LEARNING JOINT LATENT REPRESENTATIONS FOR IMAGES AND LANGUAGE

BY

LIWEI WANG

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science in the Graduate College of the University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Associate Professor Svetlana Lazebnik, Chair
Professor David Forsyth
Associate Professor Julia Hockenmaier
Assistant Professor Alexander Schwing
Associate Professor Zhuowen Tu, University of California, San Diego
ABSTRACT

Computer vision is moving from predicting discrete, categorical labels to generating rich descriptions of visual data, in particular, in the form of natural language. Learning the joint latent representations for images and language is vital to solving many image-text tasks, including image-sentence retrieval, visual grounding, and image captioning, etc.

In this thesis, we first propose two-branch neural networks for learning the similarity between these two data modalities. Two network structures are proposed to produce different output representations. The first one, referred to as an embedding network, learns an explicit shared latent embedding space with a maximum-margin ranking loss and novel neighborhood constraints. The second network structure, referred to as a similarity network, fuses the two branches via element-wise product and is trained with regression loss to directly predict a similarity score. Extensive experiments show that our networks achieve high accuracies for phrase localization in the Flickr30K Entities dataset and for bi-directional image-sentence retrieval in the Flickr30K and COCO datasets.

Then, we explore the image captioning problem using conditional variational auto-encoders (CVAEs). Standard CVAEs with a fixed Gaussian prior yield descriptions with too little variability. Instead, we propose two models that explicitly structure the latent space with $K$ components corresponding to different types of image content, and combine components to create priors for images that contain multiple types of content simultaneously (e.g. several kinds of objects). The first model uses a Gaussian Mixture model (GMM) prior while the second one defines a novel Additive Gaussian (AG) prior that linearly combines component means. Experiments show that both models produce captions that are more diverse and more accurate than a strong LSTM baseline or a “vanilla” CVAE with a fixed Gaussian prior, with AG-CVAE showing particular promise.

In order to further improve the caption decoder inherited from the AG-CVAE model, we attempt to train it by optimizing caption evaluation metrics (e.g. BLEU scores) using policy gradient from reinforcement learning. The loss function contains two terms: one is maximum likelihood estimator (MLE loss) and the other one is a reinforcement term based on a sum of non-differentiable rewards. Experiments show that training the decoder with this combination loss can help to generate more accurate captions. We also study the problem of ranking generated sentences conditioned on the image input and explore several variants of deep rankers built on top of the two-branch networks proposed earlier.
To my parents, for their love and support.
ACKNOWLEDGMENTS

I would like to start by expressing my sincere gratitude to my advisor, Professor Svetlana Lazebnik, for her excellent support of my doctoral studies. She taught me to focus on solving challenging research problems. I am very grateful for the opportunity I have had to work with her, and this thesis could not have been accomplished without her guidance.

I still remember my campus visit five years ago, the first time I came to Champaign-Urbana. In the vision group meeting, I was very excited to meet Professors Svetlana Lazebnik, David Forsyth, Derek Hoiem, as well as many other smart students. If I had to use one word to describe my feeling at that time, I would choose the word warm. And this feeling has lasted since then and will always be there in my heart in the future.

I would also like to express my thanks to Professor Alexander Schwing. I appreciated every meeting and every discussion with him over the past two years. I would also like to thank my other committee members: Professors David Forsyth, Julia Hockenmaier, and Zhuowen Tu. I have appreciated those insightful discussions and collaborations with them during my graduate studies. I am very grateful for their great suggestions for and feedback on this thesis.

I am fortunate to have met wonderful colleagues in this vision group. I have cherished the time spent in the shared office with Bryan A. Plummer, Arun Mallya, and Kevin J Shih. Working together with them for more than one thousand days over the past years is a valuable memory. I will especially remember those great moments of enjoying delicious food, searching for the best coffee in Champaign, and discussing papers together. I would also like to sincerely thank everyone I have met here including: Kevin Karsch, Amin Sadeghi, Qieyun Dai, P. Daphne Tsatsoulis, Saurabh Singh, Kevin Shih, Jason Rock, Joseph Degol, Bryan Plummer, Arun Mallya, Rajbir Kataria, Jiajun Lu, Tanmay Gupta, Chuhang Zou, Aditya Deshpande, Zhizhong Li, Mariya Vasileva, Daniel McKee, Alice Lai, and many others!

