Multimodal Learning for Vision and Language

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Junhua Mao

2017
ABSTRACT OF THE DISSERTATION

Multimodal Learning for Vision and Language

by

Junhua Mao
Doctor of Philosophy in Statistics
University of California, Los Angeles, 2017
Professor Alan Loddon Yuille, Chair

This thesis focuses on proposing and addressing various tasks in the field of vision and language, a new and challenging area which contains the hottest research topics for both computer vision and natural language processing. We first proposed an effective RNN-CNN framework (Recurrent Neural Network-Convolutional Neural Network) to address the task of image captioning (i.e. describing an image with a sentence). Based on this work, we proposed effective models and constructed large-scale datasets, for various vision and language tasks, such as unambiguous object descriptions (i.e. Referring expressions), image question answering, one-shot novel concept captioning, multimodal word embedding, and multi-label classification. Many of these tasks have not been successfully addressed or even been investigated before. Our work are among the first deep learning effort for these tasks, and achieves the state-of-the-art results. We hope the methods and datasets proposed in this thesis could provide insight for the future development of vision and language.
The dissertation of Junhua Mao is approved.

Demetri Terzopoulos
Hongjing Lu
Ying Nian Wu
Alan Loddon Yuille, Committee Chair

University of California, Los Angeles
2017
To my parents . . .

who—among so many other things—
always support me no matter what happens.
# TABLE OF CONTENTS

1 Introduction .................................................................................. 1

2 Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN) [MXY15a] 7
   2.1 Abstract ............................................................................... 7
   2.2 Introduction ......................................................................... 7
   2.3 Related Work ..................................................................... 8
   2.4 Model Architecture ............................................................... 11
      2.4.1 Simple recurrent neural network .................................. 11
      2.4.2 Our m-RNN model ....................................................... 12
   2.5 Training the m-RNN ............................................................... 14
   2.6 Sentence Generation, Image Retrieval and Sentence Retrieval .......... 15
   2.7 Learning of Sentence and Image Features ................................. 16
   2.8 Experiments ....................................................................... 17
      2.8.1 Datasets ........................................................................ 17
      2.8.2 Evaluation metrics ......................................................... 18
      2.8.3 Results on IAPR TC-12 ................................................. 18
      2.8.4 Results on Flickr8K ...................................................... 20
      2.8.5 Results on Flickr30K and MS COCO ......................... 21
   2.9 Nearest Neighbor as Reference .............................................. 23
      2.9.1 Image features for the nearest neighbor image search .......... 24
      2.9.2 Consensus Reranking ..................................................... 26
      2.9.3 Experiments ................................................................. 26
      2.9.4 Discussion ................................................................... 26
# Learning like a Child: Fast Novel Visual Concept Learning from Sentence Descriptions of Images [MWY15]

## Abstract

## Introduction

## Related Work

## The Image Captioning Model

### The Model Architecture

### The Transposed Weight Sharing (TWS)

## The Novel Concept Learning (NVCS) Task

### Fixing the originally learned weights

### Fixing the baseline probability

### The Role of Language and Vision

## Datasets

### Strategies to Construct Datasets

### The Novel Visual Concepts Datasets

## Experiments

### Evaluation Metrics

### Effectiveness of TWS and BPF

### Results on NewObj-Motor and NewObj-Car

### Results on NC-3

### Qualitative Results

## Conclusion
LIST OF FIGURES

2.1 Examples of the generated and two top-ranked retrieved sentences given the query image from IAPR TC-12 dataset. The sentences can well describe the content of the images. We show a failure case in the fourth image, where the model mistakenly treats the lake as the sky and misses all the people. More examples from the MS COCO dataset can be found on the project page: www.stat.ucla.edu/
\protect\unhbox\voidb@x\penalty\@M\{\}junhua.mao/m-RNN.html

2.2 Illustration of the simple Recurrent Neural Network (RNN) and our multimodal Recurrent Neural Network (m-RNN) architecture. (a). The simple RNN. (b). Our m-RNN model. The inputs of our model are an image and its corresponding sentence descriptions. $w_1, w_2, ..., w_L$ represents the words in a sentence. We add a start sign $w_{start}$ and an end sign $w_{end}$ to all the training sentences. The model estimates the probability distribution of the next word given previous words and the image. It consists of five layers (i.e. two word embedding layers, a recurrent layer, a multimodal layer and a softmax layer) and a deep CNN in each time frame. The number above each layer indicates the dimension of the layer. The weights are shared among all the time frames. (Best viewed in color)

2.3 The sample images and their nearest neighbors retrieved by two types of features. Compared to the original VggNet features, the features refined by the m-RNN model are better for capturing richer and more accurate visual information.

2.4 The original rank of the hypotheses and the rank after consensus reranking (CIDEr).

3.1 An illustration of the Novel Visual Concept learning from Sentences (NVCS) task. We start with a model (i.e. model-base) trained with images that does not contain the concept of “quidditch”. Using a few “quidditch” images with sentence descriptions, our method is able to learn that “quidditch” is played by people with a ball.
3.2 (a). The architecture of our image captioning model. For each word in a sentence, the model takes the current word index and the image as inputs, and outputs the next word index. The weights are shared across the sub-models for the words in a sentence. The number on the top right of each layer denotes its dimension. As in the m-RNN model [MXY15b], we add a start sign $w_{\text{start}}$ and an end sign $w_{\text{end}}$ to each training sentence. (b). The transposed weight sharing of $U_D$ and $U_M$. (Best viewed in color) ................................................................. 31

3.3 Illustration of the training for novel concepts. We only update the sub-matrix $U_{D_n}$ (the green parts in the figure) in $U_D$ that is connected to the node of new words in the One-Hot layer and the SoftMax layer during the training for novel concepts. (Best viewed in color) ................................................................. 35

3.4 Organization of the novel concept datasets ................................................................. 37

3.5 Sample images and annotations from Novel Concept-3 (NC-3) dataset ..................... 38

3.6 Performance comparison of our model with different number of training images on NewObj-cat and NewObj-Motor datasets. The red, blue, and magenta dashed line indicates the performance of Deep-NVCS using all the training images on the base test set, the NC test set and the all test set respectively. The green dashed line indicates the performance of Model-base. We show that our model trained with 10 to 50 images achieves comparable performance with the model trained on the full training set. (Best viewed in color) ................................................................. 42

3.7 The generated sentences for the test images from novel concept datasets. In these examples, cat, motorcycle, quidditch, t-rex and samisen are the novel concepts respectively. We show more in the supplementary material. ............................... 45
4.1 Illustration of our generation and comprehension system. On the left we see that the system is given an image and a region of interest; it describes it as “the man who is touching his head”, which is unambiguous (unlike other possible expressions, such as “the man wearing blue”, which would be unclear). On the right we see that the system is given an image, an expression, and a set of candidate regions (bounding boxes), and it selects the region that corresponds to the expression.

4.2 Some sample images from our Google Refexp (G-Ref) dataset. We use a green dot to indicate the object that the descriptions refer to. Since the dataset is based on MS COCO, we have access to the original annotations such as the object mask and category. Some of the objects are hard to describe, e.g., in the third image in the first row, we need to distinguish the boy from his reflection in the mirror.

4.3 Comparison between the G-Ref and UNC-Ref dataset.

4.4 Illustration of the baseline model architecture. ⟨bos⟩ and ⟨eos⟩ stand for beginning and end of sentence.

4.5 Illustration of how we train the full model using the softmax loss function. \( R \) (green) is the target region, \( R' \) are the incorrect regions. The weights of the LSTMs and CNNs are shared for \( R \) and \( R' \)'s. (Best viewed in color)

4.6 Illustration of the semi-supervised training process. See text for details.

4.7 The sample results of the description generation using our full model (above the dashed line) and the strong baseline (below the dashed line). The descriptions generated by our full model are more discriminative than those generated by the baseline.
4.8 Sample results of the description comprehension task using our full model. The first and second column shows the original image and the multibox proposals. The third to sixth columns show the results of our model when input an arbitrary description of an object in the image. The red bounding box denotes the most probable object predicted by the model while the blue dashed ones denote the bounding boxes within the margin of the most probable one. The descriptions can be the groundtruth ones in the dataset (third column) or an customized descriptions (fourth to sixth columns). (Best viewed in color).

5.1 Sample answers to the visual question generated by our model on the newly proposed Freestyle Multilingual Image Question Answering (FM-IQA) dataset.

5.2 Illustration of the mQA model architecture. We input an image and a question about the image (i.e. “What is the cat doing?”) to the model. The model is trained to generate the answer to the question (i.e. “Sitting on the umbrella”). The weight matrix in the word embedding layers of the two LSTMs (one for the question and one for the answer) are shared. In addition, as in [MWY15], this weight matrix is also shared, in a transposed manner, with the weight matrix in the Softmax layer. Different colors in the figure represent different components of the model. (Best viewed in color.)

5.3 Sample images in the FM-IQA dataset. This dataset contains 316,193 Chinese question-answer pairs with corresponding English translations.

5.4 Random examples of the answers generated by the mQA model with score “1” given by the human judges.

5.5 The sample generated questions by our model and their answers.

5.6 Failure cases of our mQA model on the FM-IQA dataset.

6.1 Sample images and their sample descriptions collected from Pinterest.
6.2 The illustration of the positive word/phrase pairs generation. We calculate a score for each annotation (i.e. a short phrase describes the items) by aggregating the click frequency of the items to which it belongs and rank them according to the score. The final list of positive phrases are generated from the top ranked phrases after removing phrases containing overlapping words with the user query phrase. See text for details.

6.3 The interface for the annotators. They are required to choose which phrase (positive and negative phrases will be randomly labeled as “A” or “B”) is more related to base phrase. They can click on the phrases to see Google search results.

6.4 The illustration of the structures of our model A, B, and C. We use a CNN to extract visual representations and use a RNN to model sentences. The numbers on the bottom right corner of the layers indicate their dimensions. We use a sampled softmax layer with 1024 negative words to accelerate the training. Model A, B, and C differ from each other by the way that we fuse the visual representation into the RNN. See text for more details.

6.5 t-SNE [MH08] visualization of the 500 most frequent words learned by our Model A.
LIST OF TABLES

2.1 Results of the sentence generation task on the IAPR TC-12 dataset. “B” is short for BLEU. .......................................................... 19

2.2 R@K and median rank (Med r) for IAPR TC-12 dataset. ................................................. 19

2.3 Results of R@K and median rank (Med r) for Flickr8K dataset. “-AlexNet” denotes the image representation based on AlexNet extracted from the whole image frame. “-RCNN” denotes the image representation extracted from possible objects detected by the RCNN algorithm. ................................................... 20

2.4 Results of R@K and median rank (Med r) for Flickr30K dataset and MS COCO dataset. .......................................................... 21

2.5 Results of generated sentences on the Flickr 30K dataset and MS COCO dataset. .................. 22

2.6 Properties of the recurrent layers for the five very recent methods. LRCN has a stack of four 1000 dimensional LSTM layers. We achieves state-of-the-art performance using a relatively small dimensional recurrent layer. LSTM ([HS97]) can be treated as a sophisticated version of the RNN. .................................................. 22

2.7 Results of the MS COCO test set evaluated by MS COCO evaluation server ......................... 23

2.8 Results of m-RNN-shared model after applying consensus reranking using nearest neighbors as references (m-RNN-shared-NNref), compared with those of the original m-RNN model on our validation set and MS COCO test server. .......................... 25

3.1 The number of images for the three datasets .......................................................... 39

3.2 The performance of our model compared with the state-of-the-art for the standard image captioning task. .................................................. 41

3.3 Performance of Deep-NVCS models with different novel concept learning strategies on NewObj-Cat. TWS and BPF improves the performance. .......................... 42
3.4 Results of our model on the NewObj-Cat and NewObj-Motor dataset using all the training samples. Model-base and Model-retrain stand for the model trained on base set (no novel concepts) and the model retrained on the combined data (all the images of base set and novel concept set) respectively. Deep-NVCS stands for model trained only with the new concept data. Deep-NVCS-1:1Inc stands for the Deep-NVCS model trained by adding equal number of training images from the base set. Compared to Model-base which is only trained on the base set, the Deep-NVCS models achieve a significant improvement on the novel concept test set while reaching comparable performance on the base test set. The performance of Deep-NVCS models only slightly worse than that of the strong baseline Model-retrain, which requires much more training.

3.5 Results of our model on the NC-3 Datasets.

3.6 The five nearest neighbors of the new words as measured by the activation of the word-embedding layer.

4.1 We measure precision@1 on the UNC-Ref validation data. Each row is a different way of training the model. The columns show performance on ground truth or multibox proposals, and ground truth (human) or generated descriptions. Thus the columns with GT descriptions evaluate the performance of the comprehension system, and the columns with GEN descriptions evaluate (in an end-to-end way) the performance of the generation system.

4.2 Precision@1 for the baseline (ML) method and our full model with the max-margin objective function on various datasets.

4.3 Performance of our full model when trained on a small strongly labeled dataset vs training on a larger dataset with automatically labeled data.

5.1 The results of our mQA model for our FM-IQA dataset.

5.2 Performance comparisons of the different mQA variants.
6.1 Scale comparison with other image descriptions benchmarks 92
6.2 Sample triplets from the Gold RPI0K dataset 95
6.3 Performance comparison of our Model A, B, C, and their variants 99
ACKNOWLEDGMENT

Thanks ...

First and foremost, to my great adviser, Professor Alan L. Yuille. I worked with him since I was a senior undergraduate in the summer of 2012. He lead me into this fascinating world of computer vision research with his passion, his deep understanding about the fundamental problems in the field, and his enlightening instructions. He is such a smart and considerate mentor. He always gave me the freedom to try my ideas and helped me if I got stuck, even if we hold different opinions.

To my thesis committee, Professor Demetri Terzopoulos, Professor Ying Nian Wu, and Professor Hongjing Lu, for their feedback and advice of this work.

To many great researchers and engineers whom I was lucky to collaborate with during the pursuit of Ph.D. degree. Thank Wei Xu, Yi Yang, and Jiang Wang, who lead me into the field of vision and language and open a new door for my research. I will never forget their patient help and suggestions when I am an intern who even barely know how to code in python at Baidu Research Sunnyvale office three years ago. Thank Kevin Murphy, Jonathan Huang, and Alex Toshev, for many helpful suggestions and comments during my internship at Google Research. Thank Jiajing Xu and Yushi Jing, for the wonderful internship at Pinterest. I learned how to applied my research to real world applications from you. Thank Congcong Li and Ury Zhilinsky, for leading me into another wonderful field of self-driving cars. I will try my best to make it happens.

To many friends I met during this journey. Thank all the group members I have worked with of Center for Cognition, Vision, and Learning (CCVL): Boyan Bonev, Xianjie Chen, Nam-gyu Cho, Xiaodi Hou, Xiaochen Lian, Xiaobai Liu, Roozbeh Mottaghi, Weichao Qiu, Zhou Ren, Chunyu Wang, Jianyu Wang, Peng Wang, Fangting Xia, Bo Xin, Xingyao Ye, Jun Zhu, Yu Zhu, Vittal Premachandran, Xuan Dong, Liang-Chieh Chen, Alex Wong, Chenxi Liu, Zhuotun Zhu, Ehsan Jahangiri, Zhishuai Zhang, Siyuan Qiao, Cihang Xie, Wei Shen, John Flynn, Yuan Gao, Chang Ma, Xinggang Wang, Lanlan Liu, Jian Dong, Yukun Zhu, Wenbin Jiang, and George Papandreou. Thank all my fellow interns and friends from Baidu Research, Google Research, Pinterest and Google X.

Last but not least, to my family for their support. Thank my father, Zhao Mao and mother Jian
Yang, for their unconditional love, understanding, and support. Thank my girlfriend, Meihui Xie, for accompanying and tolerating me throughout the journey. I could not write any sentence to express my love sufficiently to them.
VITA

2009–2013 Bachelor of Engineering, University of Science and Technology of China (USTC). Awarded with Guo Moruo Scholarship (The supreme honor for the USTC undergraduates), and the Best Bachelor’s Thesis Award.

2013–2017 Research Assistant, Center for Cognition, Vision, and Learning at UCLA, Los Angeles, California

2014.6–2014.9 Research Intern, Institute of Deep Learning, Baidu Research, Sunnyvale, California

2015.6–2015.9 Research Intern, Google Research, Mountain View, California

2016.1–2016.4 Research Intern, Visual Discovery, Pinterest Inc., San Francisco, California

2016.6–2016.9 Software Engineering Intern, Google X, Mountain View, California

PUBLICATIONS

Junhua Mao, Jiajing Xu, Yushi Jing, Alan Yuille, Training and Evaluating Multimodal Word Embeddings with Large-scale Web Annotated Images, *NIPS 2016*

Chenxi Liu, Junhua Mao, Fei Sha, Alan Yuille, Attention Correctness in Neural Image Captioning, *AAAI 2017*

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, Kevin Murphy, Generation and Comprehension of Unambiguous Object Descriptions, *CVPR 2016*

Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu, Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering, *NIPS 2015*


Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang and Alan L. Yuille, Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN), *ICLR 2015*

Jun Zhu, Junhua Mao, and Alan L. Yuille, Learning from Weakly Supervised Data by the Expectation Loss SVM (e-SVM), *NIPS 2014*

Junhua Mao, Jun Zhu, and Alan L. Yuille, An Active Patch Model for Real World Texture and Appearance Classification, *ECCV 2014*

Junhua Mao, Houqiang Li, Wengang Zhou, Shuicheng Yan and Qi Tian, Scale-based Region Growing For Scene Text Detection, *ACMMM 2013*
CHAPTER 1

Introduction

Recently, the field of **vision and language** becomes one of the hottest research areas for both computer vision and natural language processing. Indeed, human, as the oracle and target model of computer vision and natural language processing, is a perfect multimodal, multi-task learning machine. Our extraordinary ability to see the world, think, create and communicate ideas are fused and coordinated so perfectly in our body, through millions of years of evolution. Therefore, it is natural to investigate this cross-domain field of vision and language. In addition, recent advance of deep learning methods for both vision and language provides effective models for both fields, and lays the foundation for the research in this field.

I am lucky enough to enter and explore this new and exciting, yet challenging field during my PhD. This thesis will present my research effort to this field. It starts with the task of **image captioning**, which is a basic but important task for vision and language. In this task, the model needs to give a free-form natural language sentence description of an image. We publish the first paper using a RNN-CNN (Recurrent Neural Network - Convolutional Neural Network) framework to address this task [MXY14] (extended version [MXY15a], See Chapter 2 for more details). The same model can be also used to address two retrieval tasks: the sentence retrieval task given input images and the image retrieval task given input sentences. This RNN-CNN framework outperforms previous state-of-the-arts by a large margin. Shortly after [MXY14], several papers using deep learning techniques appear with record breaking results [VTB15 KF15 DAG15 KSZ14 CZ14]. Many of them are built on the RNN-CNN framework. This draws great attention and interest of the machine learning communities to this new field, and intrigues the rapid development of this field.

The task of image captioning is exiting to researchers mainly because of three reasons: Firstly, it presents the rapid improvement of the machine learning research. It is very impressive to show that
the computer can generate a human-level and complex sentence description of an image. Secondly, different from previous tasks and models that use clean annotations of object categories, it shows that machines can learn to understand the image using noisy and weak labels (e.g. sentence descriptions). Thirdly, it demonstrates the great potential to combine the research in the field of computer vision and natural language processing together.

These deep neural network based image captioning systems need a large-scale training dataset (e.g. one of the standard image captioning benchmark, the MSCOCO dataset [LMB14], contains around 120,000 train/val images, and 5 sentences per image). However, no matter how large the dataset is, we will always face the need of expanding the vocabulary of the original model to describe images with novel concepts. A novel concept is a concept that is not included in the original dataset. The most straightforward way to address the task would be training the whole model from scratch with the data from both the original dataset and the new data. But it is too costly and slow. Sometimes, the dataset for training the original model is not available. In addition, we typically only have a few images with sentence descriptions for these novel concepts. If we combined this dataset with the original large-scale dataset, the new data might be overwhelmed. On the other hand, if we naively fine-tuned the model on the new data, the model might be overfitted to the new data and its ability to describe the previously learned concept will be harmed. To address these issues, we proposed a novel framework in [MWY15] that is fast, simple and effective. This framework avoids extensive retraining the model. The overfitting problem to the new data are also carefully controlled, see Chapter 3 for more details.

Although the results of image captioning are impressive and inspiring, there are still many issues to be addressed. From the method perspective, current image captioning models are still far from perfect. As many papers point out, the generated sentences are too similar to the database sentences and only a portion of the generated sentences are novel (“novel” here refers to be different from the sentences in the training set). Therefore, some researchers argue that current image captioning models are only doing a “smart” way of nearest neighbor search. From the task perspective, the image captioning task, similar to many tasks that need semantic evaluations, is notoriously hard to evaluate. There is disparity between the results of current automatic evaluation metrics and human’s judgment. We should, of course, develop better and more objective evaluation for image captioning.
But the real problem is the ambiguity underlies in the image captioning task itself. This task does not define, to what level, should we describe the details in the image. With so many valid ways to describe the image content, how do you determine whether a particular description is better than the other? Furthermore, the image captioning task only requires the model to output a generic image descriptions, instead of a description with specific functions. This limits its practical application. However, there is no way to interact with the image captioning models. We cannot input our specific requirements to the system, or inquiring the system the information we really needed.

To have a better and more objective definition of what is a good description of an image, we propose effective models and construct large scale benchmark dataset to address the task of generating and comprehending of unambiguous object descriptions [MHT16] (See Chapter 4 for details). An unambiguous object description, also known as a referring expression, is a description that uniquely describes the relevant object or region within its context in the image, such that a listener can recover the location of the original object. This task has a number of potential applications that use natural language interfaces, such as controlling a robot, e.g. you can ask your robot to fetch you “the white cup with tea”. In this task, we have an objective definition of what makes a good description: the unambiguity given its context (i.e. the whole frame image for object descriptions).

Another, maybe more explicit way to address the ambiguity in the image captioning task, is to input a natural language sentence question about the content of the image, and let the model answer it. This task is a direct form of the Turning test [Tur50]. It requires the model to understand both the content of the image and the question, incorporate them together, and generate a natural language answer. Because it is based on the content of the image, we call it as the Image Question Answering (IQA) task. Similar to the referring expression task mentioned above, this task is another natural way for us to communicate with the machines. There are certainly difference. For the referring expression task, we give the machine an instruction, such as ”give me the red cup on the desk”. For the IQA task, we ask the machine a question to get the information we need. For this IQA task, we propose a RNN-CNN based model, construct and release a large-scale multilingual (English and Chinese) image question answering dataset [GMZ15] (See Chapter 5 for details). To evaluate the effectiveness of the model, we conduct a Turning Test style evaluation: the Visual
Turning Test. In this test, we mix the answers generated by the computer with the groundtruth answers, and let a group of annotators to determine whether an answer is given by the computer or human. The results show that 64.7% of the computer generated answers are considered to be human generated, although the average quality of the answers is still much worse than the real groundtruth.

