature which prevents us from using the phrase-guided attentional pooling in Fig. 3.10, so we use average pooling for subject matching. Despite this, our results (Line 9) still outperform all previous state-of-the-art methods. After switching to the res101-based Faster R-CNN (res101-frcn) representation, the comprehension accuracy further improves another $\sim 3\%$ (Line 10). Note our Faster R-CNN is pre-trained on COCO's training images, excluding those in RefCOCO, RefCOCO+, and RefCOCOg's validation+testing. Thus no training images are seen during our evaluation[2].

[2] Such constraint forbids us to evaluate on RefCOCOg's val* using the res101-frcn feature in Table 3.11.

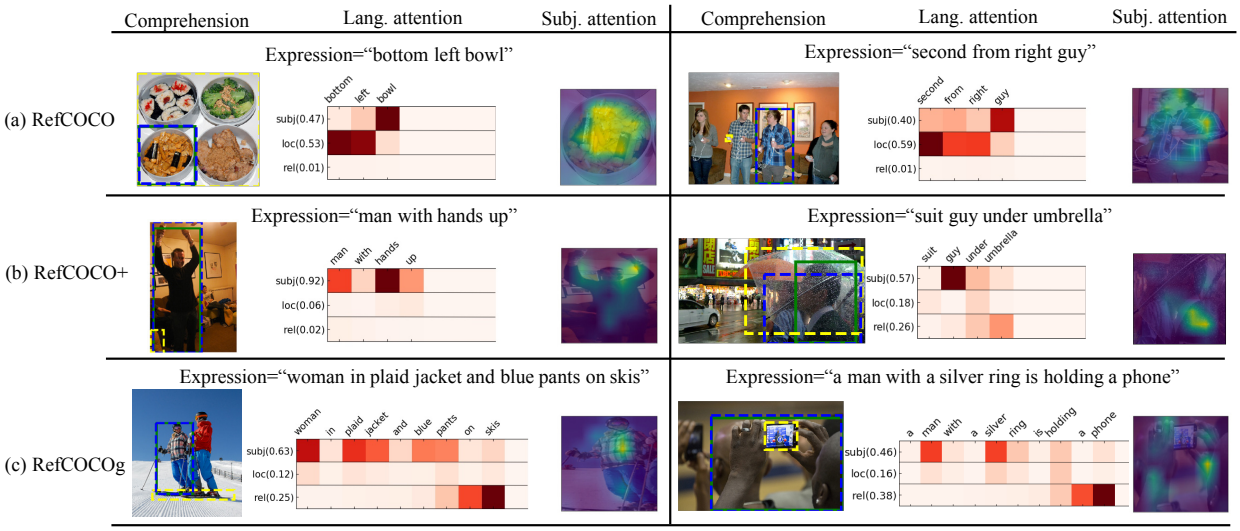


Figure 3.13: Examples of fully automatic comprehension. The blue dotted boxes show our prediction with the relative regions in yellow dotted boxes, and the green boxes are the ground-truth. The word attention is multiplied by module weight.

Our full model (Line 11) with phrase-guided attentional pooling achieves the highest accuracy over all others by a large margin.

Second, we study the benefits of each module of MAttNet by running ablation experiments (Table. 3.12) with the same res101-frcn features. As a baseline, we use the concatenation of the regional visual feature and the location feature as the visual representation and the last hidden output of LSTM-encoded expression as the language representation, then feed them into the matching function to obtain the similarity score (Line 1). Compared with this, a simple two-module MAttNet using the same features (Line 2) already outperforms the baseline, showing the advantage of modular learning. Line 3 shows the benefit of encoding location (Fig. 3.11). After adding the relationship module, the performance further improves (Line 4). Lines 5 and Line 6 show the benefits brought by the attribute sub-branch and the phrase-guided attentional pooling in our subject module. We find the attentional pooling (Line 6) greatly improves on the person category (testA of RefCOCO and RefCOCO+), demonstrating the advantage of modular attention on understanding localized details like "girl with red hat".

Third, we tried training our model using 3 hard-coded phrases from a template language parser Kazemzadeh et al. (2014), shown in Line 7 of Table. 3.12, which is $\sim$5% lower than our

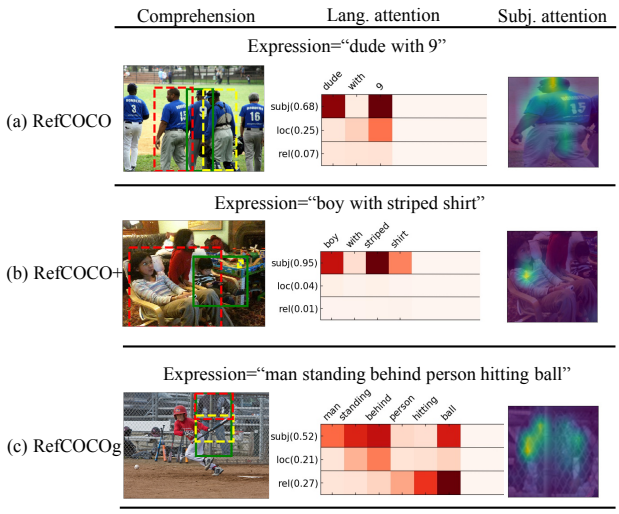


Figure 3.14: Examples of incorrect comprehensions. Red dotted boxes show our wrong prediction.

end-to-end model (Line 6). The main reason for this drop is errors made by the external parser which is not tuned for referring expressions.

Fourth, we show results using automatically detected objects from Faster R-CNN, providing an analysis of fully automatic comprehension performance. Table. 3.13 shows the ablation study of fully-automatic MAttNet. While performance drops due to detection errors, the overall improvements brought by each module are consistent with Table. 3.12, showing the robustness of MAttNet. Our results also outperform the state-of-the-art Yu et al. (2017b) (Line 1,2) with a big margin. Besides, we show the performance when using the detector branch of Mask R-CNN He et al. (2017) (res101-mrcn) in Line 9, whose results are even better than using Faster R-CNN.

Finally, we show some example visualizations of comprehension using our full model in Fig. 3.13 as well as visualizations of the attention predictions. We observe that our language model is able to attend to the right words for each module even though it is learned in a weakly-supervised manner. We also observe the expressions in RefCOCO and RefCOCO+ describe the location or details of the target object more frequently while RefCOCOg mentions the relationship between target object and its surrounding object more frequently, which accords with the dataset property. Note that for some complex expressions like "woman in plaid jacket and blue pants on skis" which contains several relationships (last row in Fig. 3.13), our language model is

RefCOCO

Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	val	42.99	33.24	22.75	12.11	2.23	45.18
MAttNet	res101-mrcn	val	**75.16**	**72.55**	**67.83**	**54.79**	**16.81**	**56.51**
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	testA	42.99	33.59	23.69	12.94	2.44	45.69
MAttNet	res101-mrcn	testA	**79.55**	**77.60**	**72.53**	**59.01**	**13.79**	**62.37**
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	testB	44.99	32.21	22.69	11.84	2.65	45.57
MAttNet	res101-mrcn	testB	**68.87**	**65.06**	**60.02**	**48.91**	**21.37**	**51.70**

RefCOCO+

Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	val	20.52	14.02	8.46	3.77	0.62	29.86
MAttNet	res101-mrcn	val	**64.11**	**61.87**	**58.06**	**47.42**	**14.16**	**46.67**
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	testA	21.22	14.43	8.99	3.91	0.49	30.48
MAttNet	res101-mrcn	testA	**70.12**	**68.48**	**63.97**	**52.13**	**12.28**	**52.39**
D+RMI+DCRF Liu et al. (2017a)	res101-DeepLab	testB	20.78	14.56	8.80	4.58	0.80	29.50
MAttNet	res101-mrcn	testB	**54.82**	**51.73**	**47.27**	**38.58**	**17.00**	**40.08**

RefCOCOg

Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
MAttNet	res101-mrcn	val	64.48	61.52	56.50	43.97	14.67	47.64
MAttNet	res101-mrcn	test	65.60	62.92	57.31	44.44	12.55	48.61

Table 3.14: Comparison of segmentation performance on RefCOCO, RefCOCO+, and our results on RefCOCOg.

able to attend to the portion that should be used by the "in-box" subject module and the portion that should be used by the "out-of-box" relationship module. Additionally our subject module also displays reasonable spatial "in-box" attention, which qualitatively explains why attentional pooling (Table. 3.12 Line 6) outperforms average pooling (Table. 3.12 Line 5). For comparison, some incorrect comprehension are shown in Fig. 3.14. Most errors are due to sparsity in the training data, ambiguous expressions, or detection error.

3.5.2.2 Segmentation from Referring Expression

Our model can also be used to address referential object segmentation Hu et al. (2016a); Liu et al. (2017a). Instead of using Faster R-CNN as the backbone net, we now turn to res101-based Mask R-CNN He et al. (2017) (res101-mrcn). We apply the same procedure described in Sec. 3.5.1 on the detected objects, and use the one with highest matching score as our prediction. Then we feed the predicted bounding box to the mask branch to obtain a pixel-wise segmentation. We evaluate the full model of MAttNet and compare with the best results reported in Liu et al.

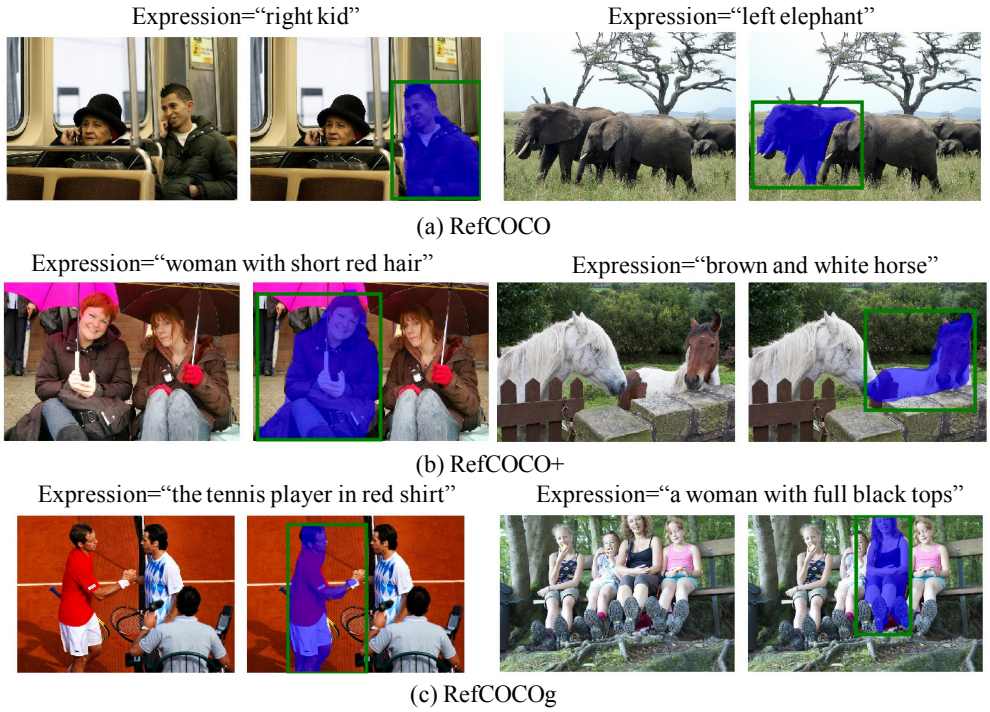


Figure 3.15: Examples of fully-automatic MAttNet referential segmentation.

(2017a). We use Precision@X (X $\in \{0.5, 0.6, 0.7, 0.8, 0.9\}$)[3] and overall Intersection-over-Union (IoU) as metrics. Results are shown in Table. 3.14 with our model outperforming state-of-the-art results by a large margin under all metrics[4]. As both Liu et al. (2017a) and MAttNet use res101 features, such big gains may be due to our proposed model. We believe decoupling box localization (comprehension) and segmentation brings a large gain over FCN-style Long et al. (2015) foreground/background mask classification Hu et al. (2016a); Liu et al. (2017a) for this instance-level segmentation problem, but a more end-to-end segmentation system may be studied in future work. Some referential segmentation examples are shown in Fig. 3.15.