I would also like to express thanks to those I collaborated with during my Ph.D. research. Finally and most importantly, I would like to thank my parents for bringing me to this beautiful world, for supporting me through this long educational journey, and for their love!
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>RELATED WORK</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Image-Language Tasks and Datasets</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Cross-modal Learning</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>Methods of Generating Image Descriptions</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>LEARNING TWO-BRANCH NEURAL NETWORKS FOR IMAGE-TEXT MATCHING TASKS</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>Embedding Network</td>
<td>12</td>
</tr>
<tr>
<td>3.2</td>
<td>Similarity Network</td>
<td>16</td>
</tr>
<tr>
<td>3.3</td>
<td>Experiments on Phrase Localization</td>
<td>16</td>
</tr>
<tr>
<td>3.4</td>
<td>Experiments on Image-Sentence Retrieval</td>
<td>21</td>
</tr>
<tr>
<td>3.5</td>
<td>Conclusion</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>GENERATING IMAGE DESCRIPTION USING A VARIATIONAL AUTO-ENCODER WITH AN ADDITIVE GAUSSIAN ENCODING SPACE</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>Conditional Variational Auto-encoders</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>Gaussian Mixture Prior and Additive Gaussian Prior</td>
<td>34</td>
</tr>
<tr>
<td>4.3</td>
<td>Experiments on Image Captioning</td>
<td>37</td>
</tr>
<tr>
<td>4.4</td>
<td>Conclusion and Future work</td>
<td>43</td>
</tr>
<tr>
<td>5</td>
<td>TRAINING WITH REINFORCEMENT LEARNING AND RE-RANKING</td>
<td>47</td>
</tr>
<tr>
<td>5.1</td>
<td>Mixed Reinforcement Training</td>
<td>48</td>
</tr>
<tr>
<td>5.2</td>
<td>Two-branch Ranker for re-ranking sentences</td>
<td>55</td>
</tr>
<tr>
<td>5.3</td>
<td>Conclusion and Future Work</td>
<td>57</td>
</tr>
<tr>
<td>6</td>
<td>CONCLUSIONS</td>
<td>61</td>
</tr>
</tbody>
</table>

REFERENCES | 62 |
CHAPTER 1: INTRODUCTION

Computer vision has expanded from predicting discrete and categorical labels to generating rich descriptions of visual data in the form of natural language. We are witnessing a surge of interest in tasks that involve cross-modal learning from image and text data, widely viewed as the “next frontier” of scene understanding. Tasks at the intersection of these two modalities include bi-directional image-sentence search, image captioning, visual grounding, etc. Bi-directional image-sentence search [1, 2, 3], as the name suggests, aims to retrieve images corresponding to a given sentence query, and vice versa. Image captioning [4, 5, 6] seeks to generate a natural language description of an input image. Another recent task, motivated by the target of creating a visual Turing test, is Visual Question Answering (VQA) [7, 8, 9], where a system learns to answer free-form questions about image content. Visual grounding tasks like referring expression understanding [10, 11] and phrase localization [12] find image regions indicated by questions, phrases, or sentences. Together with these tasks, a number of large-scale datasets and benchmarks have recently been proposed, such as the COCO [13] and Flickr30K [14] datasets for image captioning, Flickr30K Entities [12] for phrase localization, the Visual Genome dataset [15] for localizing textual description of images, and the VQA dataset [7] for question answering.

A key ingredient of solutions for all these varied tasks is a joint embedding space, where images and text can both be represented, so as to move across modalities with as little loss of information as possible. We study neural architectures for tackling this key problem – learning joint latent representations for visual (e.g. images or regions) and text data (e.g. sentences or phrases). Chapter 2 will review joint image-language tasks, background knowledge and also classical methods that are related to this dissertation. After that, we begin our investigation on joint representations in Chapter 3 by proposing a class of two-branch network architectures that operate on both images and text. The first one, referred to as an embedding network, learns an explicit shared latent embedding space with a maximum-margin ranking loss and novel neighborhood constraints. The second one, referred to as a similarity network, fuses the two branches via element-wise product and is trained with a regression loss to predict a similarity score directly. Extensive experiments show that our models achieve high accuracies for phrase localization in the Flickr30K Entities dataset and for bi-directional image-sentence retrieval in Flickr30K and COCO datasets. While our two-branch network can match images and language well in the latent space, it is unable to generate sentences from this latent space. We are interested in moving a step further and generating a sentence directly from the latent encoding space, instead of merely performing “search and match”
restricted to the database. Automatic image captioning [16, 17, 18, 19, 20, 21] has received a lot of attention of late. State-of-the-art captioning techniques [6, 22, 23, 24] are based on recurrent neural nets with long-short term memory (LSTM) units [25], which take as the input a feature representation of a provided image and are typically trained to maximize the likelihood of reference human descriptions. While the captions produced by such methods can seem surprisingly good at first glance, they are highly stereotyped and lacking in diversity [26, 27]. In Chapter 4, we introduce a method to generate image descriptions using a conditional variational auto-encoder (CVAE) with a data-dependent Gaussian prior on the encoding space. Standard CVAEs with a fixed Gaussian prior easily collapse and generate descriptions with little variability. Our approach addresses this problem by linearly combining multiple Gaussian priors based on the semantic content of the image, increasing the flexibility and representational power of the generative model. We evaluate this Additive Gaussian CVAE (AG-CVAE) approach in the COCO dataset and show that it is able to produce captions that are both more diverse and more accurate than a strong LSTM baseline and other CVAE variants.