We can also adopt the RNN-CNN framework in the task of **multimodal word embedding learning** [MXJ16]. Word embeddings are vector representation of the words in the dictionary. For a good embedding model, words which are semantically or synthetically similar to each other will have small Euclidean distance in their word embedding vector space. Most of the previous word embedding work focuses on pure text word embedding. This is mainly because of the lack of large-scale multimodal training and evaluation dataset with both images and their sentence descriptions. In our paper of [MXJ16], we construct a large-scale dataset with 300 million sentences describing over 40 million images crawled and downloaded from publicly available Pins (i.e. an image with sentence descriptions uploaded by users) on Pinterest (See Chapter 6 for details). In addition, we construct an evaluation dataset to directly assess the effectiveness of word embeddings in terms of finding semantically similar or related words and phrases. The word/phrase pairs in this evaluation dataset are collected from the click data with millions of users in an image search system, thus contain rich semantic relationships. We also proposed and compared several multimodal word-embedding models.

In addition to the work mentioned above, the same RNN-CNN framework has been successfully applied to the task of multi-label classification, as shown in our paper of [WYM16]. The idea is simple but bold and novel. The multi-label classification task is similar to the task of image captioning in the way that there are variable number of labels for each image. If we define a specific order of these labels, we can treat this problem as a image to label sequence problem, which can be addressed using the RNN-CNN framework. In the experiments, we find that with the simple order of the frequency of the labels (more sophisticated orders do not help), we can achieve the state-of-the-art performance on these tasks.

Finally, we investigate the effectiveness of the attention module in neural image captioning models. As a recent development of the original RNN-CNN framework, [XBK15] proposed a neural image captioning model with attention mechanism. Their model learns a weighted sum of the image
features over the spatial grids when generating each word in a sentence. The grid with higher weight indicates that the model pays more attention to this grid. However, they only showed the qualitative results of their attention module with a few examples. In our work of [LMS17], we proposed a novel evaluation metric to quantitatively measure the correctness of this attention mechanism. This metric is based on the correlation of the machine attention with human’s attention. Based on this quantitative measurement, we find that although the original attention module performs better than a uniform attention baseline, there is still room for improvement. We then proposed a supervised attention model, which not only improves the attention quality, but also boosts the final image captioning performance. This method can be used in both the scenarios where the groundtruth attention annotations are available, and where only the weak annotations of object categories are available.

Since most of my work is among the earliest deep learning research effort for this field, there are lots of room for future improvement. E.g., for the task of image captioning, we need to have a better evaluation metric that is robust and is consistent with human’s judgment. In addition, current training of the RNN-CNN captioning framework uses the log-likelihood of sentence given the image as the training objective. It is reasonable, but does not directly related to the quality of the generated sentences.

In sum, I have investigated the following tasks in the field of language of vision during my PhD:

1. Image captioning [MXY14, MXY15a].
2. Describing images with novel concepts [MWY15].
3. Unambiguous object descriptions (Referring Expressions) [MHT16].
4. Image Question Answering [GMZ15].
5. Multimodal work embeddings [MXJ16].
6. Multi-label classification [WYM16].
7. Attention correctness in image captioning [LMS17].
I also did some research on texture classification by Active Patches [MZY14], and weakly supervised object segmentation [ZMY14]. The active patch model [MZY14] constructs a dictionary of active patches. Each patch in the dictionary can deform itself (e.g. rotation, translation, reshape) to match a local image region. It is a powerful image representation and we use it to achieve the state-of-the-art performance in the task of texture classification. In the weakly supervised object segmentation paper [ZMY14], we propose an expectation loss SVM (e-SVM) that can learn to segment object given only weak labels of object bounding box. They are all very interesting work, but this thesis will focus on the topic of multimodal learning and will describe our research on the first five tasks in details.

The thesis is organized as follows. In Chapter 2, we demonstrate the m-RNN model for image captioning and image-sentence retrievals, which is the basic form of the RNN-CNN framework. Then in Chapter 3, we introduce a novel framework of expanding the vocabulary of an existing model efficiently to describe images with novel concept. After that, we describe two new vision and language tasks, the referring expression task and the image question answering task in Chapter 4 and Chapter 5 respectively. Finally, we present our model and dataset of large-scale multimodal word embedding training and evaluation in Chapter 6.
CHAPTER 2

Deep Captioning with Multimodal Recurrent Neural Networks
(m-RNN) [MXY15a]

2.1 Abstract

In this chapter, we present a multimodal Recurrent Neural Network (m-RNN) model for generating novel image captions. It directly models the probability distribution of generating a word given previous words and an image. Image captions are generated according to this distribution. The model consists of two sub-networks: a deep recurrent neural network for sentences and a deep convolutional network for images. These two sub-networks interact with each other in a multimodal layer to form the whole m-RNN model. The effectiveness of our model is validated on four benchmark datasets (IAPR TC-12, Flickr 8K, Flickr 30K and MS COCO). Our model outperforms the state-of-the-art methods. In addition, we apply the m-RNN model to retrieval tasks for retrieving images or sentences, and achieves significant performance improvement over the state-of-the-art methods which directly optimize the ranking objective function for retrieval. The project page of this work is: www.stat.ucla.edu/~junhua.mao/m-RNN.html

2.2 Introduction

Obtaining sentence level descriptions for images is becoming an important task and it has many applications, such as early childhood education, image retrieval, and navigation for the blind. Thanks to the rapid development of computer vision and natural language processing technologies,}

---

1 Most recently, we adopt a simple strategy to boost the performance of image captioning task significantly. More details are shown in Section 2.9. The code and related data (e.g. refined image features and hypotheses sentences generated by the m-RNN model) are available at: https://github.com/mjhucla/mRNN-CR
recent work has made significant progress on this task (see a brief review in Section 2.3). Many previous methods treat it as a retrieval task. They learn a joint embedding to map the features of both sentences and images to the same semantic space. These methods generate image captions by retrieving them from a sentence database. Thus, they lack the ability of generating novel sentences or describing images that contain novel combinations of objects and scenes.

In this work, we propose a multimodal Recurrent Neural Networks (m-RNN) model to address both the task of generating novel sentences descriptions for images, and the task of image and sentence retrieval. The whole m-RNN model contains a language model part, a vision part and a multimodal part. The language model part learns a dense feature embedding for each word in the dictionary and stores the semantic temporal context in recurrent layers. The vision part contains a deep Convolutional Neural Network (CNN) which generates the image representation. The multimodal part connects the language model and the deep CNN together by a one-layer representation. Our m-RNN model is learned using a log-likelihood cost function (see details in Section 2.5). The errors can be backpropagated to the three parts of the m-RNN model to update the model parameters simultaneously.

In the experiments, we validate our model on four benchmark datasets: IAPR TC-12 ([GCM06]), Flickr 8K ([RYH10]), Flickr 30K ([YLH14a]) and MS COCO ([LMB14]). We show that our method achieves state-of-the-art performance, significantly outperforming all the other methods for the three tasks: generating novel sentences, retrieving images given a sentence and retrieving sentences given an image. Our framework is general and can be further improved by incorporating more powerful deep representations for images and sentences.

2.3 Related Work

Deep model for computer vision and natural language. The methods based on the deep neural network developed rapidly in recent years in both the field of computer vision and natural language.

---

2 A previous version of this work appears in the NIPS 2014 Deep Learning Workshop with the title “Explain Images with Multimodal Recurrent Neural Networks” [http://arxiv.org/abs/1410.1090] ([MXY14]). We observed subsequent arXiv papers which also use recurrent neural networks in this topic and cite our work. We gratefully acknowledge them.
<table>
<thead>
<tr>
<th>Ret.</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Top view of the lights of a city at night, with a well-illuminated square in front of a church in the foreground; 2. People on the stairs in front of an illuminated cathedral with two towers at night;</td>
<td>A square with burning street lamps and a street in the foreground;</td>
</tr>
<tr>
<td>1. Tourists are sitting at a long table with beer bottles on it in a rather dark restaurant and are raising their bierglaeser; 2. Tourists are sitting at a long table with a white table-cloth in a somewhat dark restaurant;</td>
<td>Tourists are sitting at a long table with a white table cloth and are eating;</td>
</tr>
<tr>
<td>1. A dry landscape with light brown grass and green bushes and trees in the foreground and large reddish-brown rocks and a blue sky in the background; 2. A few bushes at the bottom and a clear sky in the background;</td>
<td>A dry landscape with green trees and bushes and light brown grass in the foreground and reddish-brown round rock domes and a blue sky in the background;</td>
</tr>
<tr>
<td>1. Group picture of nine tourists and one local on a grey rock with a lake in the background; 2. Five people are standing and four are squatting on a brown rock in the foreground;</td>
<td>A blue sky in the background;</td>
</tr>
</tbody>
</table>

Figure 2.1: Examples of the generated and two top-ranked retrieved sentences given the query image from IAPR TC-12 dataset. The sentences can well describe the content of the images. We show a failure case in the fourth image, where the model mistakenly treats the lake as the sky and misses all the people. More examples from the MS COCO dataset can be found on the project page: [www.stat.ucla.edu/~junhua.mao/m-RNN.html](http://www.stat.ucla.edu/~junhua.mao/m-RNN.html)

For computer vision, [KSH12] propose a deep Convolutional Neural Networks (CNN) with 8 layers (denoted as AlexNet) and outperform previous methods by a large margin in the image classification task of ImageNet challenge ([RDS14]). This network structure is widely used in computer vision, e.g. [GDD14] design a object detection framework (RCNN) based on this work. Recently, [SZ14] propose a CNN with over 16 layers (denoted as VggNet) and performs substantially better than the AlexNet. For natural language, the Recurrent Neural Network (RNN) shows the state-of-the-art performance in many tasks, such as speech recognition and word embedding learning ([MKB10], [MKB11], [MSC13]). Recently, RNNs have been successfully applied to machine translation to extract semantic information from the source sentence and generate target sentences (e.g. [KB13], [CMG14] and [SVL14]).

**Image-sentence retrieval.** Many previous methods treat the task of describing images as a retrieval task and formulate the problem as a ranking or embedding learning problem ([HYH13], [FCS13], [SLM14]). They first extract the word and sentence features (e.g. [SLM14] uses dependency tree Recursive Neural Network to extract sentence features) as well as the image features. Then they optimize a ranking cost to learn an embedding model that maps both the sentence feature and
the image feature to a common semantic feature space. In this way, they can directly calculate the
distance between images and sentences. Recently, [KJF14] show that object level image features
based on object detection results can generate better results than image features extracted at the
global level.

Generating novel sentence descriptions for images. There are generally three categories of
methods for this task. The first category assumes a specific rule of the language grammar. They
parse the sentence and divide it into several parts ([MHD12, GM12]). Each part is associated with
an object or an attribute in the image (e.g. [KPD11] uses a Conditional Random Field model and
[FHS10] uses a Markov Random Field model). This kind of method generates sentences that are
syntactically correct. The second category retrieves similar captioned images, and generates new
descriptions by generalizing and re-composing the retrieved captions ([KOB14]). The third category
of methods, which is more related to our method, learns a probability density over the space of
multimodal inputs (i.e. sentences and images), using for example, Deep Boltzmann Machines
([SS12]), and topic models ([BDF03, JSD11]). They generate sentences with richer and more
flexible structure than the first group. The probability of generating sentences using the model
can serve as the affinity metric for retrieval. Our method falls into this category. More closely
related to our tasks and method is the work of [KZS14], which is built on a Log-BiLinear model
([MH07]) and use AlexNet to extract visual features. It needs a fixed length of context (i.e. five
words), whereas in our model, the temporal context is stored in a recurrent architecture, which
allows arbitrary context length.

Shortly after [MXY14], several papers appear with record breaking results (e.g. [KSZ14,
KF14, VTB14, DHG14, FGI14, CZ14]). Many of them are built on recurrent neural networks. It
demonstrates the effectiveness of storing context information in a recurrent layer. Our work has
two major difference from these methods. Firstly, we incorporate a two-layer word embedding
system in the m-RNN network structure which learns the word representation more efficiently than
the single-layer word embedding. Secondly, we do not use the recurrent layer to store the visual
information. The image representation is inputted to the m-RNN model along with every word in
the sentence description. It utilizes of the capacity of the recurrent layer more efficiently, and allows
us to achieve state-of-the-art performance using a relatively small dimensional recurrent layer. In
Figure 2.2: Illustration of the simple Recurrent Neural Network (RNN) and our multimodal Recurrent Neural Network (m-RNN) architecture. (a). The simple RNN. (b). Our m-RNN model. The inputs of our model are an image and its corresponding sentence descriptions. \( w_1, w_2, \ldots, w_L \) represents the words in a sentence. We add a start sign \( w_{\text{start}} \) and an end sign \( w_{\text{end}} \) to all the training sentences. The model estimates the probability distribution of the next word given previous words and the image. It consists of five layers (i.e. two word embedding layers, a recurrent layer, a multimodal layer and a softmax layer) and a deep CNN in each time frame. The number above each layer indicates the dimension of the layer. The weights are shared among all the time frames. (Best viewed in color)

In the experiments, we show that these two strategies lead to better performance. Our method is still the best-performing approach for almost all the evaluation metrics.

2.4 Model Architecture

2.4.1 Simple recurrent neural network

We briefly introduce the simple Recurrent Neural Network (RNN) or Elman network (Elm90). Its architecture is shown in Figure 2.2(a). It has three types of layers in each time frame: the input word layer \( w \), the recurrent layer \( r \) and the output layer \( y \). The activation of input, recurrent and output layers at time \( t \) is denoted as \( w(t), r(t), \) and \( y(t) \) respectively. \( w(t) \) denotes the current word
vector, which can be a simple 1-of-N coding representation \( h(t) \) (i.e. the one-hot representation, which is binary and has the same dimension as the vocabulary size with only one non-zero element) \cite{MKB10}. \( y(t) \) can be calculated as follows:

\[
x(t) = [w(t) \, r(t-1)]; \quad r(t) = f_1(U \cdot x(t)); \quad y(t) = g_1(V \cdot r(t));
\]

(2.1)

where \( x(t) \) is a vector that concatenates \( w(t) \) and \( r(t-1) \), \( f_1(\cdot) \) and \( g_1(\cdot) \) are element-wise sigmoid and softmax function respectively, and \( U, V \) are weights which will be learned.

The size of the RNN is adaptive to the length of the input sequence. The recurrent layers connect the sub-networks in different time frames. Accordingly, when we do backpropagation, we need to propagate the error through recurrent connections back in time (\cite{RHW88}).

2.4.2 Our m-RNN model

The structure of our multimodal Recurrent Neural Network (m-RNN) is shown in Figure 2.2(b). It has five layers in each time frame: two word embedding layers, the recurrent layer, the multimodal layer, and the softmax layer).

The two word embedding layers embed the one-hot input into a dense word representation. It encodes both the syntactic and semantic meaning of the words. The semantically relevant words can be found by calculating the Euclidean distance between two dense word vectors in embedding layers. Most of the sentence-image multimodal models (\cite{KIF14, FCST13, SLM14, KZS14}) use pre-computed word embedding vectors as the initialization of their model. In contrast, we randomly initialize our word embedding layers and learn them from the training data. We show that this random initialization is sufficient for our architecture to generate the state-of-the-art result. We treat the activation of the word embedding layer II (see Figure 2.2(b)) as the final word representation, which is one of the three direct inputs of the multimodal layer.

After the two word embedding layers, we have a recurrent layer with 256 dimensions. The calculation of the recurrent layer is slightly different from the calculation for the simple RNN. Instead of concatenating the word representation at time \( t \) (denoted as \( w(t) \)) and the recurrent layer activation at time \( t-1 \) (denoted as \( r(t-1) \)), we first map \( r(t-1) \) into the same vector space as
\( w(t) \) and add them together:

\[
\mathbf{r}(t) = f_2(\mathbf{U}_r \cdot \mathbf{r}(t-1) + \mathbf{w}(t));
\]  

(2.2)

where “+” represents element-wise addition. We set \( f_2(.) \) to be the Rectified Linear Unit (ReLU), inspired by its recent success when training very deep structure in computer vision field ([NH10, KSH12]). This differs from the simple RNN where the sigmoid function is adopted (see Section 2.4.1). ReLU is faster, and harder to saturate or overfit the data than non-linear functions like the sigmoid. When the backpropagation through time (BPTT) is conducted for the RNN with sigmoid function, the vanishing or exploding gradient problem appears since even the simplest RNN model can have a large temporal depth \(^3\) Previous work ([MKB10, MKB11]) use heuristics, such as the truncated BPTT, to avoid this problem. The truncated BPTT stops the BPTT after \( k \) time steps, where \( k \) is a hand-defined hyperparameter. Because of the good properties of ReLU, we do not need to stop the BPTT at an early stage, which leads to better and more efficient utilization of the data than the truncated BPTT.

After the recurrent layer, we set up a 512 dimensional multimodal layer that connects the language model part and the vision part of the m-RNN model (see Figure 2.2(b)). This layer has three inputs: the word-embedding layer II, the recurrent layer and the image representation. For the image representation, here we use the activation of the 7\(^{th}\) layer of AlexNet ([KSH12]) or 15\(^{th}\) layer of VggNet ([SZ14]), though our framework can use any image features. We map the activation of the three layers to the same multimodal feature space and add them together to obtain the activation of the multimodal layer:

\[
\mathbf{m}(t) = g_2(\mathbf{V}_w \cdot \mathbf{w}(t) + \mathbf{V}_r \cdot \mathbf{r}(t) + \mathbf{V}_I \cdot \mathbf{I});
\]  

(2.3)

where “+” denotes element-wise addition, \( \mathbf{m} \) denotes the multimodal layer feature vector, \( \mathbf{I} \) denotes the image feature. \( g_2(.) \) is the element-wise scaled hyperbolic tangent function ([LBO12]):

\[
g_2(x) = 1.7159 \cdot \tanh\left(\frac{2}{3} x\right)
\]  

(2.4)

This function forces the gradients into the most non-linear value range and leads to a faster training process than the basic hyperbolic tangent function.

\(^3\)We tried Sigmoid and Scaled Hyperbolic Tangent function as the non-linear functions for RNN in the experiments but they lead to the gradient explosion problem easily.
Both the simple RNN and m-RNN models have a softmax layer that generates the probability distribution of the next word. The dimension of this layer is the vocabulary size $M$, which is different for different datasets.

### 2.5 Training the m-RNN

To train our m-RNN model we adopt a log-likelihood cost function. It is related to the Perplexity of the sentences in the training set given their corresponding images. Perplexity is a standard measure for evaluating language model. The perplexity for one word sequence (i.e. a sentence) $w_{1:L}$ is calculated as follows:

$$\log_2 \text{PPL}(w_{1:L}|I) = -\frac{1}{L} \sum_{n=1}^{L} \log_2 P(w_n|w_{1:n-1}, I)$$ (2.5)

where $L$ is the length of the word sequence, $\text{PPL}(w_{1:L}|I)$ denotes the perplexity of the sentence $w_{1:L}$ given the image $I$. $P(w_n|w_{1:n-1}, I)$ is the probability of generating the word $w_n$ given $I$ and previous words $w_{1:n-1}$. It corresponds to the activation of the SoftMax layer of our model.

The cost function of our model is the average log-likelihood of the words given their context words and corresponding images in the training sentences plus a regularization term. It can be calculated by the perplexity:

$$C = \frac{1}{N} \sum_{i=1}^{N_s} L_i \cdot \log_2 \text{PPL}(w_{1:L_i}^{(i)}|I^{(i)}) + \lambda_\theta \cdot \|\theta\|^2_2$$ (2.6)

where $N_s$ and $N$ denotes the number of sentences and the number of words in the training set receptively, $L_i$ denotes the length of $i^{th}$ sentences, and $\theta$ represents the model parameters.

Our training objective is to minimize this cost function, which is equivalent to maximize the probability of generating the sentences in the training set using the model. The cost function is differentiable and we use backpropagation to learn the model parameters.
2.6 Sentence Generation, Image Retrieval and Sentence Retrieval

We use the trained m-RNN model for three tasks: 1) Sentences generation, 2) Image retrieval (retrieving most relevant images to the given sentence), 3) Sentence retrieval (retrieving most relevant sentences to the given image).

The sentence generation process is straightforward. Starting from the start sign \(w_{\text{start}}\) or arbitrary number of reference words (e.g. we can input the first K words in the reference sentence to the model and then start to generate new words), our model can calculate the probability distribution of the next word: 
\[
P(w_n|w_{1:n-1}, I)\]
Then we can sample from this probability distribution to pick the next word. In practice, we find that selecting the word with the maximum probability performs slightly better than sampling. After that, we input the picked word to the model and continue the process until the model outputs the end sign \(w_{\text{end}}\).

For the retrieval tasks, we use our model to calculate the probability of generating a sentence \(w_{1:L}\) given an image \(I\): 
\[
P(w_{1:L}|I) = \prod_n P(w_n|w_{1:n-1}, I)\]
The probability can be treated as an affinity measurement between sentences and images.

For the image retrieval task, given the query sentence \(w_{Q1:L}\), we rank the dataset images \(I^D\) according to the probability 
\[
P(w_{Q1:L}|I^D)\]
and retrieved the top ranked images. This is equivalent to the perplexity-based image retrieval in [KZS14].

The sentence retrieval task is trickier because there might be some sentences that have high probability or perplexity for any image query (e.g. sentences consist of many frequently appeared words). To solve this problem, [KZS14] uses the perplexity of a sentence conditioned on the averaged image feature across the training set as the reference perplexity to normalize the original perplexity. Different from them, we use the normalized probability where the normalization factor is the marginal probability of \(w_{1:L}^D\):

\[
P(w_{1:L}^D|I^Q)/P(w_{1:L}^D)\quad P(w_{1:L}^D) = \sum_{I'} P(w_{1:L}^D|I') \cdot P(I')
\] (2.7)
where \(w_{1:L}^D\) denotes the sentence in the dataset, \(I^Q\) denotes the query image, and \(I'\) are images sampled from the training set. We approximate \(P(I')\) by a constant and ignore this term. This strategy leads to a much better performance than that in [KZS14] in the experiments. The normalized
probability is equivalent to the probability $P(I^Q|w^D_{1:L})$, which is symmetric to the probability $P(w^Q_{1:L}|I^D)$ used in the image retrieval task.

### 2.7 Learning of Sentence and Image Features

The architecture of our model allows the gradients from the loss function to be backpropagated to both the language modeling part (i.e. the word embedding layers and the recurrent layer) and the vision part (e.g. the AlexNet or VggNet).

For the language part, as mentioned above, we randomly initialize the language modeling layers and learn their parameters. For the vision part, we use the pre-trained AlexNet ([KSH12]) or the VggNet ([SZ14]) on ImageNet dataset ([RDS14]). Recently, [KJF14] show that using the RCNN object detection results ([GDD14]) combined with the AlexNet features performs better than simply treating the image as a whole frame. In the experiments, we show that our method performs much better than [KJF14] when the same image features are used, and is better than or comparable to their results even when they use more sophisticated features based on object detection.

We can update the CNN in the vision part of our model according to the gradient backpropagated from the multimodal layer. In this chapter, we fix the image features and the deep CNN network in the training stage due to a shortage of data. In future work, we will apply our method on large datasets (e.g. the complete MS COCO dataset, which has not yet been released) and finetune the parameters of the deep CNN network in the training stage.

The m-RNN model is trained using Baidu’s internal deep learning platform PADDLE, which allows us to explore many different model architectures in a short period. The hyperparameters, such as layer dimensions and the choice of the non-linear activation functions, are tuned via cross-validation on Flickr8K dataset and are then fixed across all the experiments. It takes 25 ms on average to generate a sentence (excluding image feature extraction stage) on a single core CPU.
2.8 Experiments

2.8.1 Datasets

We test our method on four benchmark datasets with sentence level annotations: IAPR TC-12 ([GCM06]), Flickr 8K ([RYH10]), Flickr 30K ([YLH14a]) and MS COCO ([LMB14]).