[3] Precision@0.5 is the percentage of expressions where the IoU of the predicted segmentation and ground-truth is at least 0.5.

[4] There is no experiments on RefCOCOg's val/test splits in Liu et al. (2017a), so we show our performance only for reference in Table 3.14.

CHAPTER 4: ALBUM SUMMARIZATION AND STORYTELLING

In this chapter, we address the problem of end-to-end visual storytelling. Given a photo album, our model first selects the most representative (summary) photos, and then composes a natural language story for the album. For this task, we make use of the Visual Storytelling dataset and a model composed of three hierarchically-attentive Recurrent Neural Nets Yu et al. (2017a) to: encode the album photos, select representative (summary) photos, and compose the story. Automatic and human evaluations show our model achieves better performance on selection, generation, and retrieval than baselines.

4.1 Introduction

Since we first developed language, humans have always told stories. Fashioning a good story is an act of creativity and developing algorithms to replicate this has been a long running challenge. Adding pictures as input can provide information for guiding story construction by offering visual illustrations of the storyline. In the related task of image captioning, most methods try to generate descriptions only for individual images or for short videos depicting a single activity. Very recently, datasets have been introduced that extend this task to longer temporal sequences such as movies or photo albums Rohrbach et al. (2016b); Pan et al. (2016); Lu and Grauman (2013); Huang et al. (2016).

The type of data we consider in this work provides input illustrations for story generation in the form of photo albums, sampled over a few minutes to a few days of time. For this type of data, generating textual descriptions involves telling a temporally consistent story about the depicted visual information, where stories must be coherent and take into account the temporal

context of the images. Applications of this include constructing visual and textual summaries of albums, or even enabling search through personal photo collections to find photos of life events.

Previous visual storytelling works can be classified into two types, vision-based and language-based, where image or language stories are constructed respectively. Among the vision-based approaches, unsupervised learning is commonly applied: e.g., Sigurdsson et al. (2016) learns the latent temporal dynamics given a large amount of albums, and Kim and Xing (2014) formulate the photo selection as a sparse time-varying directed graph. However, these visual summaries tend to be difficult to evaluate and selected photos may not agree with human selections. For language-based approaches, a sequence of natural language sentences are generated to describe a set of photos. To drive this work Park and Kim (2015) collected a dataset mined from Blog Posts. However, this kind of data often contains contextual information or loosely related language. A more direct dataset was recently released Huang et al. (2016), where multi-sentence stories are collected describing photo albums via Amazon Mechanical Turk.

In this work, we make use of the Visual Storytelling Dataset Huang et al. (2016). While the authors provide a seq2seq baseline, they only deal with the task of generating stories given 5-representative (summary) photos hand-selected by people from an album. Instead, we focus on the more challenging and realistic problem of end-to-end generation of stories from entire albums. This requires us to either generate a story from all of the album's photos or to learn selection mechanisms to identify representative photos and then generate stories from those summary photos. We evaluate each type of approach.

Ultimately, we propose a model of hierarchically-attentive recurrent neural nets, consisting of three RNN stages. The first RNN encodes the whole album context and each photo's content, the second RNN provides weights for photo selection, and the third RNN takes the weighted representation and decodes to the resulting sentences. Note that during training, we are only given the full input albums and the output stories, and our model needs to learn the summary photo selections latently.

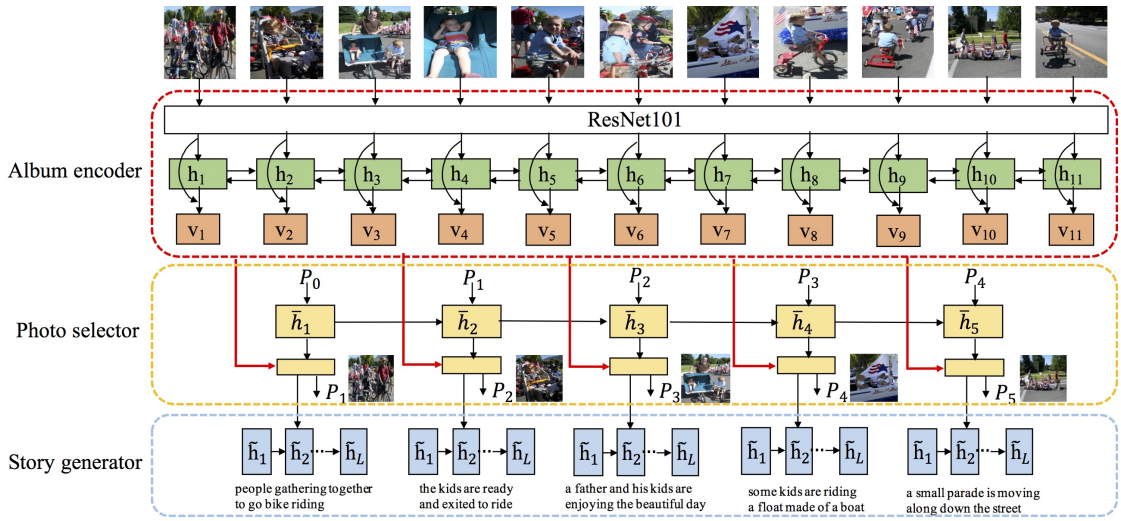


Figure 4.1: Model: the *album encoder* is a bi-directional GRU-RNN that encodes all album photos; the *photo selector* computes the probability of each photo being the tth album-summary photo; and finally, the *story generator* outputs a sequence of sentences that combine to tell a story for the album.

We show that our model achieves better performance over baselines under both automatic metrics and human evaluations. As a side product, we show that the latent photo selection also reasonably mimics human selections. Additionally, we propose an album retrieval task that can reliably pick the correct photo album given a sequence of sentences, and find that our model also outperforms the baselines on this task.

4.2 Related Work

Recent years have witnessed an explosion of interest in vision and language tasks, reviewed below.

Visual Captioning: Most recent approaches to image captioning Vinyals et al. (2015b); Xu et al. (2015) have used CNN-LSTM structures to generate descriptions. For captioning video or movie content Venugopalan et al. (2015); Pan et al. (2016), sequence-to-sequence models are widely applied, where the first sequence encodes video frames and the second sequence decodes the description. Attention techniques Xu et al. (2015); Yu et al. (2016a); Yao et al. (2015) are commonly incorporated for both tasks to localize salient temporal or spatial information.

Video Summarization: Similar to documentation summarization Rush et al. (2015); Cheng and Lapata (2016); Mei et al. (2016); Woodsend and Lapata (2010) which extracts key sentences and words, video summarization selects key frames or shots. While some approaches use unsupervised learning Lu and Grauman (2013); Khosla et al. (2013) or intuitive criteria to pick salient frames, recent models learn from human-created summaries Gygli et al. (2015); Zhang et al. (2016b,a); Gong et al. (2014a). Recently, to better exploit semantics, Choi et al. (2017) proposed textually customized summaries.

Visual Storytelling: Visual storytelling tries to tell a coherent visual or textual story about an image set. Previous works include storyline graph modeling Kim and Xing (2014), unsupervised mining Sigurdsson et al. (2016), blog-photo alignment Kim et al. (2015), and language re-telling Huang et al. (2016); Park and Kim (2015). While Park and Kim (2015) collects data by mining Blog Posts, Huang et al. (2016) collects stories using Mechanical Turk, providing more directly relevant stories.

4.3 Model

Our model (Fig. 4.1) is composed of three modules: Album Encoder, Photo Selector, and Story Generator, jointly learned during training.

4.3.1 Album Encoder

Given an album $A = \{a_1, a_2, ..., a_n\}$, composed of a set of photos, we use a bi-directional RNN to encode the local album context for each photo. We first extract the 2048-dimensional visual representation $f_i \in R^k$ for each photo using ResNet101 He et al. (2016), then a bi-directional RNN is applied to encode the full album. Following Huang et al. (2016), we choose a Gated Recurrent Unit (GRU) as the RNN unit to encode the photo sequence. The sequence output at each time step encodes the local album context for each photo (from both directions). Fused with the visual representation followed by ReLU, our final photo representation is (top module in Fig. 4.1):

$$
\begin{aligned}
f_i &= \text{ResNet}(a_i) \\
\vec{h}_i &= \vec{\text{GRU}}_{album}(f_i, \vec{h}_{i-1}) \\
\overleftarrow{h}_i &= \overleftarrow{\text{GRU}}_{album}(f_i, \overleftarrow{h}_{i+1}) \\
v_i &= \text{ReLU}([\vec{h}_i, \overleftarrow{h}_i] + f_i).
\end{aligned}
$$

4.3.2 Photo Selector

The Photo Selector (illustrated in the middle yellow part of Fig. 4.1) identifies representative photos to summarize an album's content. As discussed, we do not assume that we are given the ground-truth album summaries during training, instead regarding selection as a latent variable in the end-to-end learning. Inspired by Pointer Networks Vinyals et al. (2015a), we use another GRU-RNN to perform this task. Note while the pointer network requires grounding labels, we regard the labels as latent variables.

Given the album representation $V^{n\times k}$, the photo selector outputs probabilities $p_t \in R^n$ (likelihood of selection as t-th summary image) for all photos using soft attention.

$$
\begin{aligned}
\bar{h}_t &= \text{GRU}_{select}(p_{t-1}, \bar{h}_{t-1}), \\
p(y_{a_i}(t) = 1) &= \sigma(\text{MLP}([\bar{h}_t, v_i])),
\end{aligned}
$$

At each summarization step, t, the GRU takes the previous p_{t-1} and previous hidden state as input, and outputs the next hidden state $\bar{h}_t$. $\bar{h}_t$ is fused with each photo representation v_i to compute the i^{th} photo's attention $p_t^i = p(y_{a_i}(t) = 1)$. At test time, we simply pick the photo with the highest probability to be the summary photo at step t.

4.3.3 Story Generator

To generate an album's story, given the album representation matrix V and photo summary probabilities p_t from the first two modules, we compute the visual summary representation $g_t \in R^k$ (for the t-th summary step). This is a weighted sum of the album representations, i.e., $g_t = p_t^T V$. Each of these 5 g_t embeddings (for $t = 1$ to 5) is then used to decode 1 of the 5 story sentences respectively, as shown in the blue part of Fig. 4.1.

Given a story $S = \{s_t\}$, where s_t is t-th summary sentence. Following Donahue et al. (2015), the l-th word probability of the t-th sentence is:

$$\begin{aligned} w_{t,l-1} &= W_e s_{t,l-1}, \\ \tilde{h}_{t,l} &= \text{GRU}_{story}(w_{t,l-1}, g_t, \tilde{h}_{t,l-1}), \\ p(s_{t,l}) &= \text{softmax}(\text{MLP}(\tilde{h}_{t,l})), \end{aligned} \tag{4.1}$$

where W_e is the word embedding. The GRU takes the joint input of visual summarization g_t, the previous word embedding $w_{t,l}$, and the previous hidden state, then outputs the next hidden state. The generation loss is then the sum of the negative log likelihoods of the correct words: $L_{gen}(S) = -\sum_{t=1}^{T}\sum_{l=1}^{L_t} \log p_{t,l}(s_{t,l})$.