Image captioning, as other language generation tasks, can be evaluated by standard syntactic metrics, such as BLEU [28], METEOR [29], ROUGE [30], SPICE [31], and CIDEr [32]. To further improve the accuracy of an image captioning system, optimizing these evaluation metrics seems to be a straightforward choice. Due to the non-differentiable nature of these metrics, recent works [33, 34] use policy gradient (PG) from the reinforcement learning field [35, 36, 37] to maximize the expected rewards (e.g. BLEU scores). In Chapter 5, we propose to improve our caption decoder inherited from the pre-trained CVAE framework via optimizing BLEU score rewards with policy gradient. Our objective function has two terms. The first term is the maximum likelihood estimator (MLE loss) and the second one is the surrogate reinforcement term based on a sum of non-differentiable rewards. Experiments show that we can improve the decoder to generate more accurate sentences by using the combined loss function.

In Chapter 5, we also consider the problem of ranking sentence candidates generated by our CVAE decoder. In the inference stage, given the input image and a set of generated sentences, we need to know which sentence is the best and can be chosen as the output. In Chapter 4, we follow consensus re-ranking [22, 38] to rank generated sentences. Specifically, we calculate the consensus score of each generated sentence (hypothesis) as the mean similarity between it and all other captions of \(k\) nearest neighbor images from the training set. However, this consensus re-ranking step is very time-consuming. And it also shows the limitation when the training set is not publicly available in real applications. To this end, we use the two-branch framework from Chapter 3 to rank generated sentences. We try different triplet sampling
strategies and find that construction of triplets can affect performances. Though our best ranker can finish the re-ranking much more quickly, it still cannot outperform consensus re-ranking in accuracy.

The main contributions of this dissertation can be summarized as follows:

- For both bi-directional image-sentence retrieval and phrase localization tasks, our proposed model “two-branch networks” gives the promising performance (Chapter 3). It can also be used in many other vision-language tasks as an important component. This work appeared in CVPR 2016 [39] with an extended version in TPAMI 2018 [40].

- We study the image captioning problem from a new angle. Not only do we want to generate image captions in an “accurate” way, but we also expect them to be “diverse.” Our proposed conditional variational auto-encoder (CVAE) framework with an additive encoding space moves one step closer to this goal (Chapter 4), and it appeared in NIPS 2017 [41].

- We explore the idea of training our CVAE decoder with the reinforcement learning. By combining maximum likelihood loss and the surrogate reinforcement term, we show that the decoder can be improved. We also study the challenging problem of ranking generated sentences conditioned on the image input (Chapter 5). Though we haven’t shown satisfactory improvement for either reinforcement training of CVAE or sentence re-ranking, we hope that our explorations can inspire future work in this direction.
CHAPTER 2: RELATED WORK

This chapter contains a review of important works that are related to this dissertation. In Section 2.1, we discuss several image-language datasets that are used through this thesis. In Section 2.2, we review joint embedding methods related to our approaches introduced in Chapter 3. In Section 2.3, we discuss the historical and ongoing development of image captioning methods, which are related to our work in Chapters 4 and 5.

2.1 IMAGE-LANGUAGE TASKS AND DATASETS

In computer vision, joint image-language tasks have attracted a lot of attention in recent years. One major category of literature explores image-sentence retrieval [1, 3, 14, 42, 43, 44]. Most of these works focus on learning to match the whole image with sentences. In order to encourage more comprehensive understanding between two modalities, recently, there is a new task of visual grounding, i.e., localization of textual mentions of entities in an image. In a work that we contributed to, phrase localization [12, 45] is proposed as a visual grounding task and introduced together with a dataset called Flickr30K Entities [45]. Another related task is referring expression understanding [46, 47]. Different from phrase localization which finds the bounding box according to the input noun phrase (NP), referring expression focuses more on “uniquely” identifying an object instance in an image. Another line of research in the image-language domain focuses on generating textual descriptions for a given image [6, 22, 23, 33, 34, 48].

Apart from these, there are also emerging tasks of visual question answering (VQA) [7, 49], visual dialogue [50], etc. In this thesis, we mainly focus on addressing the joint image-language tasks of image-sentence retrieval, phrase localization, and image captioning. We hope that our proposed methods can also prove useful for other vision-language tasks.

In the following, we will provide a brief review of image-language datasets that are used in this thesis: Flickr30K [14], Flickr30K Entities [45], and COCO [13].

- **Flickr30K** [14] contains 31,783 images focusing mainly on people and animals, and 158,915 English captions. Each image has five associated ground-truth captions, which are annotated by crowdsourcing. In Chapter 3, experiments of bi-directional image-sentence retrieval are conducted on this dataset.

- **Flickr30K Entities** [45] augments Flickr30k [14] by identifying which mentions among the captions of the same image refer to the same set of entities, resulting in 244,035
coreference chains, and which image regions depict the mentioned entities, resulting in 275,775 bounding boxes. We show examples from this dataset in Fig. (2.1). In the work of [45], phrase localization is proposed as a novel task along with this dataset. In Chapter 3, we propose two-branch networks for phrase localization and show promising results in this dataset.