**IAPR TC-12.** This dataset consists of around 20,000 images taken from different locations around the world. It contains images of different sports and actions, people, animals, cities, landscapes, etc. For each image, it provides at least one sentence annotation. On average, there are about 1.7 sentence annotations for one image. We adopt the standard separation of training and testing set as previous works ([GVS10, KZS14]) with 17,665 images for training and 1962 images for testing.

**Flickr8K.** This dataset consists of 8,000 images extracted from Flickr. For each image, it provides five sentence annotations. We adopt the standard separation of training, validation and testing set provided by the dataset. There are 6,000 images for training, 1,000 images for validation and 1,000 images for testing.

**Flickr30K.** This dataset is a recent extension of Flickr8K. For each image, it also provides five sentences annotations. It consists of 158,915 crowd-sourced captions describing 31,783 images. The grammar and style for the annotations of this dataset is similar to Flickr8K. We follow the previous work ([KJF14]) which used 1,000 images for testing. This dataset, as well as the Flickr8K dataset, were originally used for the image-sentence retrieval tasks.

**MS COCO.** The current release of this recently proposed dataset contains 82,783 training images and 40,504 validation images. For each image, it provides five sentences annotations. We randomly sampled 4,000 images for validation and 1,000 images for testing from their currently released validation set. The dataset partition of MS COCO and Flickr30K is available in the project page[^1].

[^1]: [www.stat.ucla.edu/~junhua.mao/m-RNN.html](http://www.stat.ucla.edu/~junhua.mao/m-RNN.html)
2.8.2 Evaluation metrics

**Sentence Generation.** Following previous works, we use the sentence perplexity (see Equ. 2.5) and BLEU scores (i.e. B-1, B-2, B-3, and B-4) ([PRW02]) as the evaluation metrics. BLEU scores were originally designed for automatic machine translation where they rate the quality of a translated sentences given several reference sentences. Similarly, we can treat the sentence generation task as the “translation” of the content of images to sentences. BLEU remains the standard evaluation metric for sentence generation methods for images, though it has drawbacks. For some images, the reference sentences might not contain all the possible descriptions in the image and BLEU might penalize some correctly generated sentences.

**Sentence Retrieval and Image Retrieval.** We adopt the same evaluation metrics as previous works ([SLM14] [FCS13] [KJF14]) for both the tasks of sentences retrieval and image retrieval. We use R@K (K = 1, 5, 10) as the measurement. R@K is the recall rate of a correctly retrieved groundtruth given top K candidates. Higher R@K usually means better retrieval performance. Since we care most about the top-ranked retrieved results, the R@K scores with smaller K are more important.

The Med r is another metric we use, which is the median rank of the first retrieved groundtruth sentence or image. Lower Med r usually means better performance. For IAPR TC-12 datasets, we use additional evaluation metrics to conduct a fair comparison with previous work ([KZS14]).

2.8.3 Results on IAPR TC-12

The results of the sentence generation task are shown in Table 2.1. Ours-RNN-Base serves as a baseline method for our m-RNN model. It has the same architecture as m-RNN except that it does not have the image representation input.

To conduct a fair comparison, we follow the same experimental settings of [KZS14] to calculate the BLEU scores and perplexity. These two evaluation metrics are not necessarily correlated to each other for the following reasons. As mentioned in Section 2.5, perplexity is calculated according to the conditional probability of the word in a sentence given all of its previous reference words.

---

5 [KZS14] further improved their results after the publication. We compare our results with their updated ones here.
Table 2.1: Results of the sentence generation task on the IAPR TC-12 dataset. “B” is short for BLEU.

<table>
<thead>
<tr>
<th>Method</th>
<th>PPL</th>
<th>B-1</th>
<th>B-2</th>
<th>B-3</th>
<th>B-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBL, [MH07]</td>
<td>9.29</td>
<td>0.321</td>
<td>0.145</td>
<td>0.064</td>
<td>-</td>
</tr>
<tr>
<td>MLBLB-AlexNet, [KZS14]</td>
<td>9.86</td>
<td>0.393</td>
<td>0.211</td>
<td>0.112</td>
<td>-</td>
</tr>
<tr>
<td>MLBLF-AlexNet, [KZS14]</td>
<td>9.90</td>
<td>0.387</td>
<td>0.209</td>
<td>0.115</td>
<td>-</td>
</tr>
<tr>
<td>[GVJ12]</td>
<td></td>
<td>0.15</td>
<td>0.06</td>
<td>0.01</td>
<td>-</td>
</tr>
<tr>
<td>[GM12]</td>
<td></td>
<td>0.33</td>
<td>0.18</td>
<td>0.07</td>
<td>-</td>
</tr>
<tr>
<td>Ours-RNN-Base</td>
<td>7.77</td>
<td>0.307</td>
<td>0.177</td>
<td>0.096</td>
<td>0.043</td>
</tr>
<tr>
<td>Ours-m-RNN-AlexNet</td>
<td>6.92</td>
<td>0.482</td>
<td>0.357</td>
<td>0.269</td>
<td>0.208</td>
</tr>
</tbody>
</table>

Table 2.2: R@K and median rank (Med r) for IAPR TC-12 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Sentence Retrieval (Image to Text)</th>
<th>Image Retrieval (Text to Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td>Ours-m-RNN</td>
<td>20.9</td>
<td>43.8</td>
</tr>
</tbody>
</table>

Therefore, a strong language model that successfully captures the distributions of words in sentences can have a low perplexity without the image content. But the content of the generated sentences might be uncorrelated to images. From Table 2.1, we can see that although our baseline method of RNN generates a low perplexity, its BLEU score is low, indicating that it fails to generate sentences that are consistent with the content of images.

Table 2.1 shows that our m-RNN model performs much better than our baseline RNN model and the state-of-the-art methods both in terms of the perplexity and BLEU score.

For the retrieval tasks, since there are no publicly available results of R@K and Med r in this dataset, we report R@K scores of our method in Table 2.2 for future comparisons. The result shows that 20.9% top-ranked retrieved sentences and 13.2% top-ranked retrieved images are groundtruth.
<table>
<thead>
<tr>
<th></th>
<th>Sentence Retrieval (Image to Text)</th>
<th>Image Retrieval (Text to Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td>Random</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>SDT-RNN-AlexNet</td>
<td>4.5</td>
<td>18.0</td>
</tr>
<tr>
<td>Socher-avg-RCNN</td>
<td>6.0</td>
<td>22.7</td>
</tr>
<tr>
<td>DeViSE-avg-RCNN</td>
<td>4.8</td>
<td>16.5</td>
</tr>
<tr>
<td>DeepFE-AlexNet</td>
<td>5.9</td>
<td>19.2</td>
</tr>
<tr>
<td>DeepFE-RCNN</td>
<td>12.6</td>
<td>32.9</td>
</tr>
<tr>
<td>Ours-m-RNN-AlexNet</td>
<td><strong>14.5</strong></td>
<td><strong>37.2</strong></td>
</tr>
</tbody>
</table>

Table 2.3: Results of R@K and median rank (Med r) for Flickr8K dataset. “-AlexNet” denotes the image representation based on AlexNet extracted from the whole image frame. “-RCNN” denotes the image representation extracted from possible objects detected by the RCNN algorithm.

### 2.8.4 Results on Flickr8K

This dataset was widely used as a benchmark dataset for image and sentence retrieval. The R@K and Med r of different methods are shown in Table 2.3. We compare our model with several state-of-the-art methods: SDT-RNN ([SLM14]), DeViSE ([FCST13]), DeepFE ([KJF14]) with various image representations. Our model outperforms these methods by a large margin when using the same image representation (e.g. AlexNet). We also list the performance of methods using more sophisticated features in Table 2.3. “-avg-RCNN” denotes methods with features of the average CNN activation of all objects above a detection confidence threshold. DeepFE-RCNN [KJF14] uses a fragment mapping strategy to better exploit the object detection results. The results show that using these features improves the performance. Even without the help from the object detection methods, however, our method performs better than these methods in almost all the evaluation metrics. We will develop our framework using better image features based on object detection in the future work.

The \( PPL, B-1, B-2, B-3 \) and B-4 of the generated sentences using our m-RNN-AlexNet model in this dataset are 24.39, 0.565, 0.386, 0.256, and 0.170 respectively.
### 2.8.5 Results on Flickr30K and MS COCO

<table>
<thead>
<tr>
<th></th>
<th>Sentence Retrieval (Image to Text)</th>
<th>Image Retrieval (Text to Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td><strong>Flickr30K</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>DeViSE-avg-RCNN</td>
<td>4.8</td>
<td>16.5</td>
</tr>
<tr>
<td>DeepFE-RCNN</td>
<td>16.4</td>
<td>40.2</td>
</tr>
<tr>
<td>RVR</td>
<td>12.1</td>
<td>27.8</td>
</tr>
<tr>
<td>MNLM-AlexNet</td>
<td>14.8</td>
<td>39.2</td>
</tr>
<tr>
<td>MNLM-VggNet</td>
<td>23.0</td>
<td>50.7</td>
</tr>
<tr>
<td>NIC</td>
<td>17.0</td>
<td>56.0</td>
</tr>
<tr>
<td>LRCN</td>
<td>14.0</td>
<td>34.9</td>
</tr>
<tr>
<td>DeepVS</td>
<td>22.2</td>
<td>48.2</td>
</tr>
<tr>
<td>Ours-m-RNN-AlexNet</td>
<td>18.4</td>
<td>40.2</td>
</tr>
<tr>
<td>Ours-m-RNN-VggNet</td>
<td><strong>35.4</strong></td>
<td><strong>63.8</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sentence Retrieval (Image to Text)</th>
<th>Image Retrieval (Text to Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td><strong>MS COCO</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>DeepVS-RCNN</td>
<td>29.4</td>
<td>62.0</td>
</tr>
<tr>
<td>Ours-m-RNN-VggNet</td>
<td><strong>41.0</strong></td>
<td><strong>73.0</strong></td>
</tr>
</tbody>
</table>

Table 2.4: Results of R@K and median rank (Med r) for Flickr30K dataset and MS COCO dataset.

We compare our method with several state-of-the-art methods in these two recently released dataset (Note that the last six methods appear very recently, we use the results reported in their papers): DeViSE ([FCS13]), DeepFE ([KF14]), MNLM ([KSZ14]), DMSM ([FGI14]), NIC ([VTB14]), LRCN ([DHG14]), RVR ([CZ14]), and DeepVS ([KF14]). The results of the retrieval tasks and the sentence generation task are shown in Table 2.4 and Table 2.5 respectively. We also

---

6We only select the word with maximum probability each time in the sentence generation process in Table 2.5 while many comparing methods (e.g. DMSM, NIC, LRCN) uses a beam search scheme that keeps the best K candidates. The beam search scheme will lead to better performance in practice using the same model.
Table 2.5: Results of generated sentences on the Flickr 30K dataset and MS COCO dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Flickr30K PPL</th>
<th>B-1</th>
<th>B-2</th>
<th>B-3</th>
<th>B-4</th>
<th>MS COCO PPL</th>
<th>B-1</th>
<th>B-2</th>
<th>B-3</th>
<th>B-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>RVR</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.13</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.19</td>
</tr>
<tr>
<td>DeepVS-AlexNet</td>
<td>-</td>
<td>0.47</td>
<td>0.21</td>
<td>0.09</td>
<td>-</td>
<td>-</td>
<td>0.53</td>
<td>0.28</td>
<td>0.15</td>
<td>-</td>
</tr>
<tr>
<td>DeepVS-VggNet</td>
<td>21.20</td>
<td>0.50</td>
<td>0.30</td>
<td>0.15</td>
<td>-</td>
<td>19.64</td>
<td>0.57</td>
<td>0.37</td>
<td>0.19</td>
<td>-</td>
</tr>
<tr>
<td>NIC</td>
<td>-</td>
<td>0.66</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.67</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LRCN</td>
<td>-</td>
<td>0.59</td>
<td>0.39</td>
<td>0.25</td>
<td>0.16</td>
<td>-</td>
<td>0.63</td>
<td>0.44</td>
<td>0.31</td>
<td>0.21</td>
</tr>
<tr>
<td>DMSM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.21</td>
</tr>
<tr>
<td>Ours-m-RNN-AlexNet</td>
<td>35.11</td>
<td>0.54</td>
<td>0.36</td>
<td>0.23</td>
<td>0.15</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Ours-m-RNN-VggNet</td>
<td><strong>20.72</strong></td>
<td><strong>0.60</strong></td>
<td><strong>0.41</strong></td>
<td><strong>0.28</strong></td>
<td><strong>0.19</strong></td>
<td><strong>13.60</strong></td>
<td><strong>0.67</strong></td>
<td><strong>0.49</strong></td>
<td><strong>0.35</strong></td>
<td><strong>0.25</strong></td>
</tr>
</tbody>
</table>

Table 2.6: Properties of the recurrent layers for the five very recent methods. LRCN has a stack of four 1000 dimensional LSTM layers. We achieves state-of-the-art performance using a relatively small dimensional recurrent layer. LSTM ([HS97]) can be treated as a sophisticated version of the RNN.

<table>
<thead>
<tr>
<th>Method</th>
<th>Our m-RNN</th>
<th>MNLN</th>
<th>NIC</th>
<th>LRCN</th>
<th>RVR</th>
<th>DeepVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN Dim.</td>
<td>256</td>
<td>300</td>
<td>512</td>
<td>1000 (\times4)</td>
<td>100</td>
<td>300-600</td>
</tr>
<tr>
<td>LSTM</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Our method with VggNet image representation ([SZ14]) outperforms the state-of-the-art methods, including the very recently released methods, in almost all the evaluation metrics. Note that the dimension of the recurrent layer of our model is relatively small compared to the competing methods. It shows the advantage and efficiency of our method that directly inputs the visual information to the multimodal layer instead of storing it in the recurrent layer. The m-RNN model with VggNet performs better than that with AlexNet, which indicates the importance of strong image...
representations in this task. 71% of the generated sentences for MS COCO datasets are novel (i.e. different from training sentences).

We also validate our method on the test set of MS COCO by their evaluation server ([CFL15]). The results are shown in Table 2.7. We evaluate our model with greedy inference (select the word with the maximum probability each time) as well as with the beam search inference. “-c5” represents results using 5 reference sentences and “-c40” represents results using 40 reference sentences.

To further validate the importance of different components of the m-RNN model, we train several variants of the original m-RNN model and compare their performance. In particular, we show that the two-layer word embedding system outperforms the single-layer version and the strategy of directly inputting the visual information to the multimodal layer substantially improves the performance (about 5% for B-1).

<table>
<thead>
<tr>
<th></th>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
<th>CIDEr</th>
<th>ROUGE_L</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>m-RNN-greedy-c5</td>
<td>0.668</td>
<td>0.488</td>
<td>0.342</td>
<td>0.239</td>
<td>0.729</td>
<td>0.489</td>
<td>0.221</td>
</tr>
<tr>
<td>m-RNN-greedy-c40</td>
<td>0.845</td>
<td>0.730</td>
<td>0.598</td>
<td>0.473</td>
<td>0.740</td>
<td>0.616</td>
<td>0.291</td>
</tr>
<tr>
<td>m-RNN-beam-c5</td>
<td>0.680</td>
<td>0.506</td>
<td>0.369</td>
<td>0.272</td>
<td>0.791</td>
<td>0.499</td>
<td>0.225</td>
</tr>
<tr>
<td>m-RNN-beam-c40</td>
<td>0.865</td>
<td>0.760</td>
<td>0.641</td>
<td>0.529</td>
<td>0.789</td>
<td>0.640</td>
<td>0.304</td>
</tr>
</tbody>
</table>

Table 2.7: Results of the MS COCO test set evaluated by MS COCO evaluation server

2.9 Nearest Neighbor as Reference

Recently, [DGG15] proposed a nearest neighbor approach that retrieves the captions of the $k$ nearest images in the training set, ranks these captions according to the consensus of the caption w.r.t. to the rest of the captions, and output the top ranked one.

Inspired by this method, we first adopt the m-RNN model with the transposed weight sharing strategy ([MWY15], denoted as m-RNN-shared) to generate $n$ hypotheses using a beam search scheme. Specifically, we keep the $n$ best candidates in the sentence generation process until the model generates the end sign $w_{end}$. These $n$ best candidates are approximately the $n$ most probable
Compared to the original VggNet features, the features refined by the m-RNN model are better for capturing richer and more accurate visual information.

After generating the hypotheses of a target image, we retrieve its nearest neighbors in the image feature space on the training set (see details in Section 2.9.1). Then we calculate the “consensus” scores ([DCF15]) of the hypotheses w.r.t. to the groundtruth captions of the nearest neighbor images, and rerank the hypotheses according to these scores (see details in Section 2.9.2).

2.9.1 Image features for the nearest neighbor image search

We try two types of image features for the nearest neighbor image search. The first one is the original image features extracted by the VggNet ([SZ14]). We first resize the image so that its short side is 256 pixels. Then we extract features on ten $224 \times 224$ windows (the four corners, the center and their mirrored versions) on the resized image. Finally, we average pool the ten features to make it a 4,096 dimensional feature.

\[ \text{If we directly output the top hypotheses generated by the model, then } n = 5 \text{ gives us the best performance. But if we want to rerank the hypotheses, then } n = 10 \text{ gives us a better result on the validation set.} \]
The second type is the feature refined by our m-RNN model. It can be calculated as: \( I' = g_2(V_I \cdot I) \), where \( V_I \) is the weight matrix between the image representation and the multimodal layer (see Equation 2.3), and \( g_2(\cdot) \) is the scaled hyperbolic tangent function.

We show the sample images and their nearest neighbors in Figure 2.3. We find that compared to the original VggNet features, the features refined by the m-RNN model capture richer and more accurate visual information. E.g., the target image in the second row contains an old woman with a bunch of bananas. The original VggNet features do not retrieve images with bananas in them.

<table>
<thead>
<tr>
<th>MS COCO val for consensus reranking</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>m-RNN-shared</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-BLEU</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-CIDEr</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-BLEU-Orcale</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-CIDEr-Orcale</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MS COCO 2014 test server</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>m-RNN-shared</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-BLEU</td>
</tr>
<tr>
<td>m-RNN-shared-NNref-CIDEr</td>
</tr>
</tbody>
</table>

Table 2.8: Results of m-RNN-shared model after applying consensus reranking using nearest neighbors as references (m-RNN-shared-NNref), compared with those of the original m-RNN model on our validation set and MS COCO test server.
2.9.2 Consensus Reranking

Suppose we have get the $k$ nearest neighbor images in the training set as the reference. We follow \cite{DCF15} to calculate the consensus score of a hypotheses. The difference is that \cite{DCF15} treat the captions of the $k$ nearest neighbor images as the hypotheses while our hypotheses are generated by the m-RNN model. More specifically, for each hypothesis, we calculate the mean similarity between this hypothesis and all the captions of the $k$ nearest neighbor images. The consensus score of this hypothesis is the mean similarity score of the $m$ nearest captions. The similarity between a hypothesis and one of its nearest neighbor reference captions is defined by a sentence-level BLEU score \cite{PRW02} or a sentence-level CIDEr \cite{VZP14}. We cross-validate the hyperparameters $k$ and $m$. For the BLEU-based similarity, the optimal $k$ and $m$ are 60 and 175 respectively. For the CIDEr-based similarity, the optimal $k$ and $m$ are 60 and 125 respectively.

2.9.3 Experiments

We show the results of our model on our validation set and the MS COCO testing server in Table \ref{table:results}. For BLEU-based consensus reranking, we get an improvement of 3.5 points on our validation set and 3.3 points on the MS COCO test 2014 set in terms of BLEU4 score. For the CIDEr-based consensus reranking, we get an improvement of 9.4 points on our validation set and 9.8 points on the MS COCO test 2014 set in terms of CIDEr.

2.9.4 Discussion

We show the rank of the ten hypotheses before and after reranking in Figure \ref{fig:reranking}. Although the hypotheses are similar to each other, there are some variances among them (e.g., some of them capture more details of the images. Some of them might be partially wrong). The reranking process is able to improve the rank of good captions.

We also show the oracle performance of the ten hypotheses, which is the upper bound of the consensus reranking. More specifically, for each image in our validation set, we rerank the hypotheses according to the scores (BLEU or CIDEr) w.r.t to the groundtruth captions. The results
of this oracle reranking are shown in Table 2.8 (see rows with “-oracle”). The oracle performance is surprisingly high, indicating that there is still room for improvement, both for the m-RNN model itself and the reranking strategy.

2.10 Conclusion

We propose a multimodal Recurrent Neural Network (m-RNN) framework that performs at the state-of-the-art in three tasks: sentence generation, sentence retrieval given query image and image retrieval given query sentence. The model consists of a deep RNN, a deep CNN and these two sub-networks interact with each other in a multimodal layer. Our m-RNN is powerful of connecting images and sentences and is flexible to incorporate more complex image representations and more sophisticated language models.

Figure 2.4: The original rank of the hypotheses and the rank after consensus reranking (CIDEr).
CHAPTER 3

Learning like a Child: Fast Novel Visual Concept Learning from Sentence Descriptions of Images [MWY15]

3.1 Abstract

In this chapter, we address the task of learning novel visual concepts, and their interactions with other concepts, from a few images with sentence descriptions. Using linguistic context and visual features, our method is able to efficiently hypothesize the semantic meaning of new words and add them to its word dictionary so that they can be used to describe images which contain these novel concepts. Our method has an image captioning module based on [MXY15b] with several improvements. In particular, we propose a transposed weight sharing scheme, which not only improves performance on image captioning, but also makes the model more suitable for the novel concept learning task. We propose methods to prevent overfitting the new concepts. In addition, three novel concept datasets are constructed for this new task. In the experiments, we show that our method effectively learns novel visual concepts from a few examples without disturbing the previously learned concepts.

3.2 Introduction

Recognizing, learning and using novel concepts is one of the important cognitive functions of humans. When we were very young, we learned new concepts by observing the visual world and listening to the sentence descriptions of our parents. The process was slow at the beginning, but got much faster after we accumulated enough learned concepts [Blo02]. In particular, it is known that children can form quick and rough hypotheses about the meaning of new words in a sentence based
Figure 3.1: An illustration of the Novel Visual Concept learning from Sentences (NVCS) task. We start with a model (i.e. model-base) trained with images that does not contain the concept of “quidditch”\(^1\). Using a few “quidditch” images with sentence descriptions, our method is able to learn that “quidditch” is played by people with a ball.

on their knowledge of previous learned words [CB78, HM87], associate these words to the objects or their properties, and describe novel concepts using sentences with the new words [Blo02]. This phenomenon has been researched for over 30 years by the psychologists and linguists who study the process of word learning [Swi10].

For the computer vision field, several methods are proposed [FFP06, STT10, TOC14, LBB14] to handle the problem of learning new categories of objects from a handful of examples. This task is important in practice because we sometimes do not have enough data for novel concepts and hence need to transfer knowledge from previously learned categories. Moreover, we do not want to retrain the whole model every time we add a few images with novel concepts, especially when the amount of data or model parameters is very big.