To further exploit the notion of temporal coherence in a story, we add an order-preserving constraint to order the sequence of sentences within a story (related to the story-sorting idea in Agrawal et al. (2016)). For each story S we randomly shuffle its 5 sentences to generate negative story instances S'. We then apply a max-margin ranking loss to encourage correctly-ordered stories: $L_{rank}(S, S') = \max(0, m - \log p(S') + \log p(S))$. The final loss is then a combination of the generation and ranking losses:

$$L = L_{gen}(S) + \lambda L_{rank}(S, S'). \tag{4.2}$$

beam size=3				
	Bleu3	Rouge	Meteor	CIDEr
enc-dec	19.58	29.23	33.02	4.65
enc-attn-dec	19.73	28.94	32.98	4.96
h-attn	20.53	29.82	33.81	6.84
h-attn-rank	**20.78**	**29.82**	**33.94**	**7.38**
h-(gd)attn-rank	21.02	29.53	34.12	7.51

Table 4.1: Story generation evaluation.

enc-dec (29.50%)	h-attn-rank (70.50%)
enc-attn-dec (30.75%)	h-attn-rank (69.25%)
h-attn-rank (30.50%)	gd-truth (69.50%)

Table 4.2: Human evaluation showing how often people prefer one model over the other.

4.4 Experiments

We use the Visual Storytelling Dataset Huang et al. (2016), consisting of 10,000 albums with 200,000 photos. Each album contains 10-50 photos taken within a 48-hour span with two annotations: 1) 2 album summarizations, each with 5 selected representative photos, and 2) 5 stories describing the selected photos.

4.4.1 Story Generation

This task is to generate a 5-sentence story describing an album. We compare our model with two sequence-to-sequence baselines: 1) an encoder-decoder model (enc-dec), where the sequence of album photos is encoded and the last hidden state is fed into the decoder for story generation, 2) an encoder-attention-decoder model Xu et al. (2015) (enc-attn-dec) with weights computed using a soft-attention mechanism. At each decoding time step, a weighted sum of hidden states from the encoder is decoded. For fair comparison, we use the same album representation (Sec. 4.3.1) for the baselines.

We test two variants of our model trained with and without ranking regularization by controlling λ in our loss function, denoted as h-attn (without ranking), and h-attn-rank (with ranking). Evaluations of each model are shown in Table 4.1. The h-attn outperforms both baselines, and h-attn-rank achieves the best performance for all metrics. Note, we use beam-search with beam

	precision	recall
DPP	43.75%	27.41%
enc-attn-dec	38.53%	24.25%
h-attn	42.85%	27.10%
h-attn-rank	**45.51%**	**28.77%**

Table 4.3: Album summarization evaluation.

	R@1	R@5	R@10	MedR
enc-dec	10.70%	29.30%	41.40%	14.5
enc-attn-dec	11.60%	33.00%	45.50%	11.0
h-attn	18.30%	**44.50%**	**57.60%**	**6.0**
h-attn-rank	**18.40%**	43.30%	55.50%	7.0

Table 4.4: 1000 album retrieval evaluation.

size=3 during generation for a reasonable performance-speed trade-off (we observe similar improvement trends with beam size = 1).[1] To test performance under optimal image selection, we use one of the two ground-truth human-selected 5-photo-sets as an oracle to hard-code the photo selection, denoted as h-(gd)attn-rank. This achieves only a slightly higher Meteor compared to our end-to-end model.

Additionally, we also run human evaluations in a forced-choice task where people choose between stories generated by different methods. For this evaluation, we select 400 albums, each evaluated by 3 Turkers. Results are shown in Table 4.2. Experiments find significant preference for our model over both baselines. As a simple Turing test, we also compare our results with human written stories (last row of Table 4.2), indicating room for improvement of methods.

4.4.2 Album Summarization

We evaluate the precision and recall of our generated summaries (output by the photo selector) compared to human selections (the combined set of both human-selected 5-photo stories). For comparison, we evaluate enc-attn-dec on the same task by aggregating predicted attention and selecting the 5 photos with highest accumulated attention. Additionally, we also run DPP-

[1] We also compute the p-value of Meteor on 100K samples via the bootstrap test Efron and Tibshirani (1994), as Meteor has better agreement with human judgments than Bleu/Rouge Huang et al. (2016). Our h-attn-rank model has strong statistical significance ($p = 0.01$) over the enc-dec and enc-attn-dec models (and is similar to the h-attn model).

based video summarization Kulesza et al. (2012) using the same album features. Our models have higher performance compared to baselines as shown in Table 4.3 (though DPP also achieves strong results, indicating that there is still room to improve the pointer network).

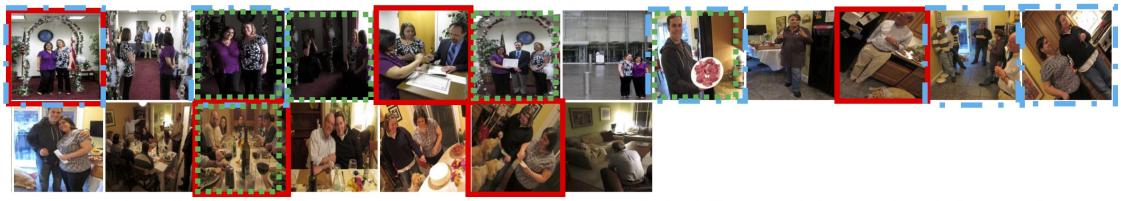
4.4.3 Output Example Analysis

Fig. 4.2 and Fig. 4.3 shows several output examples of the joint album summarization and storytelling generation. We compare our full model h-attn-rank with the baseline enc-attn-dec, as both models are able to do the album summarization and story generation tasks jointly. In Fig. 4.2 and Fig. 4.3, we use blue dashed box and red box to indicate the album summarization by the two models. As reference, we also show the ground-truth album summaries by randomly selecting 1 out of 2 human album summaries, which are highlighted with green box. Below each album are their generated stories.

4.4.4 Album Retrieval

Given a human-written story, we introduce a task to retrieve the album described by that story. We randomly select 1000 albums and one ground-truth story from each for evaluation. Using the generation loss, we compute the likelihood of each album A_m given the query story S and retrieve the album with the highest generation likelihood, $A = \text{argmax}_{A_m} p(S|A_m)$. We use Recall@k and Median Rank for evaluation. As shown in Table 4.4), we find that our models outperform the baselines, but the ranking term in Eqn.4.2 does not improve performance significantly.

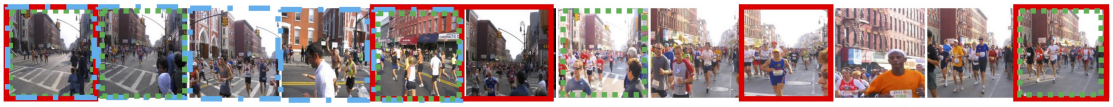
enc-attn-dec: the students were ready for the meeting . the community . they were all ready for the meeting . they were all ready . they had to make sure to make sure had to be done .
hattn-rank: i went to the organization. they were some speakers . they were a lot of the lecture . the students were very happy to see the speaker. the last speaker was the best part of the day .
gd-truth: i walked into the building to give my speech . there were many topics that need to be covered . we stood there and reviewed them . members got to ask questions . we sat and came to an agreement .



enc-attn-dec: the family gathered for dinner for dinner . they had a lot of food . the food and friends and the best together . the best part of the night was a lot of course had a lot of fun .
hattn-rank: the family gathered their friends . they had a great party . They had a lot of people showed up to celebrate the occasion . everyone was so happy to be together . after dinner was over , they had a good drink it was a success .
gd-truth: this couple are going to get married . they took pictures of this event . they got their license . their friend cooked them steaks . they all had a big dinner afterwards .

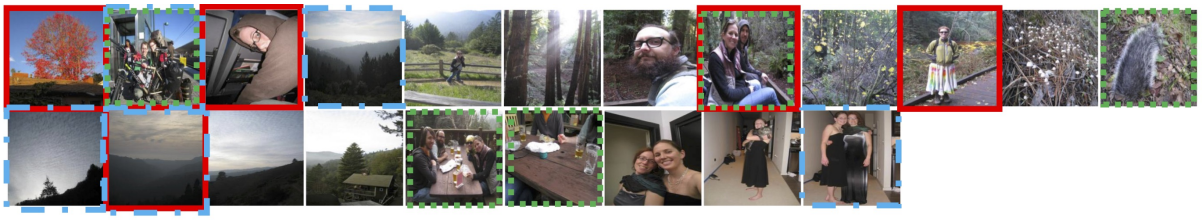
enc-attn-dec: i went to the fair . the organization was there . there were many people there . the cake was very delicious cake we had a lot of fun .
hattn-rank: the kids were excited for my birthday . the kids were decorated with balloons . the kids were playing with each other. the cake was decorated . We all had a lot of the gifts .
gd-truth: the group had a celebration among themselves . the room was decorated with balloons and ribbons . the cake was enjoyed by all . and there was plenty of food . a few antics were performed to entertain the crowd

enc-attn-dec: today was the soldiers are presenting to the ceremony . they are presenting the award for the men . they all of them . the soldiers were presented . the speech was given to the award .
hattn-rank: today was a great day for the military . the soldiers were very proud of the ceremony . the soldiers were very happy to receive the award . they were very happy to be there . the men shook hands in the event .
gd-truth: i went to the award ceremony yesterday . there were a lot of people there . everyone received an award for their effort . they had a great time . i really enjoyed being there . some of the soldiers started singing .

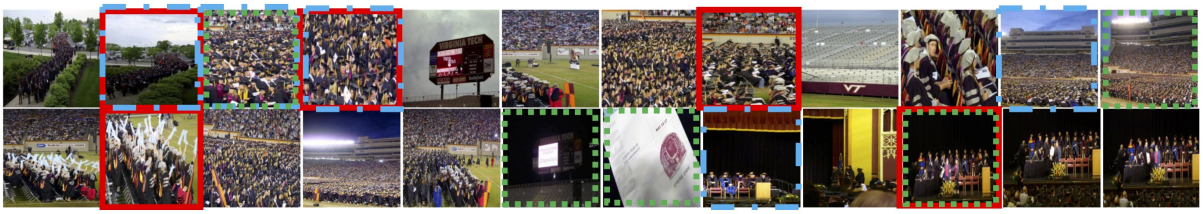


enc-attn-dec: the runners were ready for the finish line . the runners were ready to start . the runners were ready to the finish line . the runners were ready for the race . the runners .
hattn-rank: the runners were getting ready for the race . the runners were all lined up . the runners were close to the runners . the runners were very tired . the winner the finish line .
gd-truth: the front runners raced through the city . another group followed behind . some waved at supporters . there were many participants in the race . some had to walk to the finish line .

Figure 4.2: Examples of album summarization and storytelling by enc-attn-dec (blue), h-attn-rank (red), and ground-truth (green). We randomly select 1 out of 2 human album summaries as ground-truth here.



enc-attn-dec: [male] and family went to the family went to take a beautiful views . the view from the top . the view from the view from the top of the beautiful sight . we had a great view of the trip was a great way to end the day .
hattn-rank: i went on vacation last weekend . there were many people there . we had fun there . we took a lot of pictures . we had a great time at the best part of the day .
gd-truth: i rode my bike into the nature trail yesterday . i brought some friends with me . we saw some animals while we were there . we took a break and had a few drinks . we had a lot of fun outside .



enc-attn-dec: it was a big day for the graduation . the graduates . the graduates were waiting for the ceremony . the graduates were waiting for their seats . the graduates were excited . the crowd .
hattn-rank: the graduation ceremony was packed . the graduates were excited to see each other . there were many speakers there . the graduates were very proud of the ceremony . after the ceremony in the crowd is the best .
gd-truth: there was a huge audience at the graduation ceremony last week . everyone in town was there . all of the speakers took several minutes to deliver their speeches . we used the sports stadium for the event . everyone was really happy to receive their diploma at the end .

enc-attn-dec: the halloween at the with a lot . there . they all the kids . the party was a little [male] and [male] had a little too much .
hattn-rank: i went to the halloween party . there were many people dressed up costumes . there was a lot of fun . there was so much scary costumes. the costumes were all kinds of fun !
gd-truth: he stood by the fish tank waiting for things to get started . still waiting for the first guests , he decides to sit down . people start arriving at the party in crazy costumes . they started discussing the scenery in the house . everyone poses at the end of the party to take a picture .

enc-attn-dec: the family was excited to graduate . they had a great time for the graduation . the family was so many years ago . the family was happy to be together . the whole family .
hattn-rank: the graduation ceremony was very excited to graduate . she was so happy to be there . she was so proud of her . they are so happy couple . we had a lot of cake for the best birthday .
gd-truth: a picture of the happy graduate . the brother of the graduate , he is still in school . a friend of the graduate with a picture to remember the day . friends of the graduate in the picture to remember the day . a picture of the graduation sheet cake

Figure 4.3: More examples of album summarization and storytelling by enc-attn-dec (blue), h-attn-rank (red), and ground-truth (green). We randomly select 1 out of 2 human album summaries as ground-truth here.