- COCO [13] contains images of 91 types of objects and is labeled with a total of 2.5 million instances in 328k images. For each image, there are also five annotated captions. Examples from this dataset are listed in Fig. (2.2). In Chapter 3, we do experiments on image-sentence retrieval with COCO datasets (Flickr30K is also used as mentioned above). The object information from this dataset can also be very useful in generating image captions as will be shown in Chapters 4 and 5.

There are also some other popular vision-language datasets in this field. The ReferIt dataset [10] and Google Refexp dataset [48] focus on referring expressions. Visual Genome [15] provides dense annotations of regions. MadLibs [8] contains a subset of COCO images which are labeled with several types of focused fill-in-the-blank descriptions. Visual question answering (VQA) [7] and visual dialog [50] provide vision-language interactions.
2.2 CROSS-MODAL LEARNING

In this section, we will review several lines of cross-modal learning methods. These will be related to our approach in Chapter 3.

CCA-based methods

One of the most popular baselines for image-text embedding is Canonical Correlation Analysis (CCA) [52, 53]. It aims to find projections $W$ and $U$ for the two views $X$ and $Y$ such that the normalized correlation between the projected data is maximized:

$$\max_{W,U} \text{trace}(W^TX^TYU) \quad \text{s.t.} \quad W^TX^TX = I, \quad U^TY^TY = I. \quad (2.1)$$

The CCA objective function can be solved as a generalized eigenvalue problem, and entries of the top $c$ leading eigenvectors are concatenated to form $W$ and $U$. Normalized CCA is proposed by [54], which scales the columns of the CCA projection matrices by a power of the corresponding eigenvalues, and uses cosine similarity instead of Euclidean distance. And this normalization gives big improvements in practice [2, 3, 54]. Despite being a classic textbook method, CCA has turned out to be a surprisingly powerful baseline. Klein et al. [2] showed that properly normalized CCA [54] with state-of-the-art image and text features can outperform much more complicated models. The main disadvantage of CCA is its
high memory cost, as it requires loading all the data into memory to compute the data covariance matrix. To obtain a nonlinear embedding, other works have opted for kernel CCA [52, 55], which finds maximally correlated projections in reproducing kernel Hilbert spaces with corresponding kernels, but this approach does not scale beyond a couple thousand data points.

**Deep multimodal representations**

To extend CCA to learning nonlinear projections and improve its scalability to large training sets, Andrew et al. [56] and Yan and Mikolajczyk [57] proposed to cast CCA into a deep learning framework. Their methods are trained using stochastic gradient descent (SGD) and thus can be applied to large-scale datasets. However, as pointed out in [58], SGD cannot guarantee a good solution to the generalized eigenvalue problem at the heart of CCA because covariance estimation in each minibatch is unstable. In Chapter 3, our proposed networks share a similar two-branch architecture with deep CCA models [56, 57], but they are much more stable and accurate.

Apart from deep CCA, many other deep learning methods have been proposed for joint modeling of multiple modalities. Some of the earlier techniques have included restricted Boltzmann machines and autoencoders [59, 60]. For image-text tasks, recurrent text representations are the most popular among current approaches [18, 22, 61, 62, 63]. In Chapter 3, we try both orderless text features from [2] and recurrent network based language models (e.g. LSTM [25]).

**Ranking-based methods**

Some of the most successful multi-modal methods, whether they be linear models or deep networks, are trained with a ranking loss. For example, WSABIE [64] and DeViSE [65] learn linear transformations of visual and textual features into a shared space using a single-directional ranking loss, which applies a margin-based penalty to an incorrect annotation when it gets ranked higher than a correct one for describing an image. A bi-directional ranking loss adds the missing link in the opposite direction: it further ensures that for each annotation, the corresponding image gets ranked higher than unrelated images [1, 62, 66]. In Chapter 3, our proposed embedding network is also trained with the bi-directional loss, but we carefully explore a number of implementation choices, resulting in a model that can significantly outperform CCA-based methods and scale to large datasets.

**Metric learning and Siamese networks**

In Chapter 3, we propose to add constraints that preserve the neighborhood structure within each individual view to the bi-directional ranking loss. Specifically, in the learned latent space, we want images (resp. sentences) with similar meaning to be close to each other.
Such within-view neighborhood preservation constraints have been extensively explored in the metric learning literature [67, 68, 69, 70, 71, 72]. In particular, the Large Margin Nearest Neighbor (LMNN) approach [71] tries to ensure that for each image its target neighbors from the same class are closer than samples from other classes. As our work will show, these constraints are also helpful for the cross-view matching task, and for training models that can achieve high accuracy both for cross-view and within-view matching.

Our two-branch networks in Chapter 3 are related to Siamese networks in metric learning [73, 74, 75, 76, 77, 78]. However, instead of learning a similarity function between two instances from the same modality using tied weights, we learn the embedding space across two different modalities with asymmetric branches.