However, these previous methods concentrate on learning classifiers, or mappings, between single words (e.g. a novel object category) and images. We are unaware of any computer vision studies into the task of learning novel visual concepts from a few sentences and then use these concepts to describe new images – a task that children seem to do effortlessly. We call this the *Novel Visual Concept learning from Sentences (NVCS)* task (see Figure 3.1).

In this chapter, we present a novel framework to address the NVCS task. We start with a model that has already been trained with a large amount of visual concepts. We propose a method that

\(^1\)“quidditch” is a sport created in “Harry Potter”. It is played by teams of people holding brooms (see Figure 3.1).
allow the model to enlarge its word dictionary to describe the novel concepts using a few examples and without extensive retraining. We do not need to retrain models from scratch on all of the data (all the previously learned concepts and the novel concepts). We propose three datasets for the NVCS task to validate our model. These datasets will be made available for use by other researchers.

Our method requires a base model for image captioning which will be adapted to perform the NVCS task. We choose the m-RNN model [MXY15b], which performs at the state of the art, as our base model. Note that we can use most of the current image captioning models in our method. We make several changes to the model structure of m-RNN partly motivated by the desire to avoid overfitting, which is a particular danger for NVCS because we want to learn from a few new images. We note that these changes also improve performance on the original image captioning task, although this improvement is not the main focus of this paper. In particular, we introduce a transposed weight sharing (TWS) strategy (motivated by auto-encoders [Ben09]) which reduces, by a factor of one half, the number of model parameters that need to be learned. This allows us to increase the dimension of the word-embedding and multimodal layers, without overfitting the data, yielding a richer word and multimodal dense representation. We train this image captioning model on a large image dataset with sentence descriptions. This is the base model which we adapt for the NVCS task.

Now we address the task of learning the new concepts from a small new set of data that contains these concepts. There are two main difficulties. Firstly, the weights for the previously learned concepts may be disturbed by the new concepts. Although this can be solved by fixing these weights. Secondly, learning the new concepts from positive examples can introduce bias. Intuitively, the model will assign a baseline probability for each word, which is roughly proportional to the frequency of the words in the sentences. When we train the model on new data, the baseline probabilities of the new words will be unreliably high. We propose a strategy that addresses this problem by fixing the baseline probability of the new words.

We construct three datasets to validate our method, which involves new concepts of man-made objects, animals, and activities. The first two datasets are derived from the MS-COCO dataset [LMB14] which contains over 100,000 images with sentence annotations. The third new dataset is constructed by adding three uncommon concepts which do not occur in MS-COCO or other
standard datasets. These concepts are: quidditch, t-rex and samisen (see details in section 3.6). The experiments show that training our method on only a few examples of the new concepts gives us as good performance as retraining the entire model on all the examples.

3.3 Related Work

<table>
<thead>
<tr>
<th>Layer Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Embedding</td>
<td>Converts words to a vector</td>
</tr>
<tr>
<td>LSTM</td>
<td>Used for sequence modeling</td>
</tr>
<tr>
<td>Multimodal</td>
<td>Combines visual and textual</td>
</tr>
<tr>
<td>Softmax</td>
<td>Output probabilities</td>
</tr>
</tbody>
</table>

Figure 3.2: (a). The architecture of our image captioning model. For each word in a sentence, the model takes the current word index and the image as inputs, and outputs the next word index. The weights are shared across the sub-models for the words in a sentence. The number on the top right of each layer denotes its dimension. As in the m-RNN model [MXY15b], we add a start sign $w_{start}$ and an end sign $w_{end}$ to each training sentence. (b). The transposed weight sharing of $U_D$ and $U_M$.

(Best viewed in color)

Deep neural network Recently there have been dramatic progress in deep neural network models for natural language and computer vision. For natural language, the Recurrent Neural Network (RNN [Elm90, MJC14]) and the Long-Short Term Memory (LSTM [HS97]) achieve the state-of-the-art performance in many NLP tasks such as machine translation [KB13, CMG14, SVL14] and speech recognition [MKB10]. For computer vision, deep Convolutional Neural Networks (CNN [LBO12]) outperform previous methods by a large margin in the tasks of object classification [KSH12, SZ14], detection [GDD14, OLZ14, ZMY14] and segmentation [CPK15, WSL15]. This line of research provides support for the development of the methods for the multimodal learning
tasks (e.g. image captioning and sentence-image retrieval).

**Multimodal learning of language and vision** The methods of image-sentence retrieval [FCS13, SLM14] and image description generation [KPD11, MHD12, GM12] have developed very fast in recent years. Recently, [MXY14] proposed a multimodal RNN model that achieves the state-of-the-art performance for both the tasks of image-sentence retrieval and image captioning. The m-RNN model consists of three components: (I) A vision component which uses deep Convolutional Neural Networks (CNN) to provide semantic visual attributes and features for objects and scenes. (II) A language model based on recurrent neural networks. (III) A multimodal component which couples the vision and language components and generates the sentence descriptions. Closely related works includes [KSZ14, KF14, VTB14, DHG14, FGI14, CZ14, LPC14, MF14b, KLS14, XBK15]. The evaluation metrics of the image captioning task is discussed [EK14, VZP14]. All of these methods use a pre-specified and fixed word dictionary, and train their model on a large dataset.

**Zero-shot and one-shot learning** For zero-shot learning, the task is to associate dense word vectors or attributes with image features [SGM13, FCS13, ESE13, AZP14, LBB14]. The dense word vectors in these papers are pre-trained from a large amount of text corpus and the word semantic representation is captured from co-occurrence with other words [MSC13]. [LBB14] developed this idea by only showing the novel words a few times. Another closely related task is one-shot learning task of new categories [FFP06, LSG11, TOC14]. They learn new objects from only a few examples. However, these work only consider words or attributes instead of sentences and so their learning target is different from that of the task in this chapter.

### 3.4 The Image Captioning Model

We need an image captioning as the base model which will be adapted in the NVCS task. The base model is developed based on the m-RNN model [MXY15b]. Its architecture is shown in Figure 3.2(a). We make two main modifications of the architecture to make it more suitable for the NVCS task which, as a side effect, also improves performance on the original image captioning task. Firstly and most importantly, we propose a transposed weight sharing strategy which significantly reduces the number of parameters in the model (see section 3.4.2). Secondly, we replace the
3.4.1 The Model Architecture

As shown in Figure 3.2(a), the input of our model for each word in a sentence is the index of the current word in the word dictionary as well as the image. We represent this index as a one-hot vector (a binary vector with only one non-zero element indicating the index). The output is the index of the next word. The model has three components: the language component, the vision component and the multimodal component. The language component contains two word embedding layers and a LSTM layer. It maps the index of the word in the dictionary into a semantic dense word embedding space and stores the word context information in the LSTM layer. The vision component contains a 16-layer deep convolutional neural network (CNN [SZ14]) pre-trained on the ImageNet classification task [RDS14]. We remove the final SoftMax layer of the deep CNN and connect the top fully connected layer (a 4096 dimensional layer) to our model. The activation of this 4096 dimensional layer can be treated as image features that contain rich visual attributes for objects and scenes. The multimodal component contains a one-layer representation where the information from language part and vision part merge together. We build a SoftMax layer after the multimodal layer to predict the index of the next word. The weights are shared across the sub-models of the words in a sentence. As in the m-RNN model [MXY15b], we add a start sign $w_{\text{start}}$ and an end sign $w_{\text{end}}$ to each training sentence. In the testing stage for image captioning, we input the start sign $w_{\text{start}}$ into the model and pick the $K$ best words with maximum probabilities according to the SoftMax layer. We repeat the process until the model generates the end sign $w_{\text{end}}$.

3.4.2 The Transposed Weight Sharing (TWS)

For the original m-RNN model [MXY15b], most of the weights (i.e. 98.49%) are contained in the following two weight matrices: $U_D \in \mathbb{R}^{512 \times N}$ and $U_M \in \mathbb{R}^{N \times 1024}$ where $N$ represents the size of...
the word dictionary.

The weight matrix $U_D$ between the one-hot layer and first word embedding layer is used to compute the input of the first word embedding layer $w(t)$:

$$w(t) = f(U_D h(t))$$  \hspace{1cm} (3.1)

where $f(.)$ is an element-wise non-linear function, $h(t) \in \mathbb{R}^{N \times 1}$ is the one-hot vector of the current word. Note that it is fast to calculate Equation 3.1 because there is only one non-zero element in $h(t)$. In practice, we do not need to calculate the full matrix multiplication operation since only one column of $U_D$ is used for each word in the forward and backward propagation.

The weight matrix $U_M$ between the multimodal layer and the SoftMax layer is used to compute the activation of the SoftMax layer $y(t)$:

$$y(t) = g(U_M m(t) + b)$$  \hspace{1cm} (3.2)

where $m(t)$ is the activation of the multimodal layer, $g(.)$ is the SoftMax non-linear function.

Intuitively, the role of the weight matrix $U_D$ in Equation 3.1 is to encode the one-hot vector $h(t)$ into the dense semantic vector $w(t)$. The role of the weight matrix $U_M$ in Equation 3.2 is to decode the dense semantic vector $m(t)$ back to a pseudo one-hot vector $y(t)$ with the help of the SoftMax function, which is very similar to the inverse operation of Equation 3.1. The difference is that $m(t)$ is in the dense multimodal semantic space while $w(t)$ is in the dense word semantic space.

To reduce the number of the parameters, we decompose $U_M$ into two parts. The first part maps the multimodal layer activation vector to an intermediate vector in the word semantic space. The second part maps the intermediate vector to the pseudo one-hot word vector, which is the inverse operation of Equation 3.1. The sub-matrix of the second part is able to share parameters with $U_D$ in a transposed manner, which is motivated by the tied weights strategy in auto-encoders for unsupervised learning tasks [Ben09]. Here is an example of linear decomposition: $U_M = U_D^T U_1$, where $U_1 \in \mathbb{R}^{512 \times 1024}$. Equation 3.2 is accordingly changed to:

$$y(t) = g[U_D^T f(U_1 m(t)) + b]$$  \hspace{1cm} (3.3)

where $f(.)$ is a element-wise function. If $f(.)$ is an identity mapping function, it is equivalent to linearly decomposing $U_M$ into $U_D^T$ and $U_1$. In our experiments, we find that setting $f(.)$ as the scaled
hyperbolic tangent function leads to a slightly better performance than linear decomposition. This strategy can be viewed as adding an intermediate layer with dimension 512 between the multimodal and SoftMax layers as shown in Figure 3.2(b). The weight matrix between the intermediate and the SoftMax layer is shared with $U_D$ in a transposed manner. This Transposed Weight Sharing (TWS) strategy enables us to use a much larger dimensional word-embedding layer than the m-RNN model [MXY15b] without increasing the number of parameters. We also benefit from this strategy in the novel concept learning task.

### 3.5 The Novel Concept Learning (NVCS) Task

Suppose we have trained a model based on a large amount of images and sentences. Then we meet with images of novel concepts whose sentence annotations contain words not in our dictionary, what should we do? It is time-consuming and unnecessary to re-training the whole model from scratch using all the data. In many cases, we cannot even access the original training data of the model. But fine-tuning the whole model using only the new data causes severe overfitting on the new concepts and decrease the performance of the model for the originally trained concepts.

To solve these problems, we propose the following strategies that learn the new concepts with a
few images without losing the accuracy on the original concepts.

3.5.1 Fixing the originally learned weights

Under the assumption that we have learned the weights of the original words from a large amount of data and that the amount of the data for new concepts is relatively small, it is straightforward to fix the originally learned weights of the model during the incremental training. More specifically, the weight matrix \( U_D \) can be separated into two parts: \( U_D = [U_{D_o}; U_{D_n}] \), where \( U_{D_o} \) and \( U_{D_n} \) associate with the original words and the new words respectively. E.g., as shown in Figure 3.3, for the novel visual concept “cat”, \( U_{D_n} \) is associated with 29 new words, such as cat, kitten and pawing. We fix the sub-matrix \( U_{D_o} \) and update the sub-matrix \( U_{D_n} \) as illustrated in Figure 3.3.

3.5.2 Fixing the baseline probability

In Equation [3.3] there is a bias term \( b \). Intuitively, each element in \( b \) represents the tendency of the model to output the corresponding word. We can think of this term as the baseline probability of each word. Similar to \( U_D \), \( b \) can be separated into two parts: \( b = [b_o, b_n] \), where \( b_o \) and \( b_n \) associate with the original words and the new words respectively. If we only present the new data to the network, the estimation of \( b_n \) is unreliable. The network will tend to increase the value of \( b_n \) which causes the overfitting to the new data.

The easiest way to solve this problem is to fix \( b_n \) during the training for novel concepts. But this is not enough. Because the average activation \( \bar{x} \) of the intermediate layer across all the training samples is not 0, the weight matrix \( U_D \) plays a similar role to \( b \) in changing the baseline probability. To avoid this problem, we centralize the activation of the intermediate layer \( x \) and turn the original bias term \( b \) into \( b' \) as follows:

\[
y(t) = g[U_D^T(x - \bar{x}) + b']; \quad b'_o = b_o + U_D^T \bar{x}
\]

After that, we set every element in \( b'_n \) to be the average value of the elements in \( b'_o \) and fixing \( b'_n \) during the training on the new images. We call this strategy Baseline Probability Fixation (BPF).
3.5.3 The Role of Language and Vision

In the novel concept learning (NVCS) task, the sentences serve as a weak labeling of the image. The language part of the model (the word embedding layers and the LSTM layer) hypothesizes the basic properties (e.g. the parts of speech) of the new words and whether the new words are closely related to the content of the image. It also hypothesizes which words in the original dictionary are semantically and syntactically close to the new words. For example, suppose the model meets a new image with the sentence description “A woman is playing with a cat”. Also suppose there are images in the original data contain sentence description such as “A man is playing with a dog”. Then although the model does not see the word “cat” before, it will hypothesize that the word “cat” and “dog” are close to each other.

The vision part is pre-trained on the ImageNet classification task [RDS14] with 1.2 million images and 1,000 categories. It provides rich visual attributes of the objects and scenes [ZF14] that are useful not only on the 1,000 classification task itself, but also on other vision tasks [DJV13].

Combining cues from both language and vision, our model can effectively learn the new concepts using only a few examples as demonstrated in the experiments.

Figure 3.4: Organization of the novel concept datasets
3.6 Datasets

3.6.1 Strategies to Construct Datasets

We use the annotations and images from the MS COCO dataset [LMB14] to construct our Novel Concept (NC) learning datasets. The current release of MS COCO contains 82,783 training images and 40,504 validation images, with object instance annotations and around 5 sentence descriptions for each image. To construct the NC dataset with a specific new concept (e.g. “cat”), we remove all images containing the object “cat” according to the object instance annotations. We also check whether there are some images left with sentences descriptions containing cat related words. The images left are treated as the Base Set where we will train and validate our base model. The removed images are used to construct the Novel Concept set (NC set), which is used to train and validate our model for the task of novel concept learning.

3.6.2 The Novel Visual Concepts Datasets

We construct three datasets involving five different novel visual concepts:

**NewObj-Cat** and **NewObj-Motor** The corresponding new concepts of these two datasets are “cat” and “motorcycle” respectively. The model need to learn all the related words that represent and describe these concepts and their activities.

**NC-3 dataset** The above two datasets are all derived from the MS COCO dataset. To further
verify the effectiveness of our method, we construct a new dataset contains three novel concepts: “quidditch” (a recently created sport derived from “Harry Potter”), “t-rex” (a dinosaur), and “samisen” (an instrument). It contains not only object concepts (e.g. t-rex and samisen), but also activity concepts (e.g. quidditch). We labeled 100 images for each concept with 5 sentence annotations for each image. To diversify the labeled sentences for different images in the same category, the annotators are instructed to label the images with different sentences by describing the details in each image. It leads to a different style of annotation from that of the MS COCO dataset. The average length of the sentences is also 26% longer than that of the MS COCO (13.5 v.s. 10.7). There is no duplicated annotation sentences in this dataset. We construct this dataset for two reasons. Firstly, the three concepts are not included in the 1,000 categories of the ImageNet Classification task [RDS14] where we pre-trained the vision component of our model. Secondly, this dataset has richer and more diversified sentence descriptions compared to NewObj-Cat and NewObj-Motor. We denote this dataset as Novel Concept-3 dataset (NC-3). Some samples images and annotations are shown in Figure 3.5.

We randomly separate the above three datasets into training, testing and validation sets. The number of images for the three datasets are shown in Table 3.1. To investigate the possible overfitting issues on the new object dataset, in the testing stage, we randomly picked images from the testing set of the Base Set and treated them as a separate set of testing images. The number of added images is equal to the size of the original test set (e.g. 1000 images are picked for NewObj-Cat testing set). We denote the original new concept testing images as Novel Concept (NC) test set and the added base testing images as Base test set. The organization of NC datasets is illustrated in Figure 3.4.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>NC Test</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewObj-Cat</td>
<td>2840</td>
<td>1000</td>
<td>490</td>
</tr>
<tr>
<td>NewObj-Motor</td>
<td>1854</td>
<td>600</td>
<td>349</td>
</tr>
<tr>
<td>NC-3</td>
<td>150 (50 × 3)</td>
<td>120 (40 × 3)</td>
<td>30 (10 × 3)</td>
</tr>
</tbody>
</table>

Table 3.1: The number of images for the three datasets.
We will make the three datasets publicly available upon publication to encourage future research in this area.

3.7 Experiments

3.7.1 Evaluation Metrics

To evaluate the output sentence descriptions for novel visual concepts, we adopt two evaluation metrics that are widely used in recent image captioning work: BLEU scores [PRW02] (BLEU score for n-gram is denoted as as B-n in the paper) and METEOR [LA07].

Both BLEU scores and METEOR target on evaluating the overall quality of the generated sentences. In the NVCS task, however, we focus more on the accuracy for the new words than the previously learned words in the sentences. Therefore, to conduct a comprehensive evaluation, we also calculate the $f$ score for the words that describe the new concepts. E.g. for the cat dataset, there are 29 new words such as cat, cats, kitten, and pawing. The precision $p$ and recall $r$ for each new word in the dictionary ($w_n^d$) are calculated as follows:

$$p = \frac{N(w_n^d \in S_{gen} \land w_n^d \in S_{ref})}{N(w_n^d \in S_{gen})}; r = \frac{N(w_n^d \in S_{gen} \land w_n^d \in S_{ref})}{N(w_n^d \in S_{ref})}$$

where $S_{gen}$ denotes generated sentence, $S_{ref}$ denotes reference sentences, $N(condition)$ represents number of testing images that conform to the condition. Note that $p$ and $r$ are calculated on the combined testing set of the NC test set and the base test set (i.e. All test).

A high $r$ with a low $p$ indicates that the model overfits the new data (We can always get $r = 1$ if we output the new word every time) while a high $p$ with a low $r$ indicates underfitting. We use the $f = \frac{2}{p^{-1} + r^{-1}}$ as a balanced measurement between $p$ and $r$. Best $f$ score is 1. Note that $f = 0$ if either $p = 0$ or $r = 0$. Compared to METEOR and BLEU, the $f$ score show the effectiveness of the model to learn new concepts more explicitly.
### Table 3.2: The performance of our model compared with the state-of-the-art for the standard image captioning task.

<table>
<thead>
<tr>
<th></th>
<th>B-1</th>
<th>B-2</th>
<th>B-3</th>
<th>B-4</th>
<th>METEOR</th>
<th>CIDEr</th>
<th>ROUGE_L</th>
</tr>
</thead>
<tbody>
<tr>
<td>m-RNN [MXY15b]</td>
<td>0.680</td>
<td>0.506</td>
<td>0.369</td>
<td>0.272</td>
<td>0.225</td>
<td>0.791</td>
<td>0.499</td>
</tr>
<tr>
<td>ours-TWS</td>
<td>0.685</td>
<td>0.512</td>
<td>0.376</td>
<td>0.279</td>
<td>0.229</td>
<td>0.819</td>
<td>0.504</td>
</tr>
</tbody>
</table>

#### 3.7.2 Effectiveness of TWS and BPF

We test our base model with the Transposed Weight Sharing (TWS) strategy in the original image captioning task on the MS COCO test set and evaluate the results using their evaluation server [CFL15]. The results are shown in Table 3.2. Our model performs at the state-of-the-art in this task. We choose the layer dimensions of our model so that the number of parameters matches that of [MXY15b]. Models with different layer dimensions might lead to better performance, which is beyond the scope of this paper.

We also validate the effectiveness of our Transposed Weight Sharing (TWS) and Baseline Probability Fixation (BPF) strategies for the novel concept learning task on the NewObj-Cat dataset. We compare the performance of four Deep-NVCS models. Their properties and performance in terms of $f$ score for the word “cat” are summarized in Table 3.3. “BiasFix” means that we fix the bias term $b_n$ in Equation 3.3. “Centralize” means that we centralize the intermediate layer activation $x$ (see Equation 3.4) so that $U_D$ will not affect the baseline probability.

We achieve 2.5% increase of performance in terms of $f$ using TWS (Deep-NVCS-BPF-TWS v.s. Deep-NVCS-BPF-noTWS), and achieves 2.4% increase using BPF (Deep-NVCS-BPF-TWS v.s. Deep-NVCS-UnfixedBias). We use Deep-NVCS to represent Deep-NVCS-BPF-TWS in short for the rest of the paper.
Table 3.3: Performance of Deep-NVCS models with different novel concept learning strategies on NewObj-Cat. TWS and BPF improves the performance.

<table>
<thead>
<tr>
<th>Model</th>
<th>BiasFix</th>
<th>Centralize</th>
<th>TWS</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deep-NVCS-UnfixedBias</td>
<td>×</td>
<td>×</td>
<td>✓</td>
<td>0.851</td>
</tr>
<tr>
<td>Deep-NVCS-FixedBias</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>0.860</td>
</tr>
<tr>
<td>Deep-NVCS-BPF-NoTWS</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>0.850</td>
</tr>
<tr>
<td>Deep-NVCS-BPF-TWS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>0.875</td>
</tr>
</tbody>
</table>

Figure 3.6: Performance comparison of our model with different number of training images on NewObj-cat and NewObj-Motor datasets. The red, blue, and magenta dashed line indicates the performance of Deep-NVCS using all the training images on the base test set, the NC test set and the all test set respectively. The green dashed line indicates the performance of Model-base. We show that our model trained with 10 to 50 images achieves comparable performance with the model trained on the full training set. (Best viewed in color)

3.7.3 Results on NewObj-Motor and NewObj-Cat

3.7.3.1 Using all training samples

We show the performance of our Deep-NVCS models compared to strong baselines on the NewObj-Cat and NewObj-Motor datasets in Table 3.4. For Deep-NVCS, we only use the training data from the novel concept set. For Deep-NVCS-Inc1:1, we add training data randomly sampled from the training set of the base set. The number of added training images is the same as that of the training
Table 3.4: Results of our model on the NewObj-Cat and NewObj-Motor dataset using all the training samples. Model-base and Model-retrain stand for the model trained on base set (no novel concept images) and the model retrained on the combined data (all the images of base set and novel concept set) respectively. Deep-NVCS stands for model trained only with the new concept data. Deep-NVCS-1:1Inc stands for the Deep-NVCS model trained by adding equal number of training images from the base set. Compared to Model-base which is only trained on the base set, the Deep-NVCS models achieve a significant improvement on the novel concept test set while reaching comparable performance on the base test set. The performance of our Deep-NVCS models is very close to that of the strong baseline Model-retrain, which requires much more training.
disturbing the previous learned words.