CHAPTER 5: MULTI-TARGET EMBODIED QUESTION ANSWERING

Embodied Question Answering (EQA) is a relatively new task where an agent is asked to answer questions about its environment from egocentric perception. EQA as introduced in Das et al. (2018a) makes the fundamental assumption that every question, *e.g*. "what color is the car?", has exactly **one** target ("car") being inquired about. This assumption puts a direct limitation on the abilities of the agent.

We present a generalization of EQA – Multi-Target EQA (MT-EQA) Yu et al. (2019). Specifically, we study questions that have **multiple** targets in them, such as "Is the dresser in the bedroom bigger than the oven in the kitchen?", where the agent has to navigate to multiple locations ("dresser in bedroom", "oven in kitchen") and perform comparative reasoning ("dresser" bigger than "oven") before it can answer a question. Such questions require the development of entirely new modules or components in the agent. To address this, we propose a modular architecture composed of a program generator, a controller, a navigator, and a VQA module. The program generator converts the given question into sequential executable sub-programs; the navigator guides the agent to multiple locations pertinent to the navigation-related sub-programs; and the controller learns to select relevant observations along its path. These observations are then fed to the VQA module to predict the answer. We perform detailed analysis for each of the model components and show that our joint model can outperform previous methods and strong baselines by a significant margin.

5.1 Introduction

One of the grand challenges of AI is to build intelligent agents that visually perceive their surroundings, communicate with humans via natural language, and act in their environments to

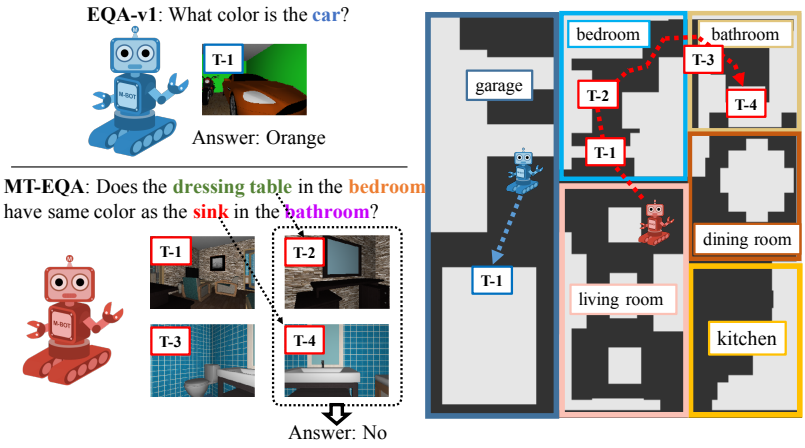


Figure 5.1: Difference between EQA-v1 and MT-EQA. While EQA-v1's question asks about a single target "car", MT-EQA's question involves multiple targets (e.g., bedroom, dressing table, bathroom, sink) to be navigated, and attribute comparison between multiple targets (e.g., dressing table and sink).

accomplish tasks. In the vision, language, and AI communities, we are witnessing a shift in focus from *internet vision* to *embodied AI* – with the creation of new tasks and benchmarks Chaplot et al. (2018); Anderson et al. (2018b); Gupta et al. (2017); Zhu et al. (2017a), instantiated on new simulation platforms Kolve et al. (2017); Savva et al. (2017); Wu et al. (2018); Xia et al. (2018); Kempka et al. (2016); Brodeur et al. (2017).

The focus of this work is one such embodied AI task, Embodied Question Answering (EQA) Das et al. (2018a), which tests an agent's overall ability to jointly perceive its surrounding, communicate with humans, and act in a physical environment. Specifically, in EQA, an agent is spawned in a random location within an environment and is asked a question about something in that environment, for example *"What color is the lamp?"*. In order to answer the question correctly, the agent needs to parse and understand the question, navigate to a good location (looking at the "lamp") based on its first-person perception of the environment and predict the right answer (*e.g.* "blue").

However, there is still much left to be done in EQA. In its original version, the EQA-v1 dataset only consists of single-target question-answer pairs, such as *"What color is the car?"*. The agent just needs to find the car then check its color based on its last observed frames. How-

ever, the single target constraint places a direct limitation on the possible set of tasks that the AI agent can tackle. For example, consider the question *"Is the kitchen larger than the bedroom?"* in EQA-v1; the agent would not be able to answer this question because it involves navigating to multiple targets –"kitchen" *and* "bedroom" – and the answer requires comparative reasoning between the two rooms, where all of these skills are not part of the original EQA task.

In this work, we present a generalization of EQA – multi-target EQA (MT-EQA). Specifically, we study questions that have multiple implicit targets in them, such as *"Is the dresser in the bedroom bigger than the oven in the kitchen?"*. At a high-level, our work is inspired by the visual reasoning work of Neural Modular Networks Andreas et al. (2016b) and CLEVR Johnson et al. (2017a). These works study compositional and modular reasoning in a fully-observable environment (an image). Our work may be viewed as embodied visual reasoning, where an agent is asked a question involving multiple modules and needs to gather information before it can execute them. In MT-EQA, we propose 6 types of compositional questions which compare attribute properties (color, size, distance) between multiple targets (objects/rooms). Fig. 5.1 shows an example from the MT-EQA dataset and contrasts it to the original EQA-v1 dataset.

The assumption in EQA-v1 of decoupling navigation from question-answering not only makes the task simpler but is also reflected in the model used – the EQA-v1 model simply consists of an LSTM navigator which after stopping, hands over frames to a VQA module. In contrast, MT-EQA introduces new modeling challenges that we address in this work. Consider the MT-EQA question in Fig. 5.1 – *"Does the table in the bedroom have same color as the sink in the bathroom?"*. From this example, it is clear that not only is it necessary to have a tighter integration between navigator and VQA, but we also need to develop fundamentally new modules. An EQA-v1 Das et al. (2018a) agent would navigate to the final target location and run the VQA module based on its last sequence of frames along the path. In this case, only the "sink" would be observed from the final frames but dressing table would be lost. Instead, we propose a new model that consists of 4 components: (a) a program generator, (b) a navigator, (c) a controller and (d) a VQA module. The program generator converts the given question into sequential

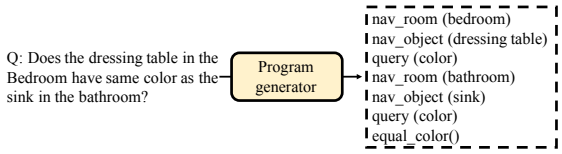


Figure 5.2: Program Generator.

executable sub-programs, as shown in Fig. 5.2. The controller executes these sub-programs sequentially and gives control to the navigator when the navigation sub-programs are invoked (*e.g*. `nav_room(bedroom)`). During navigation, the controller processes the first-person views observed by the agent and predicts whether the target of the sub-program (*e.g*. bedroom) has been reached. In addition, the controller extracts cues pertinent to the questioned property of the sub-target, *e.g*. `query(color)`. Finally, these cues are fed into the VQA module which deals with the comparison of different attributes, *e.g*. executing `equal_color()` by comparing the color of dressing table and sink in Fig. 5.1.

Empirically, we show results for our joint model and analyze the performance of each of our components. Our full model outperforms the baselines under almost every navigation and QA metric by a large margin. We also report performance for the navigator, the controller, and the VQA module, when executed separately in an effort to isolate and better understand the effectiveness of these components. Our ablation studies show that our full model is better at all sub-tasks, including room navigation, object navigation and final EQA accuracy. Additionally, we find quantitative evidence that MT-EQA questions on closer targets are relatively easier to solve as they require shorter navigation, while questions for farther targets are harder.

5.2 Related Work

Our work relates to research in embodied perception and modular predictive models for program execution.

Embodied Perception. Visual recognition from images has witnessed tremendous success in recent years with the advent of deep convolutional neural networks (CNNs) Krizhevsky et al. (2012); Szegedy et al. (2015); He et al. (2016) and large-scale datasets, such as ImageNet Rus-

sakovsky et al. (2015) and COCO Lin et al. (2014). More recently, we are beginning to witness a resurgence of *active vision*. For example, end-to-end learning methods successfully predict robotic actions from raw pixel data Levine et al. (2016). Gupta *et al*. Gupta et al. (2017) learn to navigate via mapping and planning. Sadeghi & Levine Sadeghi and Levine (2017) teach an agent to fly in simulation and show its performance in the real world. Gandhi *et al*. Dhiraj Gandhi (2017) train self-supervised agents to fly from examples of drones crashing.

At the intersection of active perception and language understanding, several tasks have been proposed, including instruction-based navigation Chaplot et al. (2018); Anderson et al. (2018b), target-driven navigation Zhu et al. (2017b); Gupta et al. (2017), embodied question answering Das et al. (2018a), interactive question answering Gordon et al. (2018), and task planning Zhu et al. (2017a). While these tasks are driven by different goals, they all require training agents that can perceive their surroundings, understand the goal – either presented visually or in language instructions – and act in a virtual environment. Furthermore, the agents need to show strong generalization ability when deployed in novel unseen environments Gupta et al. (2017); Wu et al. (2018).

Environments. There is an overbearing cost to developing real-world interactive benchmarks. Undoubtedly, this cost has hindered progress in studying embodied tasks. On the contrary, virtual environments that offer rich, efficient simulations of real-world dynamics, have emerged as promising alternatives to potentially overcome many of the challenges faced in real-world settings.

Recently there has been an explosion of simulated 3D environments in the AI community, all tailored towards different skill sets. Examples include ViZDoom Kempka et al. (2016), TorchCraft Synnaeve et al. (2016) and DeepMind Lab Beattie et al. (2016). Just in the last year, simulated environments of semantically complex, realistic 3D scenes have been introduced, such as HoME Brodeur et al. (2017), House3D Wu et al. (2018), MINOS Savva et al. (2017), Gibson Xia et al. (2018) and AI2THOR Kolve et al. (2017). In this work, we use House3D, following the original EQA

	Question Type	Template
EQA-v1	location	*"What room is the <OBJ> located in?"*
	color	*"What color is the <OBJ>?"*
	color_room	*"What color is the <OBJ> in the <ROOM>?"*
	preposition	*"What is <on/above/below/next-to> the <OBJ> in the <ROOM>?"*
MT-EQA	object_color_compare_inroom	*"Does <OBJ1> share same color as <OBJ2> in <ROOM>?"*
	object_color_compare_xroom	*"Does <OBJ1> in <ROOM1> share same color as <OBJ2> in <ROOM2>?"*
	object_size_compare_inroom	*"Is <OBJ1> bigger/smaller than <OBJ2> in <ROOM>?"*
	object_size_compare_xroom	*"Is <OBJ1> in <ROOM1> bigger/smaller than <OBJ2> in <ROOM2>?"*
	object_dist_compare	*"Is <OBJ1> closer than/farther from <OBJ2> than <OBJ3> in <ROOM>?"*
	room_size_compare	*"Is <ROOM1> bigger/smaller than <ROOM2> in the house?"*

Table 5.1: Question types and the associated templates used in EQA-v1 and MT-EQA.

task Das et al. (2018a). House3D is a rich, interactive 3D environment based on human-designed indoor scenes sourced from SUNCG Song et al. (2017).