**Classification-based methods**

Learning the similarity between images and text can be also modeled as classification. Deep models can be designed to answer whether two input visual and text samples match each other [9, 79, 80]. For example, Jabri et al. [9] used a softmax function to predict whether the input image and question match with the answer choice for VQA. Ba et al. [79] trained a two-branch network using classification loss to match visual and text data for zero-shot learning. Rohrbach et al. [81] used a softmax function to estimate the posterior probability of a phrase over all the available region proposals in an image. To fuse region and phrase features, they performed a linear transformation in each branch, followed by sum, followed by a ReLU nonlinearity and a fully connected (FC) layer. In a subsequent work, Fukui et al. [80] systematically investigated multiple feature fusion strategies and found element-wise product to be among the most effective. They then proposed a novel Multimodal Compact Bilinear Pooling (MCB) approach that slightly outperformed element-wise product. However, MCB has a high memory cost, necessitating the use of sketch approximations. Like [81], MCB uses softmax to map a phrase to a the single best region proposal from the image, with all the other regions (including ones having a high overlap with the ground truth) designated as negatives.

In Chapter 3, our second network type, the similarity network, also builds on the idea of directly predicting similarity between a phrase and a region through classification. However, instead of using softmax loss, we adopt non-exclusive logistic regression loss and treat each phrase-region pair as an independent binary classification problem – that is, for a given phrase, more than one region in the same image can be positive. At training time, this allows us to augment the ground truth region for a phrase with other positive examples having a high overlap with it. As our experiments in Chapter 3 will show, this positive data augmentation strategy plays a much more important role in improving performance than the fusion strategy, allowing us to outperform other baselines using a much simpler element-wise
2.3 METHODS OF GENERATING IMAGE DESCRIPTIONS

In this section, we will review the development of image captioning methods, which are related to our work in Chapters 4 and 5.

Compared to the image-sentence retrieval task that aims at ranking existing sentences in the dataset, being able to automatically generate a description about the content of an image is a more challenging task. Early works [17, 19, 82] have brought this topic to the vision community even before the boom of deep learning in predicting image labels [83].

More recently, a line of image captioning works [23, 24, 84] uses sequence models to generate language descriptions conditioned on the input image features. They use convolutional neural networks pre-trained on ImageNet to extract image features, which are then fed into a recurrent network (often made up of LSTM units [25]) to model the sentences word-by-word. These networks are usually trained with a maximum likelihood objective. This group of approaches is inspired by the success of sequence generation in machine translation [85, 86]. The key difference is that, instead of using a source sentence as an input, these image captioning networks start with the image features.

We should mention that, apart from using recurrent networks as the language model framework, recently, some researchers have tried convolutional neural networks for sequence modeling and achieved promising results in both image captioning and language generation [87, 88].

To improve image captioning performance, there are several lines of approaches. One category of works [23, 89, 90] focuses on bringing attention modules into the design of both vision and language networks. That is, relevant words and regions will be given larger weights in the end-to-end learning framework. Another group of works [91, 92] tries to boost the performance with additional features, e.g., attributes or concepts. Additionally, there is also an emerging trend of works [26, 33, 34, 93] that optimize non-differentiable automated sentence evaluation metrics directly with reinforcement learning. Among these metrics, BLEU [28] analyzes the co-occurrences of n-grams between the candidate sentences and reference ones. It is computed using a weighted geometric mean of n-gram precisions. CIDEr [32] measures consensus scores by weighting the Term Frequency Inverse Document Frequency (TF-IDF) for each n-gram.

Our Chapters 4 and 5 mainly focus on examining image captioning from a new angle of “diversity”. To do so, we build our decoders with vanilla LSTM units. We believe that
by using the above mentioned methods, such as convolutional language models or attention modules, our baselines can also be improved to generate more accurate captions.

**Diverse Image Captioning**

Some recent works on image captioning have started looking at the problem of generating diverse descriptions. A new beam search method is proposed in [94] to create more diversity. It is orthogonal to the task of image captioning and can be applied to any posterior. Generative adversarial networks (GANs) [95] have been successfully used in many vision tasks recently. Conditional GAN is used in [26] for image captioning, and the diversity can be brought by sampling a set of vectors \((z)\) from the noise space. Additionally, adversarial training in combination with an approximate Gumbel sampler [96] is used in [27] to match the generated captions to the human annotations.