The performance of Deep-NVCS is also comparable with, though slightly lower than that of Deep-NVCS-1:1Inc. Intuitively, if the image features can successfully capture the difference between the new concepts and the existing ones, it is sufficient to learn the new concept only from the new data. However, if the new concepts are very similar to some previously learned concepts, such as cat and dog, it is helpful to present the data of both novel and existing concepts to make it easier for the model to find the difference.

3.7.3.2 Using a few training samples

To answer the question whether we can learn novel concepts from a few examples, we also test our model under the one or few-shot scenario. Specifically, we randomly sampled $k$ images from the training set of NewObj-Cat and NewObj-Motor, and trained our Deep-NVCS model only on these images. $k$ ranges from 1 to 1000. We conduct the experiments 10 times and average the results to avoid the randomness of the sampling.

We show the performance of our model with different number of training images in Figure 3.6. We only show the results in terms of $f$ score, METEOR, B-3 and B-4 because of space limitation. The results of B-1 and B-2 and consistent with the shown metrics. The performance of the model trained with the full NC training set in the last section is indicated by the blue (Base test), red (NC test) or magenta (All test) dashed lines in Figure 3.6. These lines represent the experimental upper bounds of our model under the one or few-shot scenario. The performance of the Model-base is drawn as a green dashed line. It serves as an experimental lower bound.

The results show that using about 10 to 50 training images, the model achieves comparable performance with the Deep-NVCS model trained on the full novel concept training set. In addition, using about 5 training images, we observe a nontrivial increase of performance compared to the base model.
Figure 3.7: The generated sentences for the test images from novel concept datasets. In these examples, cat, motorcycle, quidditch, t-rex and samisen are the novel concepts respectively. We show more in the supplementary material.

### 3.7.4 Results on NC-3

The NC-3 dataset has three main difficulties. Firstly, the concepts have very similar counterparts in the original image set, such as Samisen v.s. Guitar, Quidditch v.s. football. Secondly, the three concepts rarely appear in daily life. They are not included in the ImageNet 1,000 categories where we pre-trained our vision deep CNN. Thirdly, the way we describe the three novel concepts is somewhat different from that of the common objects included in the base set. The different group of annotators as well as the requirement to diversify the annotated sentences makes the difference of the style for the annotated sentences between NC-3 and MS COCO even larger. The effect of the difference in sentence style leads to decreased performance of the base model compared to that on the NewObj-Cat and NewObj-Motor dataset (see Model-base in Table 3.5 compared to that in Table 3.4 on NC test). Furthermore, it makes it harder for the model to hypothesize the meanings of new words from a few sentences.

Facing with these difficulties, our model still learns the semantic meaning of the new concepts quite well. The $f$ scores of the model shown in Table 3.5 indicate that the model successfully learn the new concepts with a high accuracy from only 50 examples.

It is interesting that Model-retrain performs very badly in this dataset. It does not output the word “quidditch” and “samisen” in the generated sentences. The BLEU scores and METEOR are also very low. This is not surprising since there are only a few training examples (i.e. 50) for these three novel concepts and so it is easy to be overwhelmed by other concepts from the original MS...
<table>
<thead>
<tr>
<th>Evaluation Metrics</th>
<th>$f$</th>
<th>B-1</th>
<th>B-2</th>
<th>B-3</th>
<th>B-4</th>
<th>METEOR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>quidditch</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model-retrain</td>
<td>0.000</td>
<td>0.509</td>
<td>0.283</td>
<td>0.196</td>
<td>0.138</td>
<td>0.120</td>
</tr>
<tr>
<td>Model-base</td>
<td>-</td>
<td>0.506</td>
<td>0.272</td>
<td>0.193</td>
<td>0.139</td>
<td>0.122</td>
</tr>
<tr>
<td>Deep-NVCS</td>
<td>0.854</td>
<td>0.582</td>
<td>0.353</td>
<td>0.237</td>
<td>0.167</td>
<td>0.168</td>
</tr>
<tr>
<td>Deep-NVCS-1:1Inc</td>
<td>0.863</td>
<td>0.585</td>
<td>0.359</td>
<td>0.244</td>
<td>0.170</td>
<td>0.170</td>
</tr>
<tr>
<td><strong>samisen</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model-retrain</td>
<td>0.000</td>
<td>0.539</td>
<td>0.343</td>
<td>0.209</td>
<td>0.133</td>
<td>0.122</td>
</tr>
<tr>
<td>Model-base</td>
<td>-</td>
<td>0.489</td>
<td>0.304</td>
<td>0.177</td>
<td>0.105</td>
<td>0.122</td>
</tr>
<tr>
<td>Deep-NVCS</td>
<td>0.630</td>
<td>0.545</td>
<td>0.371</td>
<td>0.229</td>
<td>0.140</td>
<td>0.161</td>
</tr>
<tr>
<td>Deep-NVCS-1:1Inc</td>
<td>0.642</td>
<td>0.551</td>
<td>0.374</td>
<td>0.233</td>
<td>0.144</td>
<td>0.164</td>
</tr>
<tr>
<td><strong>t-rex</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model-retrain</td>
<td>0.213</td>
<td>0.568</td>
<td>0.357</td>
<td>0.224</td>
<td>0.141</td>
<td>0.105</td>
</tr>
<tr>
<td>Model-base</td>
<td>-</td>
<td>0.497</td>
<td>0.278</td>
<td>0.166</td>
<td>0.102</td>
<td>0.088</td>
</tr>
<tr>
<td>Deep-NVCS</td>
<td>0.861</td>
<td>0.634</td>
<td>0.422</td>
<td>0.247</td>
<td>0.144</td>
<td>0.187</td>
</tr>
<tr>
<td>Deep-NVCS-1:1Inc</td>
<td>0.856</td>
<td>0.629</td>
<td>0.419</td>
<td>0.242</td>
<td>0.132</td>
<td>0.186</td>
</tr>
<tr>
<td><strong>Base Test</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model-retrain</td>
<td>-</td>
<td>0.701</td>
<td>0.538</td>
<td>0.412</td>
<td>0.328</td>
<td>0.234</td>
</tr>
<tr>
<td>Model-base</td>
<td>-</td>
<td>0.702</td>
<td>0.537</td>
<td>0.414</td>
<td>0.325</td>
<td>0.240</td>
</tr>
<tr>
<td>Deep-NVCS</td>
<td>-</td>
<td>0.703</td>
<td>0.537</td>
<td>0.414</td>
<td>0.326</td>
<td>0.239</td>
</tr>
<tr>
<td>Deep-NVCS-1:1Inc</td>
<td>-</td>
<td>0.703</td>
<td>0.537</td>
<td>0.414</td>
<td>0.327</td>
<td>0.239</td>
</tr>
</tbody>
</table>

Table 3.5: Results of our model on the NC-3 Datasets.
<table>
<thead>
<tr>
<th>New Word</th>
<th>Five nearest neighbours</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>kitten; tabby; puppy; calico; doll;</td>
</tr>
<tr>
<td>motorcycle</td>
<td>motorbike; moped; vehicle; motor; motorbikes;</td>
</tr>
<tr>
<td>quidditch</td>
<td>soccer; football; softball; basketball; frisbees;</td>
</tr>
<tr>
<td>t-rex</td>
<td>giraffe’s; bull; pony; goat; burger;</td>
</tr>
<tr>
<td>samisen</td>
<td>guitar; wii; toothbrushes; purse; contents;</td>
</tr>
</tbody>
</table>

Table 3.6: The five nearest neighbors of the new words as measured by the activation of the word-embedding layer.

COCO dataset.

3.7.5 Qualitative Results

In Table 3.6, we show the five nearest neighbors of the new concepts using the activation of the word-embedding layer learned by our Deep-NVCS model. It shows that the learned novel word embedding vectors captures the semantic information from both language and vision. We also show some sample generated sentence descriptions of the base model and our Deep-NVCS model in Figure 3.7.

3.8 Conclusion

In this chapter, we propose the Novel Visual Concept learning from Sentences (NVCS) task. In this task, methods need to learn novel concepts from sentence descriptions of a few images. We describe a method that allows us to train our model on a small number of images containing novel concepts. This performs comparably with the model retrained from scratch on all of the data if the number of novel concept images is large. Our model performs better than the retrained model when there are only a few training images of novel concepts available. The model is based on an image captioning model [MXY15b] with several improvements, particularly the Transposed Weight Sharing (TWS),
which leads to a better performance on the novel concept learning task. We construct three novel concept datasets where we validate the effectiveness of our method.
CHAPTER 4

Generation and Comprehension of Unambiguous Object Descriptions [MHT16]

4.1 Abstract

We propose a method that can generate an unambiguous description (known as a referring expression) of a specific object or region in an image, and which can also comprehend or interpret such an expression to infer which object is being described. We show that our method outperforms previous methods that generate descriptions of objects without taking into account other potentially ambiguous objects in the scene. Our model is inspired by recent successes of deep learning methods for image captioning, but while image captioning is difficult to evaluate, our task allows for easy objective evaluation. We also present a new large-scale dataset for referring expressions, based on MS-COCO. We have released the dataset and a toolbox for visualization and evaluation, see https://github.com/mjhucla/Google_Refexp_toolbox.

4.2 Introduction

There has been a lot of recent interest in generating text descriptions of images (see e.g., [FHST10, VTB14, DHG14, CZ14, FGI14, KF14, KSZ14, MXY15b, XBK15, DGG15]). However, fundamentally this problem of image captioning is subjective and ill-posed. With so many valid ways to describe any given image, automatic captioning methods are thus notoriously difficult to evaluate. In particular, how can we decide that one sentence is a better description of an image than another?

In this chapter, we focus on a special case of text generation given images, where the goal is to generate an unambiguous text description that applies to exactly one object or region in the
Figure 4.1: Illustration of our generation and comprehension system. On the left we see that the system is given an image and a region of interest; it describes it as “the man who is touching his head”, which is unambiguous (unlike other possible expressions, such as “the man wearing blue”, which would be unclear). On the right we see that the system is given an image, an expression, and a set of candidate regions (bounding boxes), and it selects the region that corresponds to the expression.

Such a description is known as a “referring expression” [DSG06, VD08, MDR10, MDR13, FAZ13, GLK10, KOM14]. This approach has a major advantage over generic image captioning, since there is a well-defined performance metric: a referring expression is considered to be good if it uniquely describes the relevant object or region within its context, such that a listener can comprehend the description and then recover the location of the original object. In addition, because of the discriminative nature of the task, referring expressions tend to be more detailed (and therefore more useful) than image captions. Finally, it is easier to collect training data to “cover” the space of reasonable referring expressions for a given object than it is for a whole image.

We consider two problems: (1) *description generation*, in which we must generate a text expression that uniquely pinpoints a highlighted object/region in the image and (2) *description comprehension*, in which we must automatically select an object given a text expression that refers to this object (see Figure 4.1). Most prior work in the literature has focused exclusively on description generation (e.g., [KD12, KOM14]). Golland *et al.* [GLK10] consider generation and
comprehension, but they do not process real world images.

In this chapter, we jointly model both tasks of description generation and comprehension, using state-of-the-art deep learning approaches to handle real images and text. Specifically, our model is based upon recently developed methods that combine convolutional neural networks (CNNs) with recurrent neural networks (RNNs). We demonstrate that our model outperforms a baseline which generates referring expressions without regard to the listener who must comprehend the expression. We also show that our model can be trained in a semi-supervised fashion, by automatically generating descriptions for image regions.

Being able to generate and comprehend object descriptions is critical in a number of applications that use natural language interfaces, such as controlling a robot (e.g., “Rosie, please fetch me the beer from the top shelf of the fridge”, cf. [BBY15]), or interacting with photo editing software (e.g., “Picasa, please replace the third car behind the fence with a motorbike”, cf. [CZL14]). In addition, it is a good test bed for performing research in the area of vision and language systems because of the existence of a useful objective performance measure.

In order to train and evaluate our system, we have collected and released a new large scale referring expressions dataset based on the popular MS-COCO dataset [LMB14].

To summarize, our main contributions are as follows. First, we present a new large scale dataset for referring expressions. Second, we evaluate how existing image captioning methods perform at the referring expression task. Third, we develop a new method for joint generation and comprehension that outperforms current methods.

### 4.3 Related Work

**Referring expressions.** Referring expression generation is a classic NLP problem (see e.g., [Win72, KD12]). Important issues include understanding what types of attributes people typically use to describe visual objects (such as color and size) [MDR13], usage of higher-order relationships (e.g., spatial comparison) [VD08], and the phenomena of over and under-specification, which is also related to speaker variance [FAZ13].
Context (sometimes called pragmatics \cite{GL14}) plays a critical role in several ways \cite{KT02}. First, the speaker must differentiate the target object from a collection of alternatives and must thus reason about how the object differs from its context. Second, the perception of the listener is also valuable. In particular, Golland \textit{et al.} \cite{GLK10} recently proposed a game theoretic formulation of the referring expression problem showing that speakers that act optimally with respect to an explicit listener model naturally adhere to the Gricean Maxims of communication \cite{Gri70}.

In most of this previous work, authors have focused on small datasets of computer generated objects (or photographs of simple objects) \cite{DSG06,MDR10} and have not connected their text generation systems to real vision systems. However there has been recent interest in understanding referring expressions in the context of complex real world images, for which humans tend to generate longer phrases \cite{GRB15}. Kazemzadeh \textit{et al.} \cite{KOM14} were the first to collect a large scale dataset of referring expressions for complex real world photos.

We likewise collect and evaluate against a large scale dataset. However we go beyond expression generation and jointly learn both generation and comprehension models. And where prior works have had to explicitly enumerate attribute categories such as size, color (e.g. \cite{SCS12}) or manually list all possible visual phrases (e.g. \cite{SF11}), our deep learning-based models are able to learn to directly generate surface expressions from raw images without having to first convert to a formal object/attribute representation.

Concurrently, \cite{HXR16} propose a CNN-RNN based method that is similar to our baseline model and achieve state-of-the-art results on the ReferIt dataset \cite{KOM14}. But they did not use the discriminative training strategy proposed in our full model. \cite{JKF15,KZG16} investigate the task of generating dense descriptions in an image. But their descriptions are not required to be unambiguous.

**Image captioning.** Our methods are inspired by a long line of inquiry in joint models of images and text, primarily in the vision and learning communities \cite{FHS10,HYH13,SLM14,OKB11a,KPD11,YTD11,LKB11}. From a modeling perspective, our approach is closest to recent works applying RNNs and CNNs to this problem domain \cite{VTB14,DHG14,CZ14,FGI14,KF14,KSZ14,MXY15b,XBK15}. The main approach in these papers is to represent the image content using the
hidden activations of a CNN, and then to feed this as input to an RNN, which is trained to generate a sequence of words.

Most papers on image captioning have focused on describing the full image, without any spatial localization. However, we are aware of two exceptions. [XBK15] propose an attention model which is able to associate words to spatial regions within an image; however, they still focus on the full image captioning task. [KF14] propose a model for aligning words and short phrases within sentences to bounding boxes; they then train a model to generate these short snippets given features of the bounding box. Their model is similar to our baseline model, described in Section 4.6 (except we provide the alignment of phrases to boxes in the training set, similar to [PWC15]). However, we show that this approach is not as good as our full model, which takes into account other potentially confusing regions in the image.

**Visual question answering.** Referring expressions is related to the task of VQA (see e.g., [AAL15, MF14a, MRF15, GGH15, GMZ15]). In particular, referring expression comprehension can be turned into a VQA task where the speaker asks a question such as “where in the image is the car in red?” and the system must return a bounding box (so the answer is numerical, not linguistic). However there are philosophical and practical differences between the two tasks. A referring expression (and language in general) is about communication — in our problem, the speaker is finding the optimal way to communicate to the listener, whereas VQA work typically focuses only on answering questions without regard to the listener’s state of mind. Additionally, since questions tend to be more open ended in VQA, evaluating their answers can be as hard as with general image captioning, whereas evaluating the accuracy of a bounding box is easy.

### 4.4 Dataset Construction

The largest existing referring expressions dataset that we know of is the *ReferIt dataset*, which was collected by [KOM14], and contains 130,525 expressions, referring to 96,654 distinct objects, in 19,894 photographs of natural scenes. Images in this dataset are from the segmented and annotated TC-12 expansion of the ImageCLEF IAPR dataset [EHG10]. Two drawbacks of this dataset, however, are that (1) the images sometimes only contain one object of a given class, allowing
Figure 4.2: Some sample images from our Google Refexp (G-Ref) dataset. We use a green dot to indicate the object that the descriptions refer to. Since the dataset is based on MS COCO, we have access to the original annotations such as the object mask and category. Some of the objects are hard to describe, e.g., in the third image in the first row, we need to distinguish the boy from his reflection in the mirror.

Figure 4.3: Comparison between the G-Ref and UNC-Ref dataset.

speakers to use short descriptions without risking ambiguity, and (2) the ImageCLEF dataset focuses mostly on “stuff” (i.e. context) rather than “things” (i.e. objects).

In this chapter, we use a similar methodology to that of [KOM14], but building instead on top of the MSCOCO dataset [LMB14], which contains more than 300,000 images, with 80 categories of objects segmented at the instance level.

For each image, we selected objects if (1) there are between 2 and 4 instances of the same object type within the same image, and (2) if their bounding boxes occupy at least 5% of image area. This resulted in selecting 54,822 objects from 26,711 images. We constructed a Mechanical Turk task in which we presented each object in each image (by highlighting the object mask) to
a worker whose task was to generate a unique text description of this object. We then used a second task in which a different worker was presented with the image and description, and was asked to click inside the object being referred to. If the selected point was inside the original object’s segmentation mask, we considered the description as valid, and kept it, otherwise we discarded it and re-annotated it by another worker. We repeated these description generation and verification tasks on Mechanical Turk iteratively up to three times. In this way, we selected 104,560 expressions. Each object has on average 1.91 expressions, and each image has on average 3.91 expressions. This dataset is denoted as Google Refexp dataset and some samples are shown in Figure 4.2. We have released this dataset and a toolbox for visualization and evaluation, see https://github.com/mjhucla/Google_Refexp_toolbox.

While we were collecting our dataset, we learned that Tamara Berg had independently applied her ReferIt game [KOM14] to the MSCOCO dataset to generate expressions for 50,000 objects from 19,994 images. She kindly shared her data (named as UNC-Ref-COCO dataset) with us. For brevity, we call our Google Refexp dataset as G-Ref and the UNC-Ref-COCO as UNC-ref. We report results on both datasets in this chapter. However, due to differences in our collection methodologies, we have found that the descriptions in the two overlapped datasets exhibit significant qualitative differences, with descriptions in the UNC-Ref dataset tending to be more concise and to contain less flowery language than our descriptions. More specifically, the average lengths of expressions from our dataset and UNC-Ref are 8.43 and 3.61 respectively. And the size of the word dictionaries (keeping only words appearing more than 3 times) from our dataset and UNC-Ref are 4849 and 2890 respectively. See Figure 4.3 for some visual comparisons.

### 4.5 Tasks

In this section, we describe at a high level how we solve the two main tasks of description and generation. We will describe the model details and training in the next section.

---

1 According to our personal communication with the authors of the UNC-Ref dataset, the instruction and reward rule of UNC-Ref encourages the annotators to give a concise description in a limited time, while in our G-Ref dataset, we encourage the annotators to give rich and natural descriptions. This leads to different styles of annotations.
4.5.1 Generation

In the description generation task, the system is given a full image and a target object (specified via a bounding box), and it must generate a referring expression for the target object. Formally, the task is to compute \( \arg\max_S p(S|R, I) \), where \( S \) is a sentence, \( R \) is a region, and \( I \) is an image.

Since we will use RNNs to represent \( p(S|R, I) \), we can generate \( S \) one word at a time until we generate an end of sentence symbol. Computing the globally most probable sentence is hard, but we can use beam search to approximately find the most probable sentences (we use a beam size of 3). This is very similar to a standard image captioning task, except the input is a region instead of a full image. The main difference is that we will train our model to generate descriptions that distinguish the input region from other candidate regions.

4.5.2 Comprehension

In the description comprehension task, we are given a full image and a referring expression and are asked to localize the object being referred to within the image by returning a bounding box. One approach would be to train a model to directly predict the bounding box location given the referring expression (and image). However, in this chapter, we adopt a simpler, ranking-based approach. In particular, we first generate a set \( C \) of region proposals, and then ask the system to rank these by probability. Then we select the region using \( R^\ast = \arg\max_{R \in C} p(R|S, I) \), where, by Bayes’ rule, we have

\[
p(R|S, I) = \frac{p(S|R, I)p(R|I)}{\sum_{R' \in C} p(S|R', I)p(R'|I)}.
\]

If we assume a uniform prior for \( p(R|I) \)^2 we can select the region using \( R^\ast = \arg\max_{R \in C} p(S|R, I) \). This strategy is similar to image retrieval methods such as [KZS14, MXY15b], where the regions play the role of images.

At test time, we use the multibox method of [EST14] to generate objects proposals. This generates a large number of class agnostic bounding boxes. We then classify each box into one of the 80 MS-COCO categories, and discard those with low scores. We use the resulting post-

---

^2 This implies that we are equally likely to choose any region to describe. This is approximately true by virtue of the way we constructed the dataset. However, in real applications, region saliency \( p(R|I) \) should be taken into account.
classification boxes as the proposal set $C$. To get an upper bound on performance, we also use the ground truth bounding boxes for all the objects in the image. In both cases, we do not use the label for the object of interest when ranking proposals.

### 4.6 The Baseline Method

In this section we explain our baseline method for computing $p(S|R, I)$.

#### 4.6.1 Model Architecture

Our baseline model is similar to other image captioning models that use a CNN to represent the image, followed by an LSTM to generate the text (see e.g., [MXY15b, DHG14, VTB14]). The main difference is that we augment the CNN representation of the whole image with a CNN representation of the region of interest, in addition to location information. See Figure 4.4 for an illustration of our baseline model.

In more detail, we use VGGNet [SZ14] as our CNN, pre-trained on the ImageNet dataset [DDS09, KSH12]. The last 1000 dimensional layer of VGGNet is used as our representation of the object region. In addition, we compute features for the whole image, to serve as context. In experiments, we only fine-tuned the weights for the last layer of the CNN and fixed all other layers. To feed a region to the CNN, we keep the aspect ratio of the region fixed and scale it to $224 \times 224$ resolution, padding the margins with the mean pixel value (this is similar to the region warping strategy in [GDD14]). This gives us a 2000-dimensional feature vector, for the region and image.
We encode the relative location and size of the region using a 5-dimensional vector as follows:

\[
[x_{tl} \ W \ y_{tl} \ H \ x_{br} \ W \ y_{br} \ H \ S_{bbox} \ S_{image}],
\]

where \((x_{tl}, y_{tl})\) and \((x_{br}, y_{br})\) are the coordinates of the top left and bottom right corners of the object bounding box, \(H\) and \(W\) are height and width of the image, and \(S_{bbox}\) and \(S_{image}\) are the sizes of the bounding box and image respectively.

Concatenating with the region, image, and location/size features, we obtain a 2005-dimensional vector which we feed as input into an LSTM sequence model, which parameterizes the form of the distribution \(p(S|R, I)\). For our LSTMs, we use a 1024-dimensional word-embedding space, and 1024-dimensional hidden state vector. We adopt the most commonly used vanilla LSTM structure [GSK15] and feed the visual representation as input to the LSTM at each time step.