Modular Models. Neural module networks were originally introduced for visual question answering Andreas et al. (2016b). These networks decompose a question into several components and dynamically assemble a network to compute the answer, dealing with variable compositional linguistic structures. Since their introduction, modular networks have been applied to several other tasks: visual reasoning Hu et al. (2017a); Johnson et al. (2017b), relationship modeling Hu et al. (2017b), embodied question answering Das et al. (2018b), multitask reinforcement learning Andreas et al. (2017), language grounding on images Yu et al. (2018) and video understanding Liu et al. (2018). Inspired by Das et al. (2018c); Johnson et al. (2017b), we cast EQA as a partially observable version of CLEVR and extend the modular idea to this task, which we believe requires an increasingly modular model design to address visual reasoning within a 3D environment.

5.3 Multi-Target EQA Dataset

We now describe our proposed Multi-Target Embodied Question Answering (MT-EQA) task and associated dataset, contrasting it against EQA-v1. In v1 Das et al. (2018a), the authors select 750 (out of about 45,000) environments for the EQA task. Four types of questions are proposed, each questioning a property (color, location, preposition) of a single target (room,

Question Type	Functional Form
object_color_compare	select(rooms) → unique(rooms) → select(objects) → unique(objects) → pair(objects) → query(color_compare)
object_size_compare	select(rooms) → unique(rooms) → select(objects) → unique(objects) → pair(objects) →query(size_compare)
object_dist_compare	select(rooms) → unique(rooms) → select(objects) → unique(objects) → triplet(objects) →query(dist_compare)
room_size_compare	select(rooms) → unique(rooms) → pair(rooms) → query(size_compare)

Table 5.2: Functional forms of all question types in the MT-EQA dataset. Note that for each object color/size comparison question type, there exists two modes: inroom and xroom, depending on whether the two objects are in the same room or not. For example, object_color_compare_xroom compares the color of two objects in two different rooms.

object), as shown at the top of Table. 5.1. Our proposed MT-EQA task generalizes EQA-v1 and involves comparisons of various attributes (color, size, distance) between multiple targets, shown at the bottom of Table. 5.1. Next, we describe in detail the generation process, as well as useful statistics of MT-EQA.

5.3.1 Multi-Target EQA Generation

We generate question-answer pairs using the annotations available on SUNCG. We use the same number of rooms and objects as EQA-v1 (see Figure 2 in Das et al. (2018a)). Each question in MT-EQA is represented as a series of functional programs, which can be executed on the environment to yield a ground-truth answer. The functional programs consist of some elementary operations, e.g., select(), unique(), object_color_pair(), query(), *etc*., that operate on the room and object annotations.

Each question type is associated with a question template and a sequence of operations. For example, consider the question type in MT-EQA object_color_compare, whose template is *"Does <OBJ1> share same color as <OBJ2> in <ROOM>?"*. Its sequence of elementary operations is: select(rooms) → unique(rooms) → select(objects) → unique(objects) → pair(objects) → query(color_compare).

The first function, select(rooms), returns all rooms in the environment. The second function, unique(rooms), selects a single unique room from the list to avoid ambiguity. Similarly, the third function, select(objects), and fourth function, unique(objects), return unique objects

	random	q-LSTM	q-NN	q-BoW	"no"
Test Acc. (%)	49.44	48.24	53.74	49.22	53.28

Table 5.3: EQA (test) accuracy using questions and priors.

in the selected room. The fifth function, pair(objects), pairs the objects. The final function, query(color_compare), compares their colors.

We design 6 types of questions comparing different attributes between objects (inside same room/across different rooms), distance comparison, and room size comparison. All question types and templates are shown in Table 5.2.

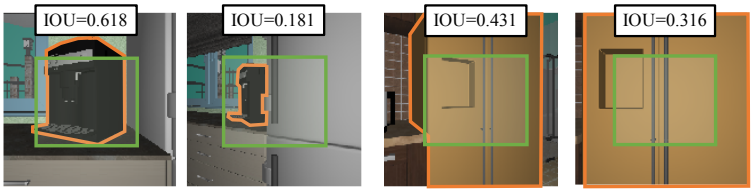
In some cases, a question instantiation returned from the corresponding program, as shown above, might not be executable, as rooms might be disconnected or not reachable. To check if a question is feasible, we execute the corresponding `nav_room()` and `nav_object()` programs and compute shortest paths connecting the targets in the question. If there is no path[1], it means the agent would not be able to look at all targets starting from its given spawn location. We filter out such impossible questions.

For computing the shortest path connecting the targets, we need to find the position (x, y, z, yaw) that best views each target. In order to do so, we first sample 100 positions near the target. For each position, we pick the yaw angle that looks at the target with the highest Intersection-Over-Union (IOU), computed using the target's mask[2] and a centered rectangular mask. Fig. 5.3 shows 4 IOU scores of *coffee machine* and *refrigerator* from different positions. We sort the 100 positions and pick the one with highest IOU as the best-view position of the target, which is used to connect the shortest-path. For each object, its highest IOU value IOU_{best} is recorded for evaluation purposes (as a reference of the target's best-view).

To minimize the bias in MT-EQA, we perform entropy-filtering, similar to Das et al. (2018a). Specifically for each unique question, we compute its answer distribution across the whole

[1] This is a result of noisy annotations in SUNCG and inaccurate occupancy maps due to the axis-aligned assumption returned by House3D.

[2] House3D returns the the ground-truth semantic segmentation for each first-person view.



(a) coffee machine (b) refrigerator

Figure 5.3: IOU between the target's mask and the centered rectangle mask. Higher IOU is achieved when the target has larger portion in the center of the view.

	Houses	Unique questions	Total questions
train	486	2,030	14,495
val	50	938	1,954
test	52	1,246	2,838

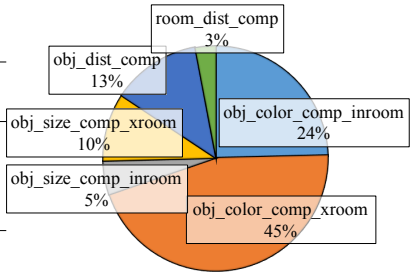


Figure 5.4: Overview of MT-EQA dataset including split statistics and question type distribution.

dataset. We exclude questions whose normalized answer distribution entropy is below 0.9[3]. This prevents the agent from memorizing easy question-answer pairs without looking at the environment. For example, the answer to *"is the bed in the living room bigger than the cup in the kitchen?"* is always *Yes*. Such questions are excluded from our dataset. After the two filtering stages, the MT-EQA questions are both balanced and feasible.

In addition, we check if MT-EQA is easily addressed by question-only or prior-only baselines. For this, we evaluate four question-based models: (a) an LSTM-based question-to-answer model, (b) a nearest neighbor (NN) baseline that finds the NN question from the training set and uses its most frequent answer as the prediction, (c) a bag-of-words (BoW) model that encodes a question followed by a learned linear classifier to predict the answer and (d) a naive "no" only answer model, since "no" is the most frequent answer by a slight margin. Table. 5.3 shows the results. There exists very little bias on the "yes/no" distribution (53.28%), and all question-based models make close to random predictions. In comparison, and as we empirically show in Sec. 5.5, our

[3] Rather than 0.5 in Das et al. (2018a), we set the normalized entropy threshold as 0.9 (maximum is 1) since all of our questions have binary answers.

1) `nav_object(phrase)`	2) `nav_room(phrase)`
3) `query(color / size / room_size)`	
4) `equal_color()`	
5) `object_size_compare(bigger / smaller)`	
6) `object_dist_compare(farther / closer)`	
7) `room_size_compare(bigger / smaller)`	

Table 5.4: MT-EQA executable programs.

results are far better than these baselines, indicating the necessity to explore the environment in order to answer the question. Besides, the results also address the concern in Anand et al. (2018) where language-only models (BoW and NN) already form competitive baselines for EQA-v1. In MT-EQA, these baselines perform close to chance as a result of the balanced binary question-answer pairs in MT-EQA.

Overall, our MT-EQA dataset consists of 19,287 questions across 588 environments[4], referring to a total of 61 unique object types in 8 unique room types. Fig. 5.4 shows the question type distribution. Approximately 32 questions are asked for each house on average, 209 at most and 1 at fewest. There are relatively fewer object_size_compare and room_size_compare questions as many frequently occurring comparisons are too easy to guess without exploring the environment and thus fail the entropy filtering. We will release the MT-EQA dataset and the generation pipeline.

5.4 Model

Our model is composed of 4 modules: the question-to-program generator, the navigator, the controller, and the VQA module. We describe these modules in detail.

5.4.1 Program Generator

The program generator takes the question as input and generates sequential programs for execution. We define 7 types of executable programs for the MT-EQA task in Table. 5.4. For

[4] The 588 environments are subset of EQA-v1's. Some environments are discarded due to entropy filtering and unavailable paths.

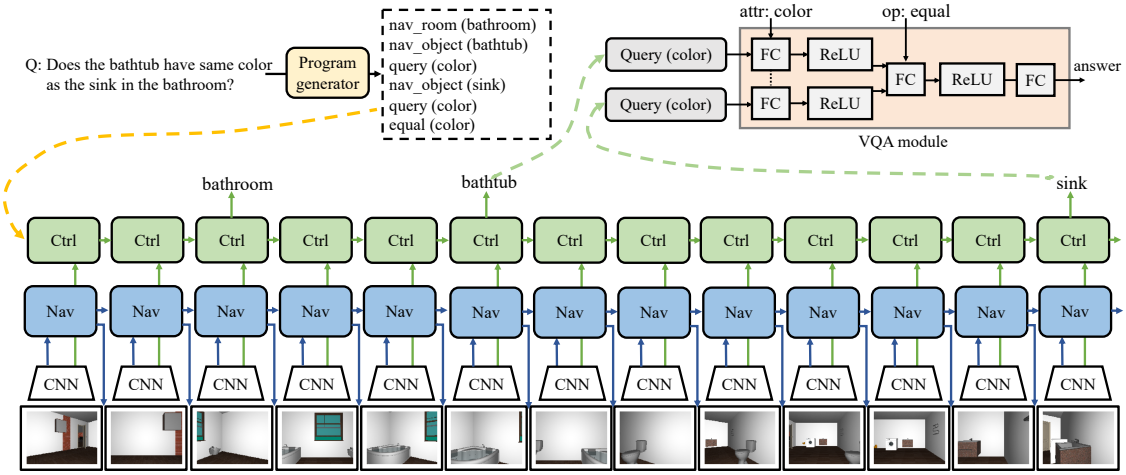


Figure 5.5: Model architecture: our model is composed of a program generator, a navigator, a controller, and a VQA module.

example, *"Is the bathtub the same color as the sink in the bathroom?"* is decomposed into a series of sequential sub-programs: `nav_room(bathroom)` → `nav_object(bathtub)` → `query_color()` → `nav_object(sink)` → `query_color()` → `equal_color()`. Similar to CLEVR Johnson et al. (2017a), the question programs are automatically generated in a templated manner (Table. 5.2), making sub-component decomposition (converting questions back to programs) simple (Table. 5.4). We use template-based rules by selecting and filling in the arguments in Table. 5.4 to generate the programs (which is always accurate). While a neural model could also be applied, a learned program generator is not the focus of our work.

5.4.2 Navigator

The navigator executes the `nav_room()` and `nav_object()` programs. As shown in Fig. 5.6(a), we use an LSTM as our core component. At each time step, the LSTM takes as inputs the current egocentric (first-person view) image, an encoding of the target phrase (*e.g.* "bathtub" if the program is `nav_object(bathtub)`), and the previous action, in order to predict the next action.

The navigator uses a CNN feature extractor that takes a 224x224 RGB image returned from the House3D renderer, and transforms it into a visual feature, which is then fed into the LSTM.