GAN-based image captioning methods seem to improve diversity but suffer from lower accuracy. Performances on metrics like BLEU scores drop drastically compared to a conventional LSTM captioning baseline. Therefore, one may ask the question, *Can we improve the diversity without losing accuracy?*

In Chapter 4, we propose a novel scheme of generating diverse image captions. It is based on a conditional variational auto-encoder with additive Gaussian latent space (AG-VAE). The diversity comes from the sampling of \(z\) vectors in the learned latent space. At the same time, we demonstrate improvements in accuracy over the standard LSTM baseline. In Chapter 5, we will see that diversity can still be kept even with an improved decoder, which is optimized by policy gradient.
CHAPTER 3: LEARNING TWO-BRANCH NEURAL NETWORKS FOR IMAGE-TEXT MATCHING TASKS

This chapter proposes a discriminative method for learning joint representations for images and language. The main idea behind “two-branch” neural networks is to transform the two inputs of images and language, each of which is fed to an input branch, into a joint embedding space and learn semantic similarities in the joint space. In this shared space, various image-text tasks can be done by doing the nearest neighbor search.

We study the joint latent representation for a core problem underlying most image-text tasks—how to measure the semantic similarity between visual data, e.g. images or regions, and text data, e.g. sentences or phrases. Learning this similarity requires connecting low-level pixel values and high-level language descriptions. Fig. (3.1) shows an example of a phrase description of an image region from the Flick30K Entities dataset. Matching the phrase “fire pit” to its corresponding region requires not only distinguishing between the correct region and background clutter, but also understanding the difference between “fire pit” and other visual concepts that might be present in the image. Naively, one might consider training binary or multi-class classifiers to estimate the probabilities of various concepts given image regions. However, the natural language vocabulary of visual concepts is very large, even if we restrict these concepts to nouns or simple noun phrases. Further, different concepts have complex semantic similarity relationships between them – for example, “fire” and “flame” are synonyms, “fireplace” is similar in meaning but not identical to “fire pit,” and attributes can modify the meaning of head nouns (“fire pit” is not the same as “pit”). This suggests that, instead of representing different phrases using separate classifiers, representing text in a continuous “semantic” embedding space is more appropriate. Furthermore, the frequencies of different phrases are highly unbalanced: the word “fire” only occurs a few times in the Flickr30K Entities dataset, while the most common words, such as “man,” show up a few hundred times. For all these reasons, training separate per-concept classifiers is undesirable. A more natural approach is to design a model that takes in continuous image and text features (for the latter, derived from continuous word embeddings like word2vec [97]) and predicts a similarity score. This approach has the advantage of treating image and text symmetrically, enabling both image-to-text and text-to-image retrieval, and of being easily extendable from individual words and simple phrases to arbitrarily complex sentences, provided a continuous feature encoding for sentences can be devised.

In this chapter, we study two variants of two-branch networks:

**Embedding Network:** The goal of this network is to map image and text features, which may initially have different dimensions, onto a joint latent space of common dimensionality
in which matching image and text features have high cosine similarity. Each branch passes
the data through two layers with nonlinearities, followed by L2 normalization. We train
the network with a bi-directional ranking loss which enforces that matched sample pairs
should have smaller distance than unmatched ones in the embedding space. We also pro-
pose augmenting this loss with neighborhood information in each modality via novel triplet
constraints and sampling strategy. In particular, where different phrases or sentences can
be used to describe the same image or region, we force them to be close to each other. We
argue that adding these constraints can help to regularize the learning of the embedding
space, especially facilitating matching within the same modality, i.e., sentence-to-sentence
retrieval.

Similarity Network: In our alternative architecture, image and text data are also passed
through branches with two layers with nonlinearities, but then, element-wise product is used
to aggregate features from the two branches into a single vector, followed by a further series
of fully connected layers. This network is trained with logistic regression loss to match the
output score to +1 for positive pairs and −1 for negative pairs.

This network is notably simpler but also less flexible than our embedding network, as it no
longer has an explicit embedding space and cannot encode structural constraints. However,
it still achieves comparable performance on the phrase localization task.

In this chapter, Sections 3.1 and 3.2 describe the network structure and models of embed-
ding network and similarity network, respectively. Our main motivation in learning the joint
representation is to improve the performance on image-language tasks. To evaluate this abil-
ity, we use our two-branch networks for phrase localization and image-sentence retrieval. In
Section 3.3, we present experiments on phrase localization. We will use our embedding net-
work to improve performance on the standard task of bi-directional image-sentence retrieval
in Section 3.4. To further show the strength of the neighborhood preserving constraints, we
also test it on the single-modality retrieval task: sentence to sentence retrieval, as will be
shown in Section 3.4.

3.1 EMBEDDING NETWORK

3.1.1 Network Architecture

The embedding network, illustrated in Fig. (3.1) (middle), has two branches, each com-
posed of a series of fully connected (FC) layers, separated by Rectified Linear Unit (ReLU)
nonlinearities. We apply batch normalization [98] right after the FC layer to improve the
convergence during training. The output vectors are further normalized by their $L2$ norm
for efficient computation of Euclidean distance.

The embedding architecture is highly flexible. The two branches can have different numbers of layers. The inputs can be either pre-computed features or outputs of other networks (e.g. CNNs or RNNs), and back-propagation of gradients to the input networks is possible. In our work, we focus on investigating the behavior of the two-branch networks and thus stick to pre-computed image features, which already give us state-of-the-art results.