4.6.2 Maximum Likelihood Training

Our training data (discussed in Section 4.4) consists of observed triplets \((I, R, S)\), where \(I\) is an image, \(R\) denotes a region within \(I\), and \(S\) denotes a referring expression for \(R\). To train the baseline model, we minimize the negative log probability of the referring expressions given their respective region and image:

\[
J(\theta) = -\sum_{n=1}^{N} \log p(S_n|R_n, I_n, \theta),
\]

where \(\theta\) are the parameters of the RNN and CNN, and where we sum over the \(N\) examples in the training set. We use ordinary stochastic gradient decent with a batch size of 16 and use an initial learning rate of 0.01 which is halved every 50,000 iterations. Gradient norms are clipped to a maximum value of 10. To combat overfitting, we regularize using dropout with a ratio of 0.5 for both the word-embedding and output layers of the LSTM.

4.7 The Full Method

The baseline method is to train the model to maximize \(p(S|R, I)\), as is common for CNN-LSTM based image captioning models. However a strategy that directly generates an expression based only on the target object (which [GLK10] calls the reflex speaker strategy) has the drawback that it may fail to generate discriminative sentences. For example, consider Figure 4.4 to generate a
Figure 4.5: Illustration of how we train the full model using the softmax loss function. $R$ (green) is the target region, $R'$ are the incorrect regions. The weights of the LSTMs and CNNs are shared for $R$ and $R'$s. (Best viewed in color)

description of the girl highlighted by the green bounding box, generating the word “pink” is useful since it distinguishes this girl from the other girl on the right. To this end, we propose a modified training objective, described below.

### 4.7.1 Discriminative (MMI) Training

Section 4.6.2 proposed a way to train the model using maximum likelihood. We now propose the following alternative objective function:

$$J'(\theta) = -\sum_{n=1}^{N} \log p(R_n|S_n, I_n, \theta),$$

(4.3)

where

$$\log p(R_n|S_n, I_n, \theta) = \log \frac{p(S_n|R_n, I_n, \theta)}{\sum_{R' \in C(I_n)} p(S_n|R', I_n, \theta)}. \quad (4.4)$$

We will call this the softmax loss. Note that this is the same as maximizing the mutual information between $S$ and $R$ (assuming a uniform prior for $p(R)$), since

$$\text{MI}(S, R) = \log \frac{p(S, R)}{p(R)p(S)} = \log \frac{p(S|R)}{p(S)}. \quad (4.5)$$

where $p(S) = \sum_{R'} p(S|R')p(R') = \sum_{R'} p(S|R')$. Hence this approach is also called Maximum Mutual Information (MMI) training [BBS86].

The main intuition behind MMI training is that we want to consider whether a listener would interpret the sentence unambiguously. We do this by penalizing the model if it thinks that a referring
expression for a target object could also be plausibly generated by some other object within the same image. Thus given a training sample \((I, R, S)\), we train a model that outputs a high \(p(S | R, I)\), while maintaining a low \(p(S | R', I)\), whenever \(R' \neq R\). Note that this stands in contrast to the Maximum Likelihood (ML) objective function in Equation 4.2 which directly maximizes \(p(S|R)\) without considering other objects in the image.

There are several ways to select the region proposals \(C\). We could use all the true object bounding boxes, but this tends to waste time on objects that are visually very easy to discriminate from the target object (hence we call these “easy ground truth negatives”). An alternative is to select true object bounding boxes belonging to objects of the same class as the target object; these are more confusable (hence we call them “hard ground truth negatives”). Finally, we can use multibox proposals, the same as we use at test time, and select the ones with the same predicted object labels as \(R\) (hence we call them “hard multibox negatives”). We will compare these different methods in Section 4.9.2. We use 5 random negatives at each step, so that all the data for a given image fits into GPU memory.

To optimize Equation 4.3, we must replicate the network (using tied weights) for each region \(R' \in \mathcal{C}(I_n)\) (including the true region \(R_n\)), as shown in Figure 4.5. The resulting MMI trained model has exactly the same number of parameters as the ML trained model, and we use the same optimization and regularization strategy as in Section 4.6.2. Thus the only difference is the objective function.

For computational reasons, it is more convenient to use the following max-margin loss, which compares the target region \(R\) against a single random negative region \(R'\):

\[
J''(\theta) = -\sum_{n=1}^{N} \{ \log p(S_n | R_n, I_n, \theta) - \\
\lambda \max(0, M - \log p(S_n | R_n, I_n, \theta) + \log p(S_n | R'_n, I_n, \theta)) \} 
\]

(4.6)

This objective, which we call max-margin MMI (or MMI-MM) intuitively captures a similar effect as its softmax counterpart (MMI-SoftMax) and as we show in Section 4.9.2 yields similar results in practice. However, since the max-margin objective only compares two regions, the network must only be replicated twice. Consequently, less memory is used per sentence, allowing for more sentences to be loaded per minibatch which in turn helps in stabilizing the gradient.
4.8 Semi-supervised Training

Collecting referring expressions data can be expensive. In this section we discuss semi-supervised training of our full model by making use of bounding boxes that do not have descriptions, and thus are more ubiquitously available. Our main intuition for why a bounding box (region) $R$ can be useful even without an accompanying description is because it allows us to penalize our model during MMI training if it generates a sentence that it cannot itself decode to correctly recover $R$ (recall that MMI encourages $p(S|R, I)$ to be higher than $p(S|R', I)$, whenever $R' \neq R$).

In this semi-supervised setting, we consider a small dataset $D_{bb+txt}$ of images with bounding boxes and descriptions, together with a larger dataset $D_{bb}$ of images and bounding boxes, but without descriptions. We use $D_{bb+txt}$ to train a model (which we call model $G$) to compute $p(S|R, I)$. We then use this model $G$ to generate a set of descriptions for the bounding boxes in $D_{bb}$ (we call this new dataset $D_{bb+auto}$). We then retrain $G$ on $D_{bb+txt} \cup D_{bb+auto}$, in the spirit of bootstrap learning.

The above strategy suffers from the flaw that not all of the generated sentences are reliable, which may “pollute” the training set. To handle this, we train an ensemble of different models on $D_{bb+txt}$ (call them model $C$), and use these to determine which of the generated sentences for $D_{bb+auto}$ are trustworthy. In particular, we apply each model in the ensemble to decode each sentence in $D_{bb+auto}$, and only keep the sentence if every model maps it to the same correct object; we will call the resulting verified dataset $D_{filtered}$. This ensures that the generator creates referring expressions that can be understood by a variety of different models, thus minimizing overfitting.
See Figure 4.6 for an illustration. In the experiments, we show that our model benefits from this semi-supervised training.

4.9 Experiments

We conducted experiments on both of the COCO referring expression datasets mentioned in Section 4.4: our G-Ref dataset and the UNC-Ref dataset. We randomly chose 5,000 objects as the validation set, 5,000 objects as the testing set and the remaining objects as the training set (44,822 for G-Ref and 40,000 for UNC-Ref).

4.9.1 Evaluation Metrics

In this section, we describe how we evaluate performance of the comprehension and generation tasks.

The comprehension task is easy to evaluate: we simply compute the Intersection over Union (IoU) ratio between the true and predicted bounding box. If IoU exceeds 0.5, we call the detection a true positive, otherwise it is a false positive (this is equivalent to computing the precision@1 measure). We then average this score over all images.

The generation task is more difficult — we can evaluate a generated description in the same way as an image description, using metrics such as CIDEr [VLP15], BLEU [PRW02] and METEOR [LA07]. However these metrics can be unreliable and do not account for semantic meaning. We rely instead on human evaluation, as was done in the most recent image captioning competition [cap]. In particular, we asked Amazon Mechanical Turk (AMT) workers to compare an automatically generated object description to a human generated object description, when presented with an image and object of interest. The AMT workers do not know which sentences are human generated and which are computer generated (we do not even tell them that some sentences might be computer generated to reduce possible bias). We simply ask them to judge which sentence is a better description, or if they are equally good.

In addition to human evaluation, which does not scale, we evaluate our entire system by passing
<table>
<thead>
<tr>
<th>Proposals Descriptions</th>
<th>GT</th>
<th>Multibox</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GEN</td>
<td>GT</td>
</tr>
<tr>
<td>ML (baseline)</td>
<td>0.803</td>
<td>0.654</td>
</tr>
<tr>
<td>MMI-MM-easy-GT-neg</td>
<td>0.851</td>
<td>0.677</td>
</tr>
<tr>
<td>MMI-MM-hard-GT-neg</td>
<td><strong>0.857</strong></td>
<td><strong>0.699</strong></td>
</tr>
<tr>
<td>MMI-MM-multibox-neg</td>
<td>0.848</td>
<td>0.695</td>
</tr>
<tr>
<td>MMI-SoftMax</td>
<td>0.848</td>
<td>0.689</td>
</tr>
</tbody>
</table>

Table 4.1: We measure precision@1 on the UNC-Ref validation data. Each row is a different way of training the model. The columns show performance on ground truth or multibox proposals, and ground truth (human) or generated descriptions. Thus the columns with GT descriptions evaluate the performance of the comprehension system, and the columns with GEN descriptions evaluate (in an end-to-end way) the performance of the generation system.

automatically generated descriptions to our comprehension system, and verifying that they get correctly decoded to the original object of interest. This end-to-end test is automatic and much more reliable than standard image captioning metrics.

### 4.9.2 Comparing different training methods

In this section, we compare different ways of training our model: maximum likelihood training (the baseline method); max-margin loss with easy ground truth negatives (“MMI-MM-easy-GT-neg”); max-margin loss with hard ground truth negatives (“MMI-MM-hard-GT-neg”); max-margin loss with hard multibox negatives (“MMI-MM-multibox-neg”); softmax/MMI loss with hard multibox negatives (“MMI-SoftMax”). For each method, we consider using either ground truth or multibox proposals at test time. In addition, we consider both ground truth descriptions and generated descriptions.

In this experiment we treat UNC-Ref as a validation set to explore various algorithmic options and hyperparameter settings for MMI. Only after having fixed these algorithmic options and
hyperparameter settings did we do experiments on our G-Ref dataset (Section 4.9.3). This reduces the risk that we will have “overfit” our hyperparameters to each particular dataset. The results are summarized in Table 4.1 and we draw the following conclusions:

- All models perform better on generated descriptions than the groundtruth ones, possibly because the generated descriptions are shorter than the groundtruth (5.99 words on average vs 8.43), and/or because the generation and comprehension models share the same parameters, so that even if the generator uses a word incorrectly (e.g., describing a “dog” as a “cat”), the comprehension system can still decode it correctly. Intuitively, a model might “communicate” better with itself using its own language than with others.

- All the variants of the Full model (using MMI training) work better than the strong baseline using maximum likelihood training.

- The softmax version of MMI training is similar to the max-margin method, but slightly worse.

- MMI training benefits more from hard negatives than easy ones.

- Training on ground truth negatives helps when using ground truth proposals, but when using multibox proposals (which is what we can use in practice), it is better to use multibox negatives.

Based on the above results, for the rest of the paper we will use max-margin training with hard multibox negatives as our Full Model.

4.9.3 Fully-supervised Training

In this section, we compare the strong baseline (maximum likelihood) with our max-margin MMI method on the validation and test sets from G-Ref and UNC-Ref. As before, we consider ground truth and multibox proposals at test time, and ground truth (human) or generated (automatic) descriptions. The results are shown in Table 4.2. We see that MMI training outperforms ML training under every setting.

In addition to the above end-to-end evaluation, we use human evaluators to judge generated
descriptions. We also train our baseline and full model on a random train, val, and test split w.r.t. to the images of our G-Ref dataset. The results are consistent with those in Table 4.2. With multibox proposals and GT descriptions, the Precision @ 1 of the baseline and full model are 0.404 and 0.444 on val set, and 0.407 and 0.451 on test set respectively.
<table>
<thead>
<tr>
<th>Proposals</th>
<th>GT</th>
<th>multibox</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descriptions</td>
<td>GEN</td>
<td>GT</td>
</tr>
</tbody>
</table>

| G-Ref-Val | Baseline | 0.751 | 0.579 | 0.468 | 0.425 |
| Full Model | **0.799** | **0.607** | **0.500** | **0.445** |

| G-Ref-Test | Baseline | 0.769 | 0.545 | 0.485 | 0.406 |
| Full Model | **0.811** | **0.606** | **0.513** | **0.446** |

| UNC-Ref-Val | Baseline | 0.803 | 0.654 | 0.564 | 0.478 |
| Full Model | **0.848** | **0.695** | **0.604** | **0.511** |

| UNC-Ref-Test | Baseline | 0.834 | 0.643 | 0.596 | 0.477 |
| Full Model | **0.851** | **0.700** | **0.603** | **0.518** |

Table 4.2: Precision@1 for the baseline (ML) method and our full model with the max-margin objective function on various datasets.

In particular, we selected 1000 objects at random from our test set, and showed them to Amazon Mechanical Turk workers. The percentage of descriptions that are evaluated as better or equal to a human caption for the baseline and the full model are 15.9% and 20.4% respectively. This shows that MMI training is much better (4.5% absolute improvement, and 28.5% relative) than ML training.

### 4.9.4 Semi-supervised Training

To conduct the semi-supervised training experiment, we separate the training set of our G-Ref dataset and the UNC-Ref dataset into two parts with the same number of objects. The first part (denoted by $D_{bb+txt}$) has the object description annotations while the second part (denoted by $D_{bb}$) only has object bounding boxes. Table 4.3 shows the results of semi-supervised training on the
Table 4.3: Performance of our full model when trained on a small strongly labeled dataset vs training on a larger dataset with automatically labeled data.

validation set of our dataset and UNC-Ref. We see that we get some improvement by training on \(D_{bb+txt} \cup D_{bb}\) over just using \(D_{bb+txt}\).

### 4.9.5 Qualitative Results

figure 4.7: The sample results of the description generation using our full model (above the dashed line) and the strong baseline (below the dashed line). The descriptions generated by our full model are more discriminative than those generated by the baseline.

In Figure 4.7 we show qualitative results of our full generation model (above the dashed line) and the baseline generation model (below the dashed line) on some of our test images. We see that the descriptions generated by our full model are typically longer and more discriminative than the baseline model. In the second image, for example, the baseline describes one of the cats as “a cat laying on a bed”, which is not sufficiently unambiguous for a listener to understand which cat is being described. Our full model, on the other hand, describes the same cat as “a cat laying on the
left” which is completely unambiguous.

Figure 4.8 shows some qualitative results of our full comprehension model on our test dataset. The first and second columns show the original image and the multibox proposals respectively. The last four columns show the bounding boxes (denoted as a red bounding box in the figure) selected by our full model in response to different input sentences (both ground truth sentences and ones we created to probe the comprehension abilities of the model). To better interpret these results, we also show the bounding boxes that are within the margin of the model (see Eqn. 4.6) with dashed blue bounding boxes. Their bounding boxes are considered as “possible candidates” but their scores (i.e. \( p(S|R, I) \)) are not as high as the chosen one.

In general, we see that the comprehension model does quite well from short two word phrases to longer descriptions. It is able to respond correctly to single word changes in a referring expression (e.g., “the man in black” to “the man in red”). It also correctly identifies that the horse is the referent of the expression “a dark horse carrying a woman” whereas the woman is the referent in “a woman on the dark horse” — note that methods that average word embeddings would most likely fail on this example. However, there are also failure cases. For example, in the fifth row, “the woman in white” selects a woman in black; this is because our model cannot handle the case where the object is not present, although it makes a reasonable guess. Also, in the fifth row, “the controller in the woman’s hand” selects the woman, the orange juice and the controller, since this particular kind of object is too small to detect, and lacks enough training data.

### 4.10 Conclusions

To conclude, we leave the reader with two simple points. First, referring expressions have been studied for decades, but in light of the recent burst of interest in image captioning, referring expressions take on new importance. Where image captioning itself is difficult to evaluate, referring expressions have an objective performance metric, and require the same semantic understanding of language and vision. Thus success on datasets such as the one contributed in this chapter is more meaningful than success by standard image captioning metrics.

Second, to be successful at generating descriptions, we must consider the listener. Our experi-
ments show that modeling a listener that must correctly decode a generated description consistently outperforms a model that simply emits captions based on region features. We hope that in addition to our dataset, these insights will spur further progress on joint models of vision and language.
Figure 4.8: Sample results of the description comprehension task using our full model. The first and second column shows the original image and the multibox proposals. The third to sixth columns show the results of our model when input an arbitrary description of an object in the image. The red bounding box denotes the most probable object predicted by the model while the blue dashed ones denote the bounding boxes within the margin of the most probable one. The descriptions can be the groundtruth ones in the dataset (third column) or an customized descriptions (fourth to sixth columns). (Best viewed in color)
CHAPTER 5

Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering [GMZ15]

5.1 Abstract

In this paper, we present the mQA model, which is able to answer questions about the content of an image. The answer can be a sentence, a phrase or a single word. Our model contains four components: a Long Short-Term Memory (LSTM) to extract the question representation, a Convolutional Neural Network (CNN) to extract the visual representation, an LSTM for storing the linguistic context in an answer, and a fusing component to combine the information from the first three components and generate the answer. We construct a Freestyle Multilingual Image Question Answering (FM-IQA) dataset to train and evaluate our mQA model. It contains over 150,000 images and 310,000 freestyle Chinese question-answer pairs and their English translations. The quality of the generated answers of our mQA model on this dataset is evaluated by human judges through a Turing Test. Specifically, we mix the answers provided by humans and our model. The human judges need to distinguish our model from the human. They will also provide a score (i.e. 0, 1, 2, the larger the better) indicating the quality of the answer. We propose strategies to monitor the quality of this evaluation process. The experiments show that in 64.7% of cases, the human judges cannot distinguish our model from humans. The average score is 1.454 (1.918 for human). The details of this work, including the FM-IQA dataset, can be found on the project page: http://idl.baidu.com/FM-IQA.html.
5.2 Introduction

Recently, there is increasing interest in the field of multimodal learning for both natural language and vision. In particular, many studies have made rapid progress on the task of image captioning \cite{MX14,KS14,KF14,VTB14,DHG14,FG14,CZ14,LPC14,KLS14,XBK15}. Most of them are built based on deep neural networks (e.g. deep Convolutional Neural Networks (CNN \cite{KSH12}), Recurrent Neural Network (RNN \cite{Elm90}) or Long Short-Term Memory (LSTM \cite{HS97})). The large-scale image datasets with sentence annotations (e.g., \cite{LMB14,YLH14a,GCM06}) play a crucial role in this progress. Despite the success of these methods, there are still many issues to be discussed and explored. In particular, the task of image captioning only requires generic sentence descriptions of an image. But in many cases, we only care about a particular part or object of an image. The image captioning task lacks the interaction between the computer and the user (as we cannot input our preference and interest).

In this paper, we focus on the task of visual question answering. In this task, the method needs to provide an answer to a freestyle question about the content of an image. We propose the mQA model to address this task. The inputs of the model are an image and a question. This model has four components (see Figure 5.2). The first component is an LSTM network that encodes a natural language sentence into a dense vector representation. The second component is a deep Convolutional Neural Network \cite{SLJ14} that extracted the image representation. This component was pre-trained on ImageNet Classification Task \cite{RDS14} and is fixed during the training. The third component is another LSTM network that encodes the information of the current word and previous words in the answer into dense representations. The fourth component fuses the information from the first three components to predict the next word in the answer. We jointly train the first, third and fourth components by maximizing the probability of the groundtruth answers in the training set using a log-likelihood loss function. To lower down the risk of overfitting, we allow the weight sharing of the word embedding layer between the LSTMs in the first and third components. We also adopt the transposed weight sharing scheme as proposed in \cite{MWY15}, which allows the weight sharing between word embedding layer and the fully connected Softmax layer.

To train our method, we construct a large-scale Freestyle Multilingual Image Question Answer-
The current version of the dataset contains 158,392 images with 316,193 Chinese question-answer pairs and their corresponding English translations. The results reported in this paper are obtained from a model trained on the first version of the dataset (a subset of the current version) which contains 120,360 images and 250,569 question-answer pairs. To diversify the annotations, the annotators are allowed to raise any question related to the content of the image. We propose strategies to monitor the quality of the annotations. This dataset contains a wide range of AI related questions, such as action recognition (e.g., “Is the man trying to buy vegetables?”), object recognition (e.g., “What is there in yellow?”), positions and interactions among objects in the image (e.g. “Where is the kitty?”) and reasoning based on commonsense and visual content (e.g. “Why does the bus park here?”), see last column of Figure 5.3.

Because of the variability of the freestyle question-answer pairs, it is hard to accurately evaluate the method with automatic metrics. We conduct a Visual Turing Test [Tur50] using human judges. Specifically, we mix the question-answer pairs generated by our model with the same set of question-answer pairs labeled by annotators. The human judges need to determine whether the answer is given by a model or a human. In addition, we also ask them to give a score of 0 (i.e. wrong), 1 (i.e. partially correct), or 2 (i.e. correct). The results show that our mQA model passes 64.7% of this test (treated as answers of a human) and the average score is 1.454. In the discussion, we analyze the failure cases of our model and show that combined with the m-RNN [MXY15b] model, our model can automatically ask a question about an image and answer that question.

5.3 Related Work

Recent work has made significant progress using deep neural network models in both the fields of computer vision and natural language. For computer vision, methods based on Convolutional Neural Network (CNN [LBO12]) achieve the state-of-the-art performance in various tasks, such as object classification [KSH12, SZ14, KSH12], detection [GDD14, ZMY14] and segmentation

\[1\text{We are actively developing and expanding the dataset, please find the latest information on the project page : http://idl.baidu.com/FM-IQA.html}\]
Figure 5.1: Sample answers to the visual question generated by our model on the newly proposed Freestyle Multilingual Image Question Answering (FM-IQA) dataset.

[CPK15]. For natural language, the Recurrent Neural Network (RNN [Elm90, MJC14]) and the Long Short-Term Memory network (LSTM [HS97]) are also widely used in machine translation [KB13, CMG14, SVL14] and speech recognition [MKB10].

The structure of our mQA model is inspired by the m-RNN model [MXY15b] for the image captioning and image-sentence retrieval tasks. It adopts a deep CNN for vision and a RNN for language. We extend the model to handle the input of question and image pairs, and generate answers. In the experiments, we find that we can learn how to ask a good question about an image using the m-RNN model and this question can be answered by our mQA model.

There has been recent effort on the visual question answering task [GGH15, BJJ10, MF14a, TML14]. However, most of them use a pre-defined and restricted set of questions. Some of these questions are generated from a template. In addition, our FM-IQA dataset is much larger than theirs (e.g., there are only 2591 and 1449 images for [GGH15] and [MF14a] respectively).

There are some concurrent and independent works on this topic: [AAL15, MRF15, RKZ15]. [AAL15] propose a large-scale dataset also based on MS COCO. They also provide some simple baseline methods on this dataset. Compared to them, we propose a stronger model for this task and evaluate our method using human judges. Our dataset also contains two different kinds of language, which can be useful for other tasks, such as machine translation. Because we use a different set of annotators and different requirements of the annotation, our dataset and the [AAL15] can be complementary to each other, and lead to some interesting topics, such as dataset transferring for visual question answering.
What is the cat doing?