(a) Navigator (b) Controller

Figure 5.6: Navigator and Controller.

Similar to Das et al. (2018a), the CNN is pre-trained under a multi-task framework consisting of three tasks: RGB-value reconstruction, semantic segmentation, and depth estimation. Thus, the extracted feature contains rich information about the scene's appearance, content, and geometry (objects, color, texture, shape, and depth). In addition to the visual feature, the LSTM is presented with two additional inputs. The first is the target embedding, where we use the average embedding of GloVE vectors Pennington et al. (2014) over words describing the target. The second is previous action, which is in the form of a look-up from an action embedding matrix.

We want to note the different perceptual skills required for room and object navigation: Room navigation relies on understanding the overall scene and finding cross-room paths (entry/exit), while object navigation requires localizing the target object within a room and finding a path to reach it. To capture the difference, we implement two separate navigation modules, `nav_room()` and `nav_object()` respectively. These two modules share same architecture but are trained separately for different targets.

In MT-EQA, the action space for navigation consists of 3 action types: turning left (30 degrees), turning right (30 degrees), and moving forward. This is almost the same as EQA-v1 Das et al. (2018a), except we use larger turning angles – as our navigation paths are much longer due to the multi-target setting. We find that this change reduces the number of actions required for navigation, leading to easier training.

5.4.3 Controller

The controller is the central module in our model, as it connects all of the other modules by: 1) creating a plan from the program generator, 2) collecting the necessary observations from the navigator, and 3) invoking the VQA module.

Fig. 5.6 (b) shows the controller, whose key component is another LSTM. Consider the question *"Does the bathtub have same color as the sink in the bathroom?"* with part of its program as example – `nav_room(bathroom)` $\rightarrow$ `nav_object(bathtub)`. The controller starts by calling the room navigator to look for "bathroom". During navigation, the controller keeps track of the first-person views, looking for the target. Particularly, it extracts the features via CNN which are then fused with the target embedding as input to the LSTM. The controller predicts SELECT if the target is found, stopping the current navigator, in our example `nav_room(bathroom)`, and starting execution of the next program, `nav_object(bathtub)`.

Finally, after the object target "bathtub" has been found, the next program – `query_color()`, is executed. The controller extracts attribute features from the first-person view containing the target. In all, there are three attribute types in MT-EQA - object's color, object's size, and room's size. Again, we treat object and room differently in our model. For object-specific attributes, we use the hidden state of the controller at the location where SELECT was predicted. This state should contain semantic information for the target, as it is where the controller is confident the target is located. For room-specific attributes, the controller collects a panorama by asking the navigator to rotate 360 degrees (by performing 12 turning-right actions) at the location where SELECT is predicted. The CNN features from this panorama view are concatenated as the representation.

During program execution by the controller, the extracted cues for all the targets are stored, and in the end they are used by the VQA module to predict the final answer.

5.4.4 VQA Module

The final task requires reasoning, *e.g.*, `object_size_compare(bigger)`, `equal_color()`, *etc*. When the controller has gathered all of the targets for comparison, it invokes the VQA module. As shown in top-right of Fig. 5.5, the VQA module embeds the stored features of multiple targets into the question-attribute space, using a FC layer followed by ReLU. The transformed features are then concatenated and fed into another FC+ReLU which is conditioned on the comparison operator (equal, bigger than, smaller than, *etc*.). The output is a binary prediction (yes/no) for that attribute comparison. We call it compositional VQA (cVQA). The cVQA module in Fig. 5.5 depicts a two-input comparison as an example, but our cVQA module also extends to three inputs, for questions like *"Is the refrigerator closer to the coffee machine than the microwave?"*.

5.4.5 Training

Training follows a two-stage approach: First, the full model is trained using Imitation Learning (IL); Second, the navigator is further fine-tuned with Reinforcement Learning (RL) using policy gradients.

First, we jointly train our full model using imitation learning. For imitation learning, we treat the shortest paths and the key positions containing the targets as our ground-truth labels for navigation and for the controller's SELECT classifier, respectively. The objective function consists of a navigation objective and a controller objective at every time step t, and a VQA objective at the final step. For the i-th question, let $P^{nav}_{i,t,a}$ be action a's probability at time t, $P^{sel}_{i,t}$ be the controller's SELECT probability at time t, and P^{vqa}_{i} be the answer probability from VQA,

then we minimize the combined loss:

$$
\begin{aligned}
L &= L_{nav} + \alpha L_{ctrl} + \beta L_{vqa} \\
&= \underbrace{-\sum_i \sum_t \sum_a y^n_{i,t,a} \log P^{nav}_{i,t,a}}_{\text{Cross-entropy on navigator action}} \\
&\quad \underbrace{-\alpha \sum_i \sum_t (y^c_{i,t} \log P^{sel}_{i,t} + (1 - y^c_{i,t}) \log(1 - P^{sel}_{i,t}))}_{\text{Binary cross-entropy on controller's SELECT}} \\
&\quad \underbrace{-\beta \sum_i (y^v_i \log P^{vqa}_i + (1 - y^v_i) \log(1 - P^{vqa}_i))}_{\text{Binary cross-entropy on VQA's answer}}.
\end{aligned}
$$

Subsequently, we use RL to fine-tune the room and object navigators.

We provide two types of reward signals to the navigators. The first is a dense reward, corresponding to the agent's progress toward the goal (positive if moving closer to the target and negative if moving away). This reward is measured by the distance change in the 2D bird-view distance space, clipped to lie within $[-1.0, 1.0]$. The second is a sparse reward that quantifies whether the agent is looking at the target object when the episode is terminated. For object targets, we compute IOU_T between the target's mask and the centered rectangle mask at termination. We use the best IOU score of the target IOU_{best} as reference and compute the ratio $\frac{\text{IOU}_T}{\text{IOU}_{best}}$. If the ratio is greater than 0.5, we set the reward to 1.0 otherwise -1.0. For room targets, we assign reward 0.2 to the agent if it is inside the target room at termination, otherwise -0.2.

5.5 Experiments

In this section we describe our experimental results. Since MT-EQA is a complex task and our model is modular, we will show both the final results (QA accuracy) and the intermediate performance (for navigation). Specifically, we first describe our evaluation setup and metrics for MT-EQA. Then, we report the comparison of our model against several strong baselines. And finally, we analyze variants of our model and provide ablation results.

5.5.1 Evaluation Setup and Metrics

Spawn Location. MT-EQA questions involve multiple targets (rooms/objects) to be found. To prevent the agent from learning biases due to spawn location, we randomly select one of the mentioned targets as reference and spawn our agent 10 actions (typically 1.9 meters) away.

EQA Accuracy. We compute overall accuracy as well as accuracy for each of the 6 types of questions in our dataset. In addition, we also categorize question difficulty level into easy, medium, and hard by binning the ground-truth action length. Easy questions are those with fewer than 25 action steps along the shortest path, medium are those with 25-70 actions, and hard are those with more than 70 actions. We report accuracy for each difficulty, $\%_{easy}$, $\%_{medium}$, $\%_{hard}$, as well as overall, $\%_{overall}$, in Table 5.5.

Navigation Accuracy. We also measure the navigation accuracy for both objects and rooms in MT-EQA. As each question involves several targets, the order of them being navigated matters. We consider the 'ground truth' ordering of targets for navigation as the order in which they are mentioned in the question, e.g., given *"Does the bathtub have same color as the sink?"*, the agent is trained and evaluated for visiting the "bathtub" first and then the "sink".

For each mentioned target object, we evaluate the agent's navigation performance by computing the distance to the target object at navigation termination, d_T, and change in distance to the target from initial spawned position to terminal position, d_Δ. We also compute the stop ratio $\%stop_o$ as in EQA-v1 Das et al. (2018a). Additionally, we propose two new metrics based on the IOU of the target object at its termination. When the navigation is done, we compute the IOU of the target w.r.t a centered rectangular box (see Fig. 5.3 as example). The first metric is mean IOU ratio $\text{IOU}_T^r = \frac{1}{N}\sum_i \frac{\text{IOU}_T(o_i)}{\text{IOU}_{best}(o_i)})$ where $\text{IOU}_{best}(o_i)$ is the highest IOU score for object o_i. The second is hit accuracy h_T – we compute the percentage of the ratio $\text{IOU}_T(o_i)/\text{IOU}_{best}(o_i)$ greater than 0.5, i.e., $h_T = \frac{1}{N}\sum_i ||\frac{\text{IOU}_T(o_i)}{\text{IOU}_{best}(o_i)} > 0.5||$. Both metrics measure to what extent the agent is looking at the target at termination.

For each mentioned target room, we evaluate the agent's navigation by recording the percentage of agents terminating inside the target room $\%r_T$ and the stop ratio $\%stop_r$.

		Object Navigation					Room Navigation		EQA				
		d_T	d_Δ	h_T	IOU_T^r	$\%stop_o$	$\%r_T$	$\%stop_r$	ep_len	$\%_{easy}$	$\%_{medium}$	$\%_{hard}$	$\%_{overall}$
1	Nav+cVQA	5.41	-0.64	0.19	0.15	36	34	60	153.13	58.42	53.29	51.46	53.24
2	Nav(RL)+cVQA	3.80	0.10	0.33	**0.30**	46	40	62	144.80	67.57	55.91	53.28	57.40
3	Nav+Ctrl+cVQA	5.25	-0.56	0.20	0.18	36	37	70	145.20	59.73	53.48	49.04	54.44
4	Nav(RL)+Ctrl+cVQA	**3.60**	**0.16**	**0.33**	0.29	**48**	**43**	**72**	**127.71**	**72.22**	**59.97**	**54.92**	**61.45**

Table 5.5: Quantitative evaluation of object/room navigation and EQA accuracy for different approaches.

		object_color_compare		object_size_compare		object_dist_compare	room_size_compare	$\%_{overall}$
		inroom	xroom	inroom	xroom	inroom	xroom	
1	Nav+cVQA	64.15	52.47	57.85	55.68	49.38	48.37	53.24
2	Nav(RL)+cVQA	71.24	53.92	74.38	60.81	51.23	46.66	57.40
3	Nav+Ctrl+cVQA	66.41	52.65	57.85	53.48	49.38	48.37	54.44
4	Nav(RL)+Ctrl+cVQA	**72.68**	**58.19**	**76.86**	**63.37**	**54.94**	**55.57**	**61.45**

Table 5.6: EQA accuracy on each question type for different approaches.

For all the above metrics except for d_T, larger is better. Additionally, we report the overall number of action steps (episode length) executed for each question, i.e., ep_len.

		object_color_compare		object_size_compare		object_dist_compare	room_size_compare	$\%_{overall}$
		inroom	xroom	inroom	xroom	inroom	xroom	
1	[BestView] + attn-VQA (cnn)	71.16	59.56	65.29	65.93	58.64	49.74	60.50
2	[BestView] + cVQA (cnn)	82.92	72.70	80.99	83.88	**69.75**	64.32	74.14
3	[ShortestPath+BestView] + Ctrl + cVQA	**90.70**	**85.4**9	**82.64**	**88.64**	68.52	**71.87**	**82.88**
4	[ShortestPath] + seq-VQA	53.32	54.44	51.24	50.55	47.53	49.74	52.36
5	[ShortestPath] + Ctrl + cVQA	76.09	69.11	75.21	79.49	64.20	61.23	69.77

Table 5.7: EQA accuracy of different approaches on each question type in oracle setting (given shortest path or best-view images).

5.5.2 EQA Results

Nav+Ctrl+cVQA is our full model, which is composed of a program generator, a navigator, a controller and a comparative VQA module. Another variant of our model, the REINFORCE fine-tuned model is denoted as Nav(RL)+Ctrl+cVQA. We also train a simplified version of our full model, Nav+cVQA. which does not use a controller. For this model, we let the navigator predict termination whenever a target is detected, then feed its hidden states to the VQA model. The training details are similar to our full model for both IL and RL. We show comparisons of both navigation and EQA accuracy in Table. 5.5.