3.1.2 Learning Cross-Modal Matching by Ranking

The embedding network is trained using stochastic gradient descent [99] with a margin-based loss that encodes both bi-directional ranking constraints and neighborhood-preserving constraints within each modality. This section will discuss the design of our loss function and the strategy of sampling triplets for mini-batches.

**Bi-directional ranking loss.** Given a visual input $x_i$ (a whole image or a region), let $Y_i^+$ and $Y_i^-$ denote its sets of matching (positive) and non-matching (negative) text samples, respectively. If $y_j$ and $y_k$ are positive and negative samples for $x_i$, we want the distance between $x_i$ and $y_j$ to be smaller than the distance between $x_i$ and $y_k$, with a margin of $m$. This leads to the following triplet-wise constraint:

$$d(x_i, y_j) + m < d(x_i, y_k)$$
$$\forall y_j \in Y_i^+, \ \forall y_k \in Y_i^-.$$  \hspace{1cm} (3.1)

Note that here and in the following, $d(x, y)$ will denote the Euclidean distance between image and text features in the embedding space.

Given a text input $y_{i'}$ (a phrase or sentence), we have analogous constraints in the other direction:

$$d(x_{i'}', y_{i'}) + m < d(x_{i'}', y_{i'})$$
$$\forall x_{i'}' \in X_{i'}^+, \ \forall x_{i'} \in X_{i'}^-,$$  \hspace{1cm} (3.2)

where $X_{i'}^+$ and $X_{i'}^-$ denote the sets of matched (positive) and non-matched (negative) visual data for $y_{i'}$.

These ranking constraints can be converted into a margin-based loss function:

$$L(X, Y) = \lambda_1 \sum_{i, j, k} [m + d(x_i, y_j) - d(x_i, y_k)]_+ + \lambda_2 \sum_{i', j', k'} [m + d(x_{i'}', y_{i'}) - d(x_{i'}', y_{i'})]_+,$$  \hspace{1cm} (3.3)
where \([t]_+ = \max(0,t)\). Our bi-directional ranking loss sums over all triplets (a target instance, a positive match, and a negative match) defined in constraints Eq. (3.1-3.2). For simplicity, we use a fixed margin \(m\) in our experiments. The weights \(\lambda_1\) and \(\lambda_2\) balance the strength of the ranking loss in each direction.

Optimizing the loss function requires enumerating triplets, which can be computationally expensive, especially for large datasets. Similar to [1, 62, 66], we use stochastic gradient descent methods to optimize the loss function and sample triplets within each mini-batch and our sampling strategy is loosely inspired by [69, 100]. Briefly, for each positive image-text pair \((x, y)\) in a mini-batch, we keep sampling triplets \((x, y, y')\) such that \((x, y')\) is a negative pair and \((y, x, x')\) such that \((x', y)\) is a negative pair.

### 3.1.3 Preserving Neighborhood Structure within Modalities

The many-to-many nature of correspondence for image-text tasks creates an additional aspect of complexity for training. For example, the same image region can be described by different phrases, while the same phrase can refer to different regions across the training set. These correspondences, in turn, induce neighborhood structure within each modality — which text (resp. image) pairs are similar because they correspond to the same image (resp. text) example. It is therefore interesting to see how this structure can help in learning the embedding.

**Neighborhood-preserving constraints.** In our conference paper [39], we proposed adding “structure constraints” (now termed neighborhood constraints) to our loss function. Let \(N(x_i)\) denote the neighborhood of \(x_i\), which is the set of images or regions described by the same text as \(x_i\). We would like to enforce a small margin of \(m\) between \(N(x_i)\) and any data point \(x\) outside of the neighborhood:

\[
d(x_i, x_j) + m < d(x_i, x_k) \\
\forall x_j \in N(x_i), \quad \forall x_k \notin N(x_i),
\]

(3.4)

Analogously to (3.4), we also define the neighborhood constraints for the text side:

\[
d(y_i', y_j') + m < d(y_i', y_k') \\
\forall y_j' \in N(y_i'), \quad \forall y_k' \notin N(y_i'),
\]

(3.5)

where \(N(y_i')\) is the set of descriptions, e.g. phrases or sentences, for the same visual data.

We then add terms corresponding to the above constraints to our baseline bi-directional...
ranking loss function in Eq. (3.3):

\[ L_{st}(X, Y) = \lambda_1 \sum_{i,j,k} [m + d(x_i, y_j) - d(x_i, y_k)]_+ + \lambda_2 \sum_{i,j,k} [m + d(x_i, x_j) - d(x_i, x_k)]_+ + \lambda_3 \sum_{i,j,k} [m + d(x_i, y_j) - d(x_i, y_k)]_+ + \lambda_4 \sum_{i,j,k} [m + d(y_i, y_j) - d(y_i, y_k)]_+ , \]  

(3.6)

where the sums are over all triplets defined in the constraints Eq. (3.1-3.2) and Eq. (3.4-3.5). Again we use a fixed margin \( m \). The weights \( \lambda_3 \) and \( \lambda_4 \) control the regularization power of the neighborhood-preserving terms, and small values give the best performance.