Sitting on the umbrella.

Figure 5.2: Illustration of the mQA model architecture. We input an image and a question about the image (i.e. “What is the cat doing?”) to the model. The model is trained to generate the answer to the question (i.e. “Sitting on the umbrella”). The weight matrix in the word embedding layers of the two LSTMs (one for the question and one for the answer) are shared. In addition, as in [MWY15], this weight matrix is also shared, in a transposed manner, with the weight matrix in the Softmax layer. Different colors in the figure represent different components of the model. (Best viewed in color.)

Both [MRF15] and [RKZ15] use a model containing a single LSTM and a CNN. They concatenate the question and the answer (for [RKZ15], the answer is a single word. [MRF15] also prefer a single word as the answer), and then feed them to the LSTM. Different from them, we use two separate LSTMs for questions and answers respectively in consideration of the different properties (e.g. grammar) of questions and answers, while allow the sharing of the word-embeddings. For the dataset, [MRF15] adopt the dataset proposed in [MF14a], which is much smaller than our FM-IQA dataset. [RKZ15] utilize the annotations in MS COCO and synthesize a dataset with four pre-defined types of questions (i.e. object, number, color, and location). They also synthesize the answer with a single word. Their dataset can also be complementary to ours.
5.4 The Multimodal QA (mQA) Model

We show the architecture of our mQA model in Figure 5.2. The model has four components: (I). a Long Short-Term Memory (LSTM [HS97]) for extracting semantic representation of a question, (II). a deep Convolutional Neural Network (CNN) for extracting the image representation, (III). an LSTM to extract representation of the current word in the answer and its linguistic context, and (IV). a fusing component that incorporates the information from the first three parts together and generates the next word in the answer. These four components can be jointly trained together. The details of the four model components are described in Section 5.4.1. The effectiveness of the important components and strategies are analyzed in Section 5.6.3.

The inputs of the model are a question and the reference image. The model is trained to generate the answer. The words in the question and answer are represented by one-hot vectors (i.e. binary vectors with the length of the dictionary size $N$ and have only one non-zero vector indicating its index in the word dictionary). We add a ⟨BOA⟩ sign and a ⟨EOA⟩ sign, as two spatial words in the word dictionary, at the beginning and the end of the training answers respectively. They will be used for generating the answer to the question in the testing stage.

In the testing stage, we input an image and a question about the image into the model first. To generate the answer, we start with the start sign ⟨BOA⟩ and use the model to calculate the probability distribution of the next word. We then use a beam search scheme that keeps the best $K$ candidates with the maximum probabilities according to the Softmax layer. We repeat the process until the model generates the end sign of the answer ⟨BOA⟩.

5.4.1 The Four Components of the mQA Model

(I). The semantic meaning of the question is extracted by the first component of the model. It contains a 512 dimensional word embedding layer and an LSTM layer with 400 memory cells. The function of the word embedding layer is to map the one-hot vector of the word into a dense semantic space. We feed this dense word representation into the LSTM layer.

In practice, we fix the CNN part because the gradient returned from LSTM is very noisy. Finetuning the CNN takes a much longer time than just fixing it, and does not improve the performance significantly.
LSTM [HS97] is a Recurrent Neural Network [Elm90] that is designed for solving the gradient explosion or vanishing problem. The LSTM layer stores the context information in its memory cells and serves as the bridge among the words in a sequence (e.g. a question). To model the long term dependency in the data more effectively, LSTM add three gate nodes to the traditional RNN structure: the input gate, the output gate and the forget gate. The input gate and output gate regulate the read and write access to the LSTM memory cells. The forget gate resets the memory cells when their contents are out of date. Different from [MRF15, RKZ15], the image representation does not feed into the LSTM in this component. We believe this is reasonable because questions are just another input source for the model, so we should not add images as the supervision for them. The information stored in the LSTM memory cells of the last word in the question (i.e. the question mark) will be treated as the representation of the sentence.

(II). The second component is a deep Convolutional Neural Network (CNN) that generates the representation of an image. In this paper, we use the GoogleNet [SLJ14]. Note that other CNN models, such as AlexNet [KSH12] and VggNet [SZ14], can also be used as the component in our model. We remove the final SoftMax layer of the deep CNN and connect the remaining top layer to our model.

(III). The third component also contains a word embedding layer and an LSTM. The structure is similar to the first component. The activation of the memory cells for the words in the answer, as well as the word embeddings, will be fed into the fusing component to generate the next words in the answer.

In [MRF15, RKZ15], they concatenate the training question and answer, and use a single LSTM. Because of the different properties (i.e. grammar) of question and answer, in this paper, we use two separate LSTMs for questions and answers respectively. We denote the LSTMs for the question and the answer as LSTM(Q) and LSTM(A) respectively in the rest of the paper. The weight matrix in LSTM(Q) is not shared with the LSTM(A) in the first components. Note that the semantic meaning of single words should be the same for questions and answers so that we share the parameters in the word-embedding layer for the first and third component.

(IV). Finally, the fourth component fuses the information from the first three layers. Specifically,
the activation of the fusing layer $f(t)$ for the $t^{th}$ word in the answer can be calculated as follows:

$$f(t) = g(V_{rQ}r_Q + V_I I + V_{rA}r_A(t) + V_w w(t));$$ (5.1)

where “+” denotes element-wise addition, $r_Q$ stands for the activation of the LSTM(Q) memory cells of the last word in the question, $I$ denotes the image representation, $r_A(t)$ and $w(t)$ denotes the activation of the LSTM(A) memory cells and the word embedding of the $t^{th}$ word in the answer respectively. $V_{rQ}$, $V_I$, $V_{rA}$, and $V_w$ are the weight matrices that need to be learned. $g(\cdot)$ is an element-wise non-linear function.

After the fusing layer, we build an intermediate layer that maps the dense multimodal representation in the fusing layer back to the dense word representation. We then build a fully connected Softmax layer to predict the probability distribution of the next word in the answer. This strategy allows the weight sharing between word embedding layer and the fully connected Softmax layer as introduced in [MWY15] (see details in Section 5.4.2).

Similar to [MWY15], we use the sigmoid function as the activation function of the three gates and adopt ReLU [NH10] as the non-linear function for the LSTM memory cells. The non-linear activation function for the word embedding layer, the fusing layer and the intermediate layer is the scaled hyperbolic tangent function [LBO12]: $g(x) = 1.7159 \cdot \tanh(\frac{2}{3}x)$.

### 5.4.2 The Weight Sharing Strategy

As mentioned in Section 5.2 our model adopts different LSTMs for the question and the answer because of the different grammar properties of questions and answers. However, the meaning of single words in both questions and answers should be the same. Therefore, we share the weight matrix between the word-embedding layers of the first component and the third component.

In addition, this weight matrix for the word-embedding layers is shared with the weight matrix in the fully connected Softmax layer in a transposed manner. Intuitively, the function of the weight matrix in the word-embedding layer is to encode the one-hot word representation into a dense word representation. The function of the weight matrix in the Softmax layer is to decode the dense word representation into a pseudo one-word representation, which is the inverse operation of the word-embedding. This strategy will reduce nearly half of the parameters in the model and is shown
to have better performance in image captioning and novel visual concept learning tasks [MWY15].

5.4.3 Training Details

The CNN we used is pre-trained on the ImageNet classification task [RDS14]. This component is fixed during the QA training. We adopt a log-likelihood loss defined on the word sequence of the answer. Minimizing this loss function is equivalent to maximizing the probability of the model to generate the groundtruth answers in the training set. We jointly train the first, second and the fourth components using stochastic gradient decent method. The initial learning rate is 1 and we decrease it by a factor of 10 for every epoch of the data. We stop the training when the loss on the validation set does not decrease within three epochs. The hyperparameters of the model are selected by cross-validation.

For the Chinese question answering task, we segment the sentences into several word phrases. These phrases can be treated equivalently to the English words.

5.5 The Freestyle Multilingual Image Question Answering (FM-IQA) Dataset

Our method is trained and evaluated on a large-scale multilingual visual question answering dataset. In Section 5.5.1 we will describe the process to collect the data, and the method to monitor the quality of annotations. Some statistics and examples of the dataset will be given in Section 5.5.2. The latest dataset is available on the project page: [http://idl.baidu.com/FM-IQA.html](http://idl.baidu.com/FM-IQA.html)

5.5.1 The Data Collection

We start with the 158,392 images from the newly released MS COCO [LMB14] training, validation and testing set as the initial image set. The annotations are collected using Baidu’s online crowdsourcing server[^3]. To make the labeled question-answer pairs diversified, the annotators are free to give any type of questions, as long as these questions are related to the content of the image. The question should be answered by the visual content and commonsense (e.g., we are not expecting to

[^3]: [http://test.baidu.com](http://test.baidu.com)
get questions such as “What is the name of the person in the image?”). The annotators need to give an answer to the question themselves.

On the one hand, the freedom we give to the annotators is beneficial in order to get a freestyle, interesting and diversified set of questions. On the other hand, it makes it harder to control the quality of the annotation compared to a more detailed instruction. To monitor the annotation quality, we conduct an initial quality filtering stage. Specifically, we randomly sampled 1,000 images as a quality monitoring dataset from the MS COCO dataset as an initial set for the annotators (they do not know this is a test). We then sample some annotations and rate their quality after each annotator finishes some labeling on this quality monitoring dataset (about 20 question-answer pairs per annotator). We only select a small number of annotators (195 individuals) whose annotations are satisfactory (i.e. the questions are related to the content of the image and the answers are correct). We also give preference to the annotators who provide interesting questions that require high level reasoning to give the answer. Only the selected annotators are permitted to label the rest of the images. We pick a set of good and bad examples of the annotated question-answer pairs from the quality monitoring dataset, and show them to the selected annotators as references. We also provide reasons for selecting these examples. After the annotation of all the images is finished, we further refine the dataset and remove a small portion of the images with badly labeled questions and answers.

5.5.2 The Statistics of the Dataset

Currently there are 158,392 images with 316,193 Chinese question-answer pairs and their English translations. Each image has at least two question-answer pairs as annotations. The average lengths of the questions and answers are 7.38 and 3.82 respectively measured by Chinese words. Some sample images are shown in Figure 5.3. We randomly sampled 1,000 question-answer pairs and their corresponding images as the test set.

The questions in this dataset are diversified, which requires a vast set of AI capabilities in order to answer them. They contain some relatively simple image understanding questions of, e.g., the actions of objects (e.g., “What is the boy in green cap doing?”), the object class (e.g., “Is there any
Figure 5.3: Sample images in the FM-IQA dataset. This dataset contains 316,193 Chinese question-answer pairs with corresponding English translations.

The answers are also diversified. The annotators are allowed to give a single phrase or a single word as the answer (e.g. “Yellow”) or, they can give a complete sentence (e.g. “The frisbee is yellow”).
5.6 Experiments

For the very recent works for visual question answering ([RKZ15, MRF15]), they test their method on the datasets where the answer of the question is a single word or a short phrase. Under this setting, it is plausible to use automatic evaluation metrics that measure the single word similarity, such as Wu-Palmer similarity measure (WUPS) [WP94]. However, for our newly proposed dataset, the answers in the dataset are freestyle and can be complete sentences. For most of the cases, there are numerous choices of answers that are all correct. The possible alternatives are BLEU score [PRW02], METEOR [LA07], CIDEr [VZP14] or other metrics that are widely used in the image captioning task [MXY15b]. The problem of these metrics is that there are only a few words in an answer that are semantically critical. These metrics tend to give equal weights (e.g. BLEU and METEOR) or different weights according to the tf-idf frequency term (e.g. CIDEr) of the words in a sentence, hence cannot fully show the importance of the keywords. The evaluation of the image captioning task suffers from the same problem (not as severe as question answering because it only needs a general description).

To avoid these problems, we conduct a real Visual Turing Test using human judges for our model, which will be described in details in Section 5.6.1. In addition, we rate each generated sentences with a score (the larger the better) in Section 5.6.2, which gives a more fine-grained evaluation of our method. In Section 5.6.3 we provide the performance comparisons of different variants of our mQA model on the validation set.

5.6.1 The Visual Turing Test

In this Visual Turing Test, a human judge will be presented with an image, a question and the answer to the question generated by the testing model or by human annotators. He or she need to determine, based on the answer, whether the answer is given by a human (i.e. pass the test) or a machine (i.e. fail the test).

In practice, we use the images and questions from the test set of our FM-IQA dataset. We use our mQA model to generate the answer for each question. We also implement a baseline model of the question answering without visual information. The structure of this baseline model is similar
to mQA, except that we do not feed the image information extracted by the CNN into the fusing layer. We denote it as blind-QA. The answers generated by our mQA model, the blind-QA model and the groundtruth answer are mixed together. This leads to 3000 question answering pairs with the corresponding images, which will be randomly assigned to 12 human judges.

The results are shown in Table 5.1. It shows that 64.7% of the answers generated by our mQA model are treated as answers provided by a human. The blind-QA performs very badly in this task. But some of the generated answers pass the test. Because some of the questions are actually multi-choice questions, it is possible to get a correct answer by random guess based on pure linguistic clues.

To study the variance of the VTT evaluation across different sets of human judges, we conduct two additional evaluations with different groups of judges under the same setting. The standard deviations of the passing rate are 0.013, 0.019 and 0.024 for human, the blind-mQA model and mQA model respectively. It shows that VTT is a stable and reliable evaluation metric for this task.

### 5.6.2 The Score of the Generated Answer

The Visual Turing Test only gives a rough evaluation of the generated answers. We also conduct a fine-grained evaluation with scores of “0”, “1”, or “2”. “0” and “2” mean that the answer is totally wrong and perfectly correct respectively. “1” means that the answer is only partially correct (e.g., the general categories are right but the sub-categories are wrong) and makes sense to the human judges. The human judges for this task are not necessarily the same people for the Visual Turing Test. After collecting the results, we find that some human judges also rate an answer with “1” if the

<table>
<thead>
<tr>
<th></th>
<th>Visual Turing Test</th>
<th>Human Rated Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pass</td>
<td>Fail</td>
</tr>
<tr>
<td>Human</td>
<td>948</td>
<td>52</td>
</tr>
<tr>
<td>blind-QA</td>
<td>340</td>
<td>660</td>
</tr>
<tr>
<td>mQA</td>
<td>647</td>
<td>353</td>
</tr>
</tbody>
</table>

Table 5.1: The results of our mQA model for our FM-IQA dataset.
question is very hard to answer so that even a human, without carefully looking at the image, will possibly make mistakes. We show randomly sampled images whose scores are “1” in Figure 5.4.

The results are shown in Table 5.1. We show that among the answers that are not perfectly correct (i.e. scores are not 2), over half of them are partially correct. Similar to the VTT evaluation process, we also conduct two additional groups of this scoring evaluation. The standard deviations of human and our mQA model are 0.020 and 0.041 respectively. In addition, for 88.3% and 83.9% of the cases, the three groups give the same score for human and our mQA model respectively.

5.6.3 Performance Comparisons of the Different mQA Variants

In order to show the effectiveness of the different components and strategies of our mQA model, we implement three variants of the mQA in Figure 5.2. For the first variant (i.e. “mQA-avg-question”), we replace the first LSTM component of the model (i.e. the LSTM to extract the question embedding)

<table>
<thead>
<tr>
<th></th>
<th>Word Error</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>mQA-avg-question</td>
<td>0.442</td>
<td>2.17</td>
</tr>
<tr>
<td>mQA-same-LSTMs</td>
<td>0.439</td>
<td>2.09</td>
</tr>
<tr>
<td>mQA-noTWS</td>
<td>0.438</td>
<td>2.14</td>
</tr>
<tr>
<td>mQA-complete</td>
<td><strong>0.393</strong></td>
<td><strong>1.91</strong></td>
</tr>
</tbody>
</table>

Table 5.2: Performance comparisons of the different mQA variants.

Figure 5.4: Random examples of the answers generated by the mQA model with score “1” given by the human judges.
Figure 5.5: The sample generated questions by our model and their answers.

with the average embedding of the words in the question using word2vec [MSC13]. It is used
to show the effectiveness of the LSTM as a question embedding learner and extractor. For the
second variant (i.e. “mQA-same-LSTMs”), we use two shared-weights LSTMs to model question
and answer. It is used to show the effectiveness of the decoupling strategy of the weights of the
LSTM(Q) and the LSTM(A) in our model. For the third variant (i.e. “mQA-noTWS”), we do not
adopt the Transposed Weight Sharing (TWS) strategy. It is used to show the effectiveness of TWS.

The word error rates and losses of the three variants and the complete mQA model (i.e. mQA-
complete) are shown in Table 5.2. All of the three variants performs worse than our mQA model.

5.7 Discussion

In this paper, we present the mQA model, which is able to give a sentence or a phrase as the answer
to a freestyle question for an image. To validate the effectiveness of the method, we construct

Figure 5.6: Failure cases of our mQA model on the FM-IQA dataset.
a Freestyle Multilingual Image Question Answering (FM-IQA) dataset containing over 310,000 question-answer pairs. We evaluate our method using human judges through a real Turing Test. It shows that 64.7% of the answers given by our mQA model are treated as the answers provided by a human. The FM-IQA dataset can be used for other tasks, such as visual machine translation, where the visual information can serve as context information that helps to remove ambiguity of the words in a sentence.

We also modified the LSTM in the first component to the multimodal LSTM shown in [MWY15]. This modification allows us to generate a free-style question about the content of image, and provide an answer to this question. We show some sample results in Figure 5.5.

We show some failure cases of our model in Figure 5.6. The model sometimes makes mistakes when the commonsense reasoning through background scenes is incorrect (e.g., for the image in the first column, our method says that the man is surfing but the small yellow frisbee in the image indicates that he is actually trying to catch the frisbee. It also makes mistakes when the targeting object that the question focuses on is too small or looks very similar to other objects (e.g. images in the second and fourth column). Another interesting example is the image and question in the fifth column of Figure 5.6. Answering this question is very hard since it needs high level reasoning based on the experience from everyday life. Our model outputs a ⟨OOV⟩ sign, which is a special word we use when the model meets a word which it has not seen before (i.e. does not appear in its word dictionary).

In future work, we will try to address these issues by incorporating more visual and linguistic information (e.g. using object detection or using attention models).
CHAPTER 6

Training and Evaluating Multimodal Word Embeddings with Large-scale Web Annotated Images [MXJ16]

6.1 Abstract

In this chapter, we focus on training and evaluating effective word embeddings with both text and visual information. More specifically, we introduce a large-scale dataset with 300 million sentences describing over 40 million images crawled and downloaded from publicly available Pins (i.e. an image with sentence descriptions uploaded by users) on Pinterest [pin]. This dataset is more than 200 times larger than MS COCO [LMB14], the standard large-scale image dataset with sentence descriptions. In addition, we construct an evaluation dataset to directly assess the effectiveness of word embeddings in terms of finding semantically similar or related words and phrases. The word/phrase pairs in this evaluation dataset are collected from the click data with millions of users in an image search system, thus contain rich semantic relationships. Based on these datasets, we propose and compare several Recurrent Neural Networks (RNNs) based multimodal (text and image) models. Experiments show that our model benefits from incorporating the visual information into the word embeddings, and a weight sharing strategy is crucial for learning such multimodal embeddings. The project page is: [http://www.stat.ucla.edu/~junhua_mao/multimodal_embedding.html](http://www.stat.ucla.edu/~junhua_mao/multimodal_embedding.html)\(^1\)

\(^1\)The datasets introduced in this work will be gradually released on the project page.
6.2 Introduction

Word embeddings are dense vector representations of words with semantic and relational information. In this vector space, semantically related or similar words should be close to each other. A large-scale training dataset with billions of words is crucial to train effective word embedding models. The trained word embeddings are very useful in various tasks and real-world applications that involve searching for semantically similar or related words and phrases.

A large proportion of the state-of-the-art word embedding models are trained on pure text data only. Since one of the most important functions of language is to describe the visual world, we argue that the effective word embeddings should contain rich visual semantics. Previous work has shown that visual information is important for training effective embedding models. However, due to the lack of large training datasets of the same scale as the pure text dataset, the models are either trained on relatively small datasets (e.g. [HK14]), or the visual constraints are only applied to limited number of pre-defined visual concepts (e.g. [LPB15]). Therefore, such work did not fully explore the potential of visual information in learning word embeddings.

In this chapter, we introduce a large-scale dataset with both text descriptions and images, crawled and collected from Pinterest, one of the largest database of annotated web images. On Pinterest, users save web images onto their boards (i.e. image collectors) and supply their descriptions of the images. More descriptions are collected when the same images are saved and commented by other users. Compared to MS COCO (i.e. the image benchmark with sentences descriptions [LMB14]), our dataset is much larger (40 million images with 300 million sentences compared to 0.2 million images and 1 million sentences in the current release of MS COCO) and is at the same scale as the standard pure text training datasets (e.g. Wikipedia Text Corpus). Some sample images and their descriptions are shown in Figure 6.1 in Section 6.4.1. We believe training on this large-scale dataset will lead to richer and better generalized models. We denote this dataset as the Pinterest40M dataset.

One challenge for word embeddings learning is how to directly evaluate the quality of the model with respect to the tasks (e.g. the task of finding related or similar words and phrases). State-of-the-art neural language models often use the negative log-likelihood of the predicted
words as their training loss, which is not always correlated with the effectiveness of the learned embedding. Current evaluation datasets (e.g. [BTB14, HRK15, FGM11]) for word similarity or relatedness contain only less than a thousand word pairs and cannot comprehensively evaluate all the embeddings of the words appearing in the training set.

The challenge of constructing large-scale evaluation datasets is partly due to the difficulty of finding a large number of semantically similar or related word/phrase pairs. In this chapter, we utilize user click information collected from Pinterest’s image search system to generate millions of these candidate word/phrase pairs. Because user click data are somewhat noisy, we removed inaccurate entries in the dataset by using crowdsourcing human annotations. This led to a final gold standard evaluation dataset consists of 10,674 entries.

Equipped with these datasets, we propose, train and evaluate several Recurrent Neural Network (RNN [Elm90]) based models with input of both text descriptions and images. Some of these models directly minimize the Euclidean distance between the visual features and the word embeddings or RNN states, similar to previous work (e.g. [HK14, LPB15]). The best performing model is inspired by recent image captioning models [DAG15, MXY15a, VTB15], with the additional weight-sharing strategy originally proposed in [MWY15] to learn novel visual concepts. This strategy imposes soft constraints between the visual features and all the related words in the sentences. Our experiments validate the effectiveness and importance of incorporating visual information into the learned word embeddings.

We make three major contributions: Firstly, we constructed a large-scale multimodal dataset with both text descriptions and images, which is at the same scale as the pure text training set. Secondly, we collected and labeled a large-scale evaluation dataset for word and phrase similarity and relatedness evaluation. Finally, we proposed and compared several RNN based models for learning multimodal word embeddings effectively. To facilitate research in this area, we will gradually release the datasets proposed in this chapter on our project page.
6.3 Related Work

Image-Sentence Description Datasets The image descriptions datasets, such as Flickr8K [HYH13], Flickr30K [YLH14b], IAPR-TC12 [GCM06], and MS COCO [LMB14], greatly facilitated the development of models for language and vision tasks such as image captioning. Because it takes lots of resources to label images with sentences descriptions, the scale of these datasets are relatively small (MS COCO, the largest dataset among them, only contains 1 million sentences while our Pinterest40M dataset has 300 million sentences). In addition, the language used to describe images in these datasets is relatively simple (e.g. MS COCO only has around 10,000 unique words appearing at least 3 times while there are 335,323 unique words appearing at least 50 times in Pinterest40M). The Im2Text dataset proposed in [OKB11b] adopts a similar data collection process to ours by using 1 million images with 1 million user annotated captions from Flickr. But its scale is still much smaller than our Pinterest40M dataset.