RL helps both navigation and EQA accuracies. Both object and room navigation performance are improved after RL finetuning. We notice without finetuning d_Δ for both models (Row 1 &3) are negative, which means the agent has moved farther away from the target during navigation. After RL finetuning, d_Δ jumps from -0.56 to 0.16 (Row 3 & 4). The hit accuracy also improves from 20% to 33%, indicating that the RL-finetuned agent is more likely to find the target mentioned in the question. Episode lengths from the stronger navigators are shorter, indicating that better navigators find their target more quickly. And, higher EQA accuracy is also achieved with the help of RL finetuning (from 54.44% to 61.45%). After breaking down the EQA into different types, we observe the same trend in Table. 5.6 – our full model with RL far outperforms the others.

Controller is important. Comparing our full model (Row 4) to the one without a controller (Row 2), we notice that the former outperforms the latter across almost all the metrics. One possible reason is that the VQA task and navigation task are quite different, such that the features (hidden state) from the navigator cannot help improve the VQA module. On the contrary, our controller decouples the two tasks, letting the navigator and VQA module focus on their own roles.

Questions with shorter ground-truth path are easier. We observe that our agent is far better at dealing with easy questions than hard ones (72.22% over 54.92% in Table. 5.5 Row 4). One reason is that the targets mentioned in the easy questions, e.g., sink and toilet in *"Does the sink have same color as the toilet in the bathroom?"*, are typically closer to each other, thus are relatively easier to be explored, whereas questions like *"Is the kitchen bigger than the garage?"* requires a very long trajectory and the risk of missing one (kitchen or garage) is increased. The same observation is found in Table. 5.6, where we get higher accuracy for "in-room" questions than "cross-room" ones.

5.5.3 Oracle Comparisons

To better understand each module of our model, we run ablation studies. Table. 5.7 shows EQA accuracy of different approaches given the shortest paths or best-view frames.

Our VQA module helps. We first compare the performance of our VQA module against an attention-based VQA. Given the best view of each target, we can directly feed the features from those images to the VQA module, using the CNN features instead of hidden states from controller side. The attention-based VQA architecture is similar to Das et al. (2018a), which uses an LSTM to encode questions and then uses its representation to pool image features with attention. Comparing the two methods in Table. 5.7, Row 1 & 2, our VQA module achieves 13.64% higher accuracy. The benefit mainly comes from the decomposition of attribute representation and comparison in our VQA module.

Controller's features help. We compare the controller's features to raw CNN features for VQA. When given both shortest path and best-view position, we run our full model with these annotations and feed the hidden states from the controller's LSTM to our VQA model. As shown in Table. 5.7, Row 2 & 3, the controller's features are far better than raw CNN features, especially for object_color_compare and object_size_compare question types.

Controller's SELECT matters. Our controller predicts SELECT and extracts the features at that moment. One possible question is how important is this moment selection. To demonstrate its advantage, we trained another VQA module which uses a LSTM to encode the whole sequence of frames along the shortest path and uses its final hidden state to predict the answer, denoted as seq-VQA. The hypothesis is that the final hidden state might be able to encode all relevant information, as the LSTM has gone through the whole sequence of frames. Table. 5.7, Row 4, shows its results, which is nearly random. On the contrary, when controller is used to SELECT frames in Row 5, the results are far better. However, there is still much space for improvement. Comparing Table. 5.7, Row 3 & 5, the overall accuracy drops 13% when using features from the predicted SELECT instead of oracle moments, and 20% when using additional navigators

(comparing Table. 5.7, Row 3, & Table. 5.6, Row 4), indicating the necessity of both accurate SELECT and navigation.

CHAPTER 6: DISCUSSION AND FUTURE WORK

6.1 Summary of Contributions

We have reviewed our recent work on vision and language, including Visual Madlibs as question answering in Chapter 2, referring expression generation and comprehension in Chapter 3, album summarization and storytelling in Chapter 4, and embodied question answering in Chapter 5. There has been a great deal of progress on each of these tasks, largely due to the growing availability of large labeled datasets and neural learning based methods. Moving forward, we expect the vision and language tasks to move one more step into the real world where intelligent agents collaborate and communicate with people.

6.2 Future Directions

Though we have achieved substantial success on the above mentioned tasks, there still may exist some problems that is relevant to the future work. Special attention may be put on generalization ability and robustness for vision and language.

Consider referring expression comprehension as an example. While our MAttNet Yu et al. (2018) has achieved state-of-the-art performance, it suffers the lack of generalization. Note it was only trained on the expressions for the 80 MS COCO categories Lin et al. (2014), thus cannot comprehend sentences describing beyond the 80 categories, e.g., "red shoes on the tall boy", "the blue sky", "left hand side tree", etc. Such issue may be addressed by zero-shot learning techniques, among which a further breaking down of neural modules might be one possible solution. We hypothesize that the detection module could be popped out via deeper neural language parser, so that this part could be replaced by pre-trained larger-scale detector. We may also consider the

joint segmentation of instances and stuff Kirillov et al. (2018) for referring to everything. Besides, we should expect our system to judge if the input referring expression makes sense or not. For example, an input sentence “girl dancing” should be judged as fake expression if there was no girl in the picture.

Robustness is another issue. What we found in MAttNet or MT-EQA is once we move a few pixels horizontally or vertically on the image or change the view or illumination in the environment, our model may predict quite different result. One possible reason is our training data cannot cover all kinds of variant views, images and sentences, thus making some discrepancy between training and testing. Literature work Xu et al. (2018); Hsieh et al. (2018) shows current attention, localization and compositional internal structure is vulnerable to adversarial attack. How to generate or mine more informative data and learn from adversarial negatives might be key to address such issue.

REFERENCES

Agrawal, H., Chandrasekaran, A., Batra, D., Parikh, D., and Bansal, M. (2016). Sort story: Sorting jumbled images and captions into stories. In *EMNLP*.

Aker, A. and Gaizauskas, R. (2010). Generating image descriptions using dependency relational patterns. In *ACL*.

Anand, A., Belilovsky, E., Kastner, K., Larochelle, H., and Courville, A. (2018). Blindfold baselines for embodied qa. *arXiv preprint arXiv:1811.05013*.

Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. (2018a). Bottom-up and top-down attention for image captioning and visual question answering. *CVPR*.

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and van den Hengel, A. (2018b). Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*.

Andreas, J. and Klein, D. (2016). Reasoning about pragmatics with neural listeners and speakers. *EMNLP*.

Andreas, J., Klein, D., and Levine, S. (2017). Modular multitask reinforcement learning with policy sketches. *ICML*.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016a). Learning to compose neural networks for question answering. *NAACL*.

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. (2016b). Neural module networks. In *CVPR*.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., et al. (2016). Deepmind lab. *arXiv preprint arXiv:1612.03801*.

Bell, S., Zitnick, C. L., Bala, K., and Girshick, R. B. (2016). Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. *CVPR*.

Berg, A. C., Berg, T. L., III, H. D., Dodge, J., Goyal, A., Han, X., Mensch, A., Mitchell, M., Sood, A., Stratos, K., and Yamaguchi, K. (2012). Understanding and predicting importance in images. In *CVPR*.

Bordes, A., Chopra, S., and Weston, J. (2014a). Question answering with subgraph embeddings. In *EMNLP*.

Bordes, A., Weston, J., and Usunier, N. (2014b). Open question answering with weakly supervised embedding models. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer.

Brodeur, S., Perez, E., Anand, A., Golemo, F., Celotti, L., Strub, F., Rouat, J., Larochelle, H., and Courville, A. (2017). Home: A household multimodal environment. *arXiv preprint arXiv:1711.11017*.

Brown-Schmidt, S. and Tanenhaus, M. K. (2006). Watching the eyes when talking about size: An investigation of message formulation and utterance planning. *Journal of Memory and Language*.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R. (2018). Gated-attention architectures for task-oriented language grounding. In *AAAI*.

Chen, X. and Lawrence Zitnick, C. (2015). Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*.

Cheng, J. and Lapata, M. (2016). Neural summarization by extracting sentences and words. In *ACL*.

Choi, J., Oh, T.-H., and Kweon, I. S. (2017). Textually customized video summaries. *arXiv preprint arXiv:1702.01528*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *JMLR*.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. (2018a). Embodied question answering. *CVPR*.

Das, A., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. (2018b). Neural Modular Control for Embodied Question Answering. *CoRL*.

Das, A., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. (2018c). Neural modular control for embodied question answering. *arXiv preprint arXiv:1810.11181*.

De Marneffe, M.-C., MacCartney, B., Manning, C. D., et al. (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*.

Dhiraj Gandhi, Lerrel Pinto, A. G. (2017). Learning to fly by crashing. *IROS*.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*.

Efron, B. and Tibshirani, R. J. (1994). *An introduction to the bootstrap*. CRC press.

Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. (2014). Scalable object detection using deep neural networks. In *CVPR*.

Fang, H., Gupta, S., Iandola, F., Srivastava, R. K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J. C., et al. (2015). From captions to visual concepts and back. In *CVPR*.

Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., and Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. In *ECCV*.

Feng, Y. and Lapata, M. (2010). Topic models for image annotation and text illustration. In *ACL*.

FitzGerald, N., Artzi, Y., and Zettlemoyer, L. S. (2013). Learning distributions over logical forms for referring expression generation. In *EMNLP*.

Fukui, A., Park, D. H., Yang, D., Rohrbach, A., Darrell, T., and Rohrbach, M. (2016). Multimodal compact bilinear pooling for visual question answering and visual grounding. In *EMNLP*.

Gatt, A. and Belz, A. (2009). Introducing shared tasks to nlg: The tuna shared task evaluation challenges. In *EMNLP*. Springer.

Geman, D., Geman, S., Hallonquist, N., and Younes, L. (2015). Visual turing test for computer vision systems. *Proceedings of the National Academy of Sciences*.

Girshick, R. (2015). Fast r-cnn. In *ICCV*.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*.

Gong, B., Chao, W.-L., Grauman, K., and Sha, F. (2014a). Diverse sequential subset selection for supervised video summarization. In *NIPS*.

Gong, Y., Ke, Q., Isard, M., and Lazebnik, S. (2014b). A multi-view embedding space for modeling internet images, tags, and their semantics. *IJCV*.

Gordon, D., Kembhavi, A., Rastegari, M., Redmon, J., Fox, D., and Farhadi, A. (2018). Iqa: Visual question answering in interactive environments. In *CVPR*.

Grubinger, M., Clough, P., Müller, H., and Deselaers, T. (2006). The iapr tc-12 benchmark: A new evaluation resource for visual information systems. In *International workshop ontoImage*.

Gupta, S., Davidson, J., Levine, S., Sukthankar, R., and Malik, J. (2017). Cognitive mapping and planning for visual navigation. In *CVPR*.

Gygli, M., Grabner, H., and Van Gool, L. (2015). Video summarization by learning submodular mixtures of objectives. In *CVPR*.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *ICCV*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Hendricks, L. A., Wang, O., Shechtman, E., Sivic, J., Darrell, T., and Russell, B. C. (2017). Localizing moments in video with natural language. *ICCV*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*.

Hsieh, C.-J., Yi, J., Chen, H., Zhang, H., and Chen, P.-Y. (2018). Attacking visual language grounding with adversarial examples: A case study on neural image captioning. In *ACL*.

Hu, R., Andreas, J., Rohrbach, M., Darrell, T., and Saenko, K. (2017a). Learning to reason: End-to-end module networks for visual question answering. *ICCV*.

Hu, R., Rohrbach, M., and Darrell, T. (2016a). Segmentation from natural language expressions. In *ECCV*.

Hu, R., Rohrbacnh, M., Andreas, J., Darrell, T., and Saenko, K. (2017b). Modeling relationship in referential expressions with compositional modular networks. In *CVPR*.