For phrase localization, we typically have multiple phrases corresponding to the same region (derived from multiple sentences corresponding to the same image), and multiple regions corresponding to the same phrase (these can be regions in different training images, or overlapping positive regions in the same image). Thus, both neighborhood-preserving terms given by Eq. (3.4) and (3.5) are meaningful. However, for image-sentence retrieval, while each image is paired with multiple sentences, Flickr30K and COCO datasets do not allow us to determine when the same sentence can apply to multiple images. Therefore, the image-view constraints (Eq. (3.4)) cannot be applied.

**Neighborhood sampling.** The use of neighborhood-preserving constraints requires that the same mini-batch contain more than one positive match for each target sample, i.e., at least two texts that are matched to the same image and vice versa. To ensure this, after performing regular triplet sampling, for any target image feature \( x \), we add new triplets to the mini-batch as necessary to ensure that there are at least two triplets \( (x, y_1, y_1') \) and \( (x, y_2, y_2') \) that pair the target \( x \) with different positive matches \( y_1 \) and \( y_2 \) – and analogously for any target text feature \( y \). This is done by searching all positive pairs that contain \( x \) (resp. \( y \)), which can be pre-computed.

In our original work [39], we introduced neighborhood sampling solely as a way to provide triplets upon which neighborhood constraints could be imposed. However, somewhat surprisingly, in image-sentence retrieval experiments, we have since found out that doing neighborhood sampling by itself, even without adding the corresponding terms to the objective function, already accounts for most of the improvement we observe with the neighborhood constraints. Accordingly, in Sections 3.3 and 3.4, we will evaluate the impact of our neighborhood sampling strategy apart from the constraints.
3.2 SIMILARITY NETWORK

The complexity of the embedding network’s objective function and triplet sampling strategy motivates us to consider as an alternative a more straightforward classification-based similarity network, shown on the right of Fig. (3.1). Our similarity network is inspired by the element-wise product baseline of Fukui et al. [80]. It shares the same architecture of the two branches as the embedding network, including FC, batch normalization, ReLU, and $L_2$ normalization. The network then merges the output of the two branches using element-wise product, followed by a series of FC and ReLU layers (we found three to give the best results).

For each input pair $(x_i, y_j)$, the similarity network generates a score $p_{ij}$ seeking to match the correct ground truth label (+1 for positive pairs and −1 for negative pairs). Our training objective is thus a logistic regression loss defined over the samples $\{x_i, y_j, z_{ij}\}$, where $z_{ij} = +1$ if $x_i$ and $y_j$ match each other, and −1 otherwise:

$$L(X, Y, Z) = \sum_{i,j} \log(1 + \exp(-z_{ij}p_{ij})).$$

(3.7)

To train the similarity network, we only need to sample positive and negative image-text pairs, which is much simpler and more efficient than sampling triplets. The only subtlety we found is that it is necessary to balance the number of positive and negative pairs in each mini-batch. Otherwise, the network will be dominated by the large number of negative pairs. More specifically, we maintain an equal number of positives and negatives in every mini-batch, though the sizes of different mini-batches can vary, especially for the phrase localization task.

3.3 EXPERIMENTS ON PHRASE LOCALIZATION

This section presents our experiments on the task of phrase localization on the Flickr30K Entities benchmark [12]. Flickr30K Entities augments the Flickr30K [14] image-sentence dataset, consisting of 31783 images with five sentences each, with annotations that link 244K mentions of distinct entities in sentences to 276K ground-truth bounding boxes.

To recap briefly, given a query noun phrase from an image caption and a set of region proposals from the same image, we rank the proposals using the region-phrase similarity scores produced by one of our trained networks. Consistent with Plummer et al. [12], for each image we generate region proposals using the category-independent EdgeBox method [101]. A proposal is considered to be a correct match for the query phrase if it has an Intersection over Union (IoU) score of at least 0.5 with the ground-truth bounding box for that phrase.
Accuracy is evaluated using Recall@K, defined as the percentage of phrases for which the correct region is ranked among the top K.

3.3.1 Training Set Construction

In our experience, properly defining positive/negative region-phrase pairs and sampling pairs and triplets of examples during training is crucial for achieving the best performance. Phrase localization is akin to object detection, in that region-phrase scores produced by the embedding should be sensitive not only to semantic correspondence, but also to localization quality, i.e., how much a given region proposal overlaps the ground truth box for a query phrase. By default, given a phrase from a description of a specific image, Flickr30K Entities annotations specify a unique ground truth region. Our conference paper [39], together with other related work like MCB [80], only used the ground truth boxes as positive regions during training. However, we have since realized that it is highly beneficial to augment the ground truth positive region with other proposals that have sufficiently high overlap