Recently, [TSF16] proposed and released the YFCC100M dataset, which is a large-scale multimedia dataset contains metadata of 100 million Flickr images. It provides rich information about images, such as tags, titles, and locations where they were taken. The users’ comments can be obtained by querying the Flickr API. Because of the different functionality and user groups between Flickr and Pinterest, the users’ comments of Flickr images are quite different from those of Pinterest (e.g. on Flickr, users tend to comment more on the photography techniques). This dataset provides complementary information to our Pinterest40M dataset.

Word Similarity-Relatedness Evaluation The standard benchmarks, such as WordSim-353/WS-Sim [FGM01, AAH09], MEN [BTB14], and SimLex-999 [HRK15], consist of a couple hundreds of word pairs and their similarity or relatedness scores. The word pairs are composed by asking human subjects to write the first related, or similar, word that comes into their mind when presented with a concept word (e.g. [NMS04, FGM01]), or by randomly selecting frequent words in large text corpus and manually searching for useful pairs (e.g. [BTB14]). In this work, we are able to collect a large number of word/phrase pairs with good quality by mining them from the click data of Pinterest’s image search system used by millions of users. In addition, because this dataset is collected through a visual search system, it is more suitable to evaluate multimodal embedding.
models. Another related evaluation is the analogy task proposed in [MSC13]. They ask the model questions like “man to woman is equal king to what?” as their evaluation. But such questions do not directly measure the word similarity or relatedness, and cannot cover all the semantic relationships of million of words in the dictionary.

**RNN for Language and Vision** Our models are inspired by recent RNN-CNN based image captioning models [DAG15, MXY15a, VTB15, KF15, CZ14, KSZ14, MWY15], which can be viewed as a special case of the sequence-to-sequence learning framework [SVL14, CMG14]. We adopt Gated Recurrent Units (GRUs [CMG14]), a variation of the simple RNN model.

**Multimodal Word Embedding Models** For pure text, one of the most effective approaches to learn word embeddings is to train neural network models to predict a word given its context words in a sentence (i.e. the continuous bag-of-word model [BSS06]) or to predict the context words given the current word (i.e. the skip-gram model [MSC13]). There is a large literature on word embedding models that utilize visual information. One type of methods takes a two-step strategy that first extracts text and image features separately and then fuses them together using singular value decomposition [BTB14], stacked autoencoders [SL14], or even simple concatenation [KB14]. [HK14, LPB15, KVM15] learn the text and image features jointly by fusing visual or perceptual information in a skip-gram model [MSC13]. However, because of the lack of large-scale multimodal datasets, they only associate visual content with a pre-defined set of nouns (e.g. [LPB15]) or perception domains (e.g. [HRK15]) in the sentences, or focus on abstract scenes (e.g. [KVM15]). By contrast, our best performing model places a soft constraint between visual features and all the words in the sentences by a weight sharing strategy as shown in Section 6.5.

### 6.4 Datasets

We constructed two datasets: one for training our multimodal word-embeddings (see Section 6.4.1) and another one for the evaluation of the learned word-embeddings (see Section 6.4.2).
This strawberry limeade cake is fruity, refreshing, and gorgeous! Those lovely layers are impossible to resist.

This is the place I will be going (hopefully) on my first date with Prince Stephen. It’s the palace gardens, and they are gorgeous. I cannot wait to get to know him and exchange photography ideas!

Make two small fishtail braids on each side, then put them together with a ponytail.

White and gold ornate library with decorated ceiling, iron-work balcony, crystal chandelier, and glass-covered shelves. (I don’t know if you’re allowed to read a beat-up paperback in this room.)

This flopsy-wopsy who just wants a break from his walk. | 18 German Shepherd Puppies Who Need To Be Snuggled Immediately

Figure 6.1: Sample images and their sample descriptions collected from Pinterest.

6.4.1 Training Dataset

Pinterest is one of the largest repository of Web images. Users commonly tag images with short descriptions and share the images (and descriptions) with others. Since a given image can be shared and tagged by multiple, sometimes thousands of users, many images have a very rich set of descriptions, making this source of data ideal for training model with both text and image inputs.

The dataset is prepared in the following way: first, we crawled the public available data on Pinterest to construct our training dataset of more than 40 million images. Each image is associated with an average of 12 sentences, and we removed duplicated or short sentences with less than 4 words. The duplication detection is conducted by calculating the overlapped word unigram ratios. Some sample images and descriptions are shown in Figure 6.1. We denote this dataset as the Pinterest40M dataset.

Our dataset contains 40 million images with 300 million sentences (around 3 billion words), which is much larger than the previous image description datasets (see Table 6.1). In addition, because the descriptions are annotated by users who expressed interest in the images, the descriptions in our dataset are more natural and richer than the annotated image description datasets. In our dataset, there are 335,323 unique words with a minimum number of occurrence of 50, compared with 10,232 and 65,552 words appearing at least 3 times in MS COCO and IM2Text dataset respectively.
Table 6.1: Scale comparison with other image descriptions benchmarks.

<table>
<thead>
<tr>
<th></th>
<th>Image</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flickr8K [HYH13]</td>
<td>8K</td>
<td>40K</td>
</tr>
<tr>
<td>Flickr30K [YLH14b]</td>
<td>30K</td>
<td>150K</td>
</tr>
<tr>
<td>IAPR-TC12 [GCM06]</td>
<td>20K</td>
<td>34K</td>
</tr>
<tr>
<td>MS COCO [LMB14]</td>
<td>200K</td>
<td>1M</td>
</tr>
<tr>
<td>Im2Text [OKB11b]</td>
<td>1M</td>
<td>1M</td>
</tr>
<tr>
<td>Pinterset40M</td>
<td>40M</td>
<td>300M</td>
</tr>
</tbody>
</table>

To the best of our knowledge, there is no previous paper that trains a multimodal RNN model on a dataset of such scale.

6.4.2 Evaluation Datasets

This work proposes to use labeled phrase triplets – each triplet is a three-phrase tuple containing phrase A, phrase B and phrase C, where A is considered as semantically closer to B than A is to C. At testing time, we compute the distance in the word embedding space between A/B and A/C, and consider a test triplet as positive if \( d(A, B) < d(A, C) \). This relative comparison approach was commonly used to evaluate and compare different word embedding models [SLM13].

In order to generate large number of phrase triplets, we rely on user-click data collected from Pinterest image search system. At the end, we construct a large-scale evaluation dataset with 9.8 million triplets (see Section 6.4.2.1), and its cleaned up gold standard version with 10 thousand triplets (see Section 6.4.2.2).

6.4.2.1 The Raw Evaluation Dataset from User Clickthrough Data

It is very hard to obtain a large number of semantically similar or related word and phrase pairs. This is one of the challenges for constructing a large-scale word/phrase similarity and relatedness
Figure 6.2: The illustration of the positive word/phrase pairs generation. We calculate a score for each annotation (i.e. a short phrase describes the items) by aggregating the click frequency of the items to which it belongs and rank them according to the score. The final list of positive phrases are generated from the top ranked phrases after removing phrases containing overlapping words with the user query phrase. See text for details.

More specifically, given a query from a user (e.g. “hair styles”), the search system returns a list of items, and each item is composed of an image and a set of annotations (i.e. short phrases or words that describe the item). Please note that the same annotation can appear in multiple items, e.g., “hair tutorial” can describe items related to prom hair styles or ponytails. We derive a matching score for each annotation by aggregating the click frequency of the items containing the annotation. The annotations are then ranked according to the matching scores, and the top ranked annotations are considered as the positive set of phrases or words with respect to the user query.

To increase the difficulty of this dataset, we remove the phrases that share common words with the user query from the initial list of positive phrases. E.g. “hair tutorials” will be removed because the word “hair” is contained in the query phrase “hair styles”. A stemmer in Python’s “stemmer” package is also adopted to find words with the same root (e.g. “cake” and “cakes” are considered as the same word). This pruning step also prevents giving bias to methods which measure the similarity between the positive phrase and the query phrase by counting the number of overlapping words between them. In this way, we collected 9,778,508 semantically similar phrase pairs.

Previous word similarity/relatedness datasets (e.g. [FGM01] [HRK15]) manually annotated each
word pair with an absolute score reflecting how much the words in this pair are semantically related. In the testing stage, a predicted similarity score list of the word pairs generated by the model in the dataset is compared with the groundtruth score list. The Spearman’s rank correlation between the two lists is calculated as the score of the model. However, it is often too hard and expensive to label the absolute related score and maintain the consistency across all the pairs in a large-scale dataset, even if we average the scores of several annotators.

We adopt a simple strategy by composing triplets for the phrase pairs. More specifically, we randomly sample negative phrases from a pool of 1 billion phrases. The negative phrase should not contain any overlapping word (a stemmer is also adopted) with both of the phrases in the original phrase pair. In this way, we construct 9,778,508 triplets with the format of (base phrase, positive phrase, negative phrase). In the evaluation, a model should be able to distinguish the positive phrase from the negative phrase by calculating their similarities with the base phrase in the embedding space. We denote this dataset as Related Phrase 10M (RP10M) dataset.

6.4.2.2 The Cleaned-up Gold Standard Dataset

Figure 6.3: The interface for the annotators. They are required to choose which phrase (positive and negative phrases will be randomly labeled as “A” or “B”) is more related to base phrase. They can click on the phrases to see Google search results.

Because the raw Related Query 10M dataset is built upon user click information, it contains some noisy triplets (e.g. the positive and base phrase are not related, or the negative phrase is strongly related to the base phrase). To create a gold standard dataset, we conduct a clean up step
Table 6.2: Sample triplets from the Gold RP10K dataset.

<table>
<thead>
<tr>
<th>Base Phrase</th>
<th>Positive Phrase</th>
<th>Negative Phrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>hair style</td>
<td>ponytail</td>
<td>pink nail</td>
</tr>
<tr>
<td>summer lunch</td>
<td>salads sides</td>
<td>packaging bottle</td>
</tr>
<tr>
<td>oil painting ideas</td>
<td>art tips</td>
<td>snickerdoodle muffins</td>
</tr>
<tr>
<td>la multi ani birthdays</td>
<td>wishes</td>
<td>tandoori</td>
</tr>
<tr>
<td>teach activities</td>
<td>preschool</td>
<td>rental house ideas</td>
</tr>
<tr>
<td>karting</td>
<td>go carts</td>
<td>office waiting area</td>
</tr>
<tr>
<td>looking down</td>
<td>the view</td>
<td>soft curls for medium hair</td>
</tr>
<tr>
<td>black ceiling</td>
<td>home ideas</td>
<td>paleo potluck</td>
</tr>
<tr>
<td>new marriage quotes</td>
<td>true love</td>
<td>winter travel packing</td>
</tr>
<tr>
<td>sexy scientist costume</td>
<td>labs</td>
<td>personal word wall</td>
</tr>
<tr>
<td>framing a mirror</td>
<td>decorating bathroom</td>
<td>celebrity style inspiration</td>
</tr>
</tbody>
</table>

using the crowdsourcing platform CrowdFlower [cro] to remove these inaccurate triplets. A sample question and choices for the crowdsourcing annotators are shown in Figure 6.3. The positive and negative phrases in a triplet are randomly given as choice “A” or “B”. The annotators are required to choose which phrase is more related to the base phrase, or if they are both related or unrelated. To help the annotators understand the meaning of the phrases, they can click on the phrases to get Google search results.

We annotate 21,000 triplets randomly sampled from the raw Related Query 10M dataset. Three to five annotators are assigned to each question. A triplet is accepted and added in the final cleaned up dataset only if more than 50% of the annotators agree with the original positive and negative label of the queries (note that they do not know which one is positive in the annotation process). In practice, 70% of the selected phrases triplets have more than 3 annotators to agree. This leads to a gold standard dataset with 10,674 triplets. We denote this dataset as Gold Phrase Query 10K (Gold RP10K) dataset.

This dataset is very challenging and a successfully model should be able to capture a variety of
semantic relationships between words or phrases. Some sample triplets are shown in Table 6.2.

6.5 The Multimodal Word Embedding Models

Figure 6.4: The illustration of the structures of our model A, B, and C. We use a CNN to extract visual representations and use a RNN to model sentences. The numbers on the bottom right corner of the layers indicate their dimensions. We use a sampled softmax layer with 1024 negative words to accelerate the training. Model A, B, and C differ from each other by the way that we fuse the visual representation into the RNN. See text for more details.

We propose three RNN-CNN based models to learn the multimodal word embeddings, as illustrated in Figure 6.4. All of the models have two parts in common: a Convolutional Neural Network (CNN [KSH12]) to extract visual representations and a Recurrent Neural Network (RNN [Elm90]) to model sentences.

For the CNN part, we resize the images to $224 \times 224$, and adopt the 16-layer VGGNet [SZ15] as the visual feature extractor. The binarized activation (i.e. 4096 binary vectors) of the layer before its SoftMax layer are used as the image features and will be mapped to the same space of the state of RNN (Model A, B) or the word embeddings (Model C), depends on the structure of the model, by a fully connected layer and a Rectified Linear Unit function ($\text{ReLU} [\text{NH10}], \text{ReLU}(x) = \max(0, x)$).

For the RNN part, we use a Gated Recurrent Unit (GRU [CMG14]), an recently very popular RNN structure, with a 512 dimensional state cell. The state of GRU $h_t$ for each word with index $t$
in a sentence can be represented as:

\[ r_t = \sigma(W_r[e_t, h_{t-1}] + b_r) \]  
(6.1)

\[ u_t = \sigma(W_u[e_t, h_{t-1}] + b_u) \]  
(6.2)

\[ c_t = \tanh(W_c[e_t, r_t \odot h_{t-1}] + b_c) \]  
(6.3)

\[ h_t = u_t \odot h_{t-1} + (1 - u_t) \odot c_t \]  
(6.4)

where \( \odot \) represents the element-wise product, \( \sigma(.) \) is the sigmoid function, \( e_t \) denotes the word embedding for the word \( w_t \), \( r_t \) and \( u_t \) are the reset gate and update gate respectively. The inputs of the GRU are words in a sentence and it is trained to predict the next words given the previous words.

We add all the words that appear more than 50 times in the Pinterest40M dataset into the dictionary. The final vocabulary size is 335,323. Because the vocabulary size is very huge, we adopt the sampled SoftMax loss \([\text{CMB15}]\) to accelerate the training. For each training step, we sample 1024 negative words according to their log frequency in the training data and calculate the sampled SoftMax loss for the positive word. This sampled SoftMax loss function of the RNN part is adopted with Model A, B and C. Minimizing this loss function can be considered as approximately maximizing the probability of the sentences in the training set.

As illustrated in Figure 6.4, Model A, B and C have different ways to fuse the visual information in the word embeddings. Model A is inspired by the CNN-RNN based image captioning models \([\text{VTB15, MWY15}]\). We map the visual representation in the same space as the GRU states to initialize them (i.e. set \( h_0 = \text{ReLU}(W_I f_I) \)). Since the visual information is fed after the embedding layer, it is usually hard to ensure that this information is fused in the learned embeddings. We adopt a transposed weight sharing strategy proposed in \([\text{MWY15}]\) that was originally used to enhance the models’ ability to learn novel visual concepts. More specifically, we share the weight matrix of the SoftMax layer \( U_M \) with the matrix \( U_w \) of the word embedding layer in a transposed manner. In this way, \( U_M^T \) is learned to decode the visual information and is enforced to incorporate this information
into the word embedding matrix \( U_w \). In the experiments, we show that this strategy significantly improve the performance of the trained embeddings. Model A is trained by maximizing the log likelihood of the next words given the previous words conditioned on the visual representations, similar to the image captioning models.

Compared to Model A, we adopt a more direct way to utilize the visual information for Model B and Model C. We add direct supervisions of the final state of the GRU (Model B) or the word embeddings (Model C), by adding new loss terms, in addition to the negative log-likelihood loss from the sampled SoftMax layer:

\[
L_{\text{state}} = \frac{1}{n} \sum_s \| h_s - \text{ReLU}(W_f f_s) \| \tag{6.5}
\]

\[
L_{\text{emb}} = \frac{1}{n} \sum_s \frac{1}{l_s} \sum_t \| e_t - \text{ReLU}(W_f f_s) \| \tag{6.6}
\]

where \( l_s \) is the length of the sentence \( s \) in a mini-batch with \( n \) sentences, Eqn. 6.5 and Eqn. 6.6 denote the additional losses for model B and C respectively. The added loss term is balanced by a weight hyperparameter \( \lambda \) with the negative log-likelihood loss from the sampled SoftMax layer.

### 6.6 Experiments

#### 6.6.1 Training Details

We convert the words in all sentences of the Pinterest40M dataset to lower cases. All the non-alphanumeric characters are removed. A start sign \( \langle \text{bos} \rangle \) and an end sign \( \langle \text{eos} \rangle \) are added at the beginning and the end of all the sentences respectively.

We use the stochastic gradient descent method with a mini-batch size of 256 sentences and a learning rate of 1.0. The gradient is clipped to 10.0. We train the models until the loss does not decrease on a small validation set with 10,000 images and their descriptions. The models will scan the dataset for roughly five 5 epochs. The bias terms of the gates (i.e. \( b_r \) and \( b_u \) in Eqn. 6.1 and 6.2) in the GRU layer are initialized to 1.0.
Table 6.3: Performance comparison of our Model A, B, C, and their variants.

<table>
<thead>
<tr>
<th>Model Description</th>
<th>Gold RP10K</th>
<th>RP10M</th>
<th>dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pure text RNN</td>
<td>0.748</td>
<td>0.633</td>
<td>128</td>
</tr>
<tr>
<td>Model A without weight sharing</td>
<td>0.773</td>
<td>0.681</td>
<td>128</td>
</tr>
<tr>
<td>Model A (weight shared multimodal RNN)</td>
<td><strong>0.843</strong></td>
<td><strong>0.725</strong></td>
<td>128</td>
</tr>
<tr>
<td>Model B (direct visual supervisions on the final RNN state)</td>
<td>0.705</td>
<td>0.646</td>
<td>128</td>
</tr>
<tr>
<td>Model C (direct visual supervisions on the embeddings)</td>
<td>0.771</td>
<td>0.687</td>
<td>128</td>
</tr>
<tr>
<td>Word2Vec-GoogleNews [MSC13]</td>
<td>0.716</td>
<td>0.596</td>
<td>300</td>
</tr>
<tr>
<td>GloVe-Twitter [PSM14]</td>
<td>0.693</td>
<td>0.617</td>
<td>200</td>
</tr>
</tbody>
</table>

6.6.2 Evaluation Details

We use the trained embedding models to extract embeddings for all the words in a phrase and aggregate them by average pooling to get the phrase representation. We then check whether the cosine distance between the (base phrase, positive phrase) pair are smaller than the (base phrase, negative phrase) pair. The average precision over all the triplets in the raw Related Phrases 10M (RP10M) dataset and the Gold standard Related Phrases 10K (Gold RP10K) dataset are reported.

6.6.3 Results on the Gold RP10K and RP10M datasets

We evaluate and compare our Model A, B, C, their variants and several strong baselines on our RP10M and Gold RP10K datasets. The results are shown in Table 6.3. “Pure Text RNN” denotes the baseline model without input of the visual features trained on Pinterest40M. It have the same model structure as our Model A except that we initialize the hidden state of GRU with a zero vector. “Model A without weight sharing” denotes a variant of Model A where the weight matrix $U_w$ of the word embedding layer is not shared with the weight matrix $U_M$ of the sampled SoftMax layer (see Figure 6.4 for details). \(^2\) “Word2Vec-GoogleNews” denotes the state-of-the-art off-the-shelf word

\(^2\)We also try to adopt the weight sharing strategy in Model B and C, but the performance is very similar to the non-weight sharing version.
embedding models of Word2Vec [MSC13] trained on the Google-News data (about 300 billion words). “GloVe-Twitter” denotes the GloVe model [PSM14] trained on the Twitter data (about 27 billion words). They are pure text models, but trained on a very large dataset (our model only trains on 3 billion words). Comparing these models, we can draw the following conclusions:

- Under our evaluation criteria, visual information significantly helps the learning of word embeddings when the model successfully fuses the visual and text information together. E.g., our Model A outperforms the Word2Vec model by 9.5% and 9.2% on the Gold RP10K and RP10M datasets respectively. Model C also outperforms the pure text RNN baselines.

- The weight sharing strategy is crucial to enhance the ability of Model A to fuse visual information into the learned embeddings. E.g., our Model A outperforms the baseline without this sharing strategy by 7.0% and 4.4% on Gold RP10K and RP10M respectively.

- Model A performs the best among all the three models. It shows that soft supervision imposed by the weight-sharing strategy is more effective than direct supervision. This is not surprising since not all the words are semantically related to the content of the image and a direct and hard constraint might hinder the learning of the embeddings for these words.

- Model B does not perform very well. The reason might be that most of the sentences have more than 8 words and the gradient from the final state loss term $L_{state}$ cannot be easily passed to the embedding of all the words in the sentence.

- All the models trained on the Pinterest40M dataset performs better than the skip-gram model [MSC13] trained on a much larger dataset of 300 billion words.

### 6.7 Discussion

In this chapter, we investigate the task of training and evaluating word embedding models. We introduce Pinterest40M, the largest image dataset with sentence descriptions to the best of our knowledge, and construct two evaluation dataset (i.e. RP10M and Gold RP10K) for word/phrase
similarity and relatedness evaluation. Based on these datasets, we propose several CNN-RNN based multimodal models to learn effective word embeddings. Experiments show that visual information significantly helps the training of word embeddings, and our proposed model successfully incorporates such information into the learned embeddings.

There are lots of possible extensions of the proposed model and the dataset. E.g., we plan to separate semantically similar or related phrase pairs from the Gold RP10K dataset to better understand the performance of the methods, similar to [AAH09]. We will also give relatedness or similarity scores for the pairs (base phrase, positive phrase) to enable same evaluation strategy as previous datasets (e.g. [BTB14, FGM01]). Finally, we plan to propose better models for phrase representations.
CHAPTER 7

Conclusion

To conclude, the multimodal learning field is a very promising field with numerous real world applications, such as image annotation, multimodal image search system, and visual question answering systems for the visually impaired person. This field is also crucial as we are exploring the way toward general artificial intelligence and cannot be bypassed. In this thesis, we provide the first of its kind and state-of-the-art methods and large-scale datasets to many of the core tasks in this field, including image captioning, novel visual concept learning, referring expressions, visual question answering, multimodal word embedding learning, and multi-label classification.

Despite the current success of these models, there is still a very long way to go to address this tasks fully under different conditions. For example, the image captioning model works very well on the existing datasets where we train the model, but its performance drops when we input images whose scene or style are different from the ones in the datasets. This indicates that the model might overfit to the dataset and we should improve the model as well as the diversity of the images in the datasets.

Finally, we should not limited to the area of vision and language, and should include other modalities, such as speech, action, into our learning machine. I strongly believe that the multimodal, multitask learning tasks will be very promising and important in the future.
REFERENCES


111