Hu, R., Xu, H., Rohrbach, M., Feng, J., Saenko, K., and Darrell, T. (2016b). Natural language object retrieval. *CVPR*.

Huang, T.-H. K., Ferraro, F., Mostafazadeh, N., Misra, I., Agrawal, A., Devlin, J., Girshick, R., He, X., Kohli, P., Batra, D., et al. (2016). Visual storytelling. In *NACCL*.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R. (2017a). Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*.

Johnson, J., Hariharan, B., van der Maaten, L., Hoffman, J., Fei-Fei, L., Zitnick, C. L., and Girshick, R. (2017b). Inferring and executing programs for visual reasoning. *ICCV*.

Johnson, J., Karpathy, A., and Fei-Fei, L. (2016). Densecap: Fully convolutional localization networks for dense captioning. *CVPR*.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

Kazemzadeh, S., Ordonez, V., Matten, M., and Berg, T. (2014). Referitgame: Referring to objects in photographs of natural scenes. In *EMNLP*.

Kempka, M., Wydmuch, M., Runc, G., Toczek, J., and Jaśkowski, W. (2016). Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*.

Khosla, A., Hamid, R., Lin, C.-J., and Sundaresan, N. (2013). Large-scale video summarization using web-image priors. In *CVPR*.

Kim, G., Moon, S., and Sigal, L. (2015). Joint photo stream and blog post summarization and exploration. In *CVPR*.

Kim, G. and Xing, E. P. (2014). Reconstructing storyline graphs for image recommendation from web community photos. In *CVPR*.

Kirillov, A., He, K., Girshick, R. B., Rother, C., and Dollár, P. (2018). Panoptic segmentation. *CVPR*.

Kiros, R., Salakhutdinov, R., and Zemel, R. S. (2014). Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*.

Kolve, E., Mottaghi, R., Gordon, D., Zhu, Y., Gupta, A., and Farhadi, A. (2017). Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Krishna, R., Hata, K., Ren, F., Fei-Fei, L., and Niebles, J. C. (2017). Dense-captioning events in videos. *ICCV*.

Krishnamoorthy, N., Malkarnenkar, G., Mooney, R., Saenko, K., and Guadarrama, S. (2013). Generating natural-language video descriptions using text-mined knowledge. In *AAAI*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kulesza, A., Taskar, B., et al. (2012). Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*.

Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., and Berg, T. L. (2011). Baby talk: Understanding and generating image descriptions. In *CVPR*.

Kulkarni, G., Premraj, V., Ordonez, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., and Berg, T. (2013). Babytalk: Understanding and generating simple image descriptions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*.

Kuznetsova, P., Ordonez, V., Berg, A. C., Berg, T. L., and Choi, Y. (2012). Collective generation of natural image descriptions. In *ACL*.

Lebret, R., Pinheiro, P. O., and Collobert, R. (2015). Phrase-Based Image Captioning. In *ICML*.

Lei, J., Yu, L., Bansal, M., and Berg, T. L. (2018). Tvqa: Localized, compositional video question answering. In *EMNLP*.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *JMLR*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*.

Lin, X. and Parikh, D. (2015). Don't just listen, use your imagination: Leveraging visual common sense for non-visual tasks. *arXiv preprint arXiv:1502.06108*.

Liu, B., Yeung, S., Chou, E., Huang, D.-A., Fei-Fei, L., and Niebles, J. C. (2018). Temporal modular networks for retrieving complex compositional activities in videos. In *ECCV*.

Liu, C., Lin, Z., Shen, X., Yang, J., Lu, X., and Yuille, A. (2017a). Recurrent multimodal interaction for referring image segmentation. In *ICCV*.

Liu, J., Wang, L., and Yang, M.-H. (2017b). Referring expression generation and comprehension via attributes. In *ICCV*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. E., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *ECCV*.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *CVPR*.

Lu, Z. and Grauman, K. (2013). Story-driven summarization for egocentric video. In *CVPR*.

Luo, R. and Shakhnarovich, G. (2017). Comprehension-guided referring expressions. *CVPR*.

Malinowski, M. and Fritz, M. (2014). A multi-world approach to question answering about real-world scenes based on uncertain input. In *NIPS*.

Mao, J., Huang, J., Toshev, A., Camburu, O., Yuille, A. L., and Murphy, K. (2016). Generation and comprehension of unambiguous object descriptions. In *CVPR*.

Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., and Yuille, A. (2015). Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*.

Mason, R. (2013). Domain-independent captioning of domain-specific images. In *HLT-NAACL*.

Mei, H., Bansal, M., and Walter, M. R. (2016). What to talk about and how? selective generation using lstms with coarse-to-fine alignment. In *NAACL*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mitchell, M., Han, X., Dodge, J., Mensch, A., Goyal, A., Berg, A., Yamaguchi, K., Berg, T., Stratos, K., and Daumé III, H. (2012). Midge: Generating image descriptions from computer vision detections. In *EACL*.

Mitchell, M., Reiter, E., and van Deemter, K. (2013a). Typicality and object reference. In *Cognitive Science (CogSci)*.

Mitchell, M., van Deemter, K., and Reiter, E. (2010). Natural reference to objects in a visual domain. In *International Natural Language Generation Conference (INLG)*.

Mitchell, M., Van Deemter, K., and Reiter, E. (2013b). Generating expressions that refer to visible objects. In *HLT-NAACL*.

Nagaraja, V. K., Morariu, V. I., and Davis, L. S. (2016). Modeling context between objects for referring expression understanding. In *ECCV*.

Ordonez, V., Kulkarni, G., and Berg, T. L. (2011). Im2text: Describing images using 1 million captioned photographs. In *NIPS*.

Pan, Y., Mei, T., Yao, T., Li, H., and Rui, Y. (2016). Jointly modeling embedding and translation to bridge video and language. In *CVPR*.

Park, C. C. and Kim, G. (2015). Expressing an image stream with a sequence of natural sentences. In *NIPS*.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *EMNLP*.

Rashtchian, C., Young, P., Hodosh, M., and Hockenmaier, J. (2010). Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Rohrbach, A., Rohrbach, M., Hu, R., Darrell, T., and Schiele, B. (2016a). Grounding of textual phrases in images by reconstruction. *ECCV*.

Rohrbach, A., Torabi, A., Rohrbach, M., Tandon, N., Pal, C., Larochelle, H., Courville, A., and Schiele, B. (2016b). Movie description. *IJCV*.

Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *EMNLP*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *IJCV*.

Sadeghi, F. and Levine, S. (2017). CAD2RL: Real single-image flight without a single real image. *RSS*.

Sadeghi, F., Zitnick, C. L., and Farhadi, A. (2015). Visalogy: Answering visual analogy questions. In *NIPS*.

Savva, M., Chang, A. X., Dosovitskiy, A., Funkhouser, T., and Koltun, V. (2017). MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv:1712.03931*.

Sigurdsson, G. A., Chen, X., and Gupta, A. (2016). Learning visual storylines with skipping recurrent neural networks. In *ECCV*.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Socher, R., Bauer, J., Manning, C. D., et al. (2013). Parsing with compositional vector grammars. In *ACL*.

Song, S., Yu, F., Zeng, A., Chang, A. X., Savva, M., and Funkhouser, T. (2017). Semantic scene completion from a single depth image. In *CVPR*.

Su, J.-C., Wu, C., Jiang, H., and Maji, S. (2017). Reasoning about fine-grained attribute phrases using reference games. *ICCV*.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). Weakly supervised memory networks. In *NIPS*.

Synnaeve, G., Nardelli, N., Auvolat, A., Chintala, S., Lacroix, T., Lin, Z., Richoux, F., and Usunier, N. (2016). Torchcraft: a library for machine learning research on real-time strategy games. *arXiv preprint arXiv:1611.00625*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *CVPR*.

van Deemter, K., van der Sluis, I., and Gatt, A. (2006). Building a semantically transparent corpus for the generation of referring expressions. In *International Conference on Natural Language Generation (INLG)*.

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. (2015). Sequence to sequence-video to text. In *ICCV*.

Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., and Saenko, K. (2014). Translating videos to natural language using deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*.

Viethen, J. and Dale, R. (2008a). The use of spatial relations in referring expression generation. In *International Natural Language Generation Conference (INLG)*.

Viethen, J. and Dale, R. (2008b). The use of spatial relations in referring expression generation. In *Proceedings of the Fifth International Natural Language Generation Conference*.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015a). Pointer networks. In *NIPS*.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015b). Show and tell: A neural image caption generator. In *CVPR*.

Wang, L., Li, Y., and Lazebnik, S. (2016). Learning deep structure-preserving image-text embeddings. *CVPR*.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Woodsend, K. and Lapata, M. (2010). Automatic generation of story highlights. In *ACL*.

Wu, Q., Shen, C., Wang, P., Dick, A., and van den Hengel, A. (2017). Image captioning and visual question answering based on attributes and external knowledge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wu, Y., Wu, Y., Gkioxari, G., and Tian, Y. (2018). Building generalizable agents with a realistic and rich 3d environment. *ICLR workshop*.

Xia, F., Zamir, A. R., He, Z., Sax, A., Malik, J., and Savarese, S. (2018). Gibson env: Real-world perception for embodied agents. In *CVPR*.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *ICML*.

Xu, X., Chen, X., Liu, C., Rohrbach, A., Darrell, T., and Song, D. X. (2018). Fooling vision and language models despite localization and attention mechanism. *CVPR*.

Yang, Y., Teo, C. L., Daumé III, H., and Aloimonos, Y. (2011). Corpus-guided sentence generation of natural images. In *EMNLP*.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A. J., and Hovy, E. H. (2016). Hierarchical attention networks for document classification. In *HLT-NAACL*.

Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., and Courville, A. (2015). Describing videos by exploiting temporal structure. In *ICCV*.

Yao, T., Pan, Y., Li, Y., Qiu, Z., and Mei, T. (2016). Boosting image captioning with attributes. *arXiv preprint arXiv:1611.01646*.

You, Q., Jin, H., Wang, Z., Fang, C., and Luo, J. (2016). Image captioning with semantic attention. In *CVPR*.

Young, P., Lai, A., Hodosh, M., and Hockenmaier, J. (2014). From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*.

Yu, H., Wang, J., Huang, Z., Yang, Y., and Xu, W. (2016a). Video paragraph captioning using hierarchical recurrent neural networks. In *CVPR*.

Yu, L., Bansal, M., and Berg, T. L. (2017a). Hierarchically-attentive rnn for album summarization and storytelling. In *EMNLP*.

Yu, L., Chen, X., Gkioxari, G., Bansal, M., Berg, T. L., and Batra, D. (2019). Multi-target embodied question answering. In *CVPR*.

Yu, L., Lin, Z., Shen, X., Yang, J., Lu, X., Bansal, M., and Berg, T. L. (2018). Mattnet: Modular attention network for referring expression comprehension. In *CVPR*.

Yu, L., Park, E., Berg, A. C., and Berg, T. L. (2015). Visual madlibs: Fill in the blank image generation and question answering. In *ICCV*.

Yu, L., Poirson, P., Yang, S., Berg, A. C., and Berg, T. L. (2016b). Modeling context in referring expressions. In *ECCV*.

Yu, L., Tan, H., Bansal, M., and Berg, T. L. (2017b). A joint speaker-listener-reinforcer model for referring expressions. In *CVPR*.

Zhang, K., Chao, W.-L., Sha, F., and Grauman, K. (2016a). Summary transfer: Exemplar-based subset selection for video summarization. In *CVPR*.

Zhang, K., Chao, W.-L., Sha, F., and Grauman, K. (2016b). Video summarization with long short-term memory. In *ECCV*.

Zhu, Y., Gordon, D., Kolve, E., Fox, D., Fei-Fei, L., Gupta, A., Mottaghi, R., and Farhadi, A. (2017a). Visual semantic planning using deep successor representations. In *ICCV*.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. (2017b). Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*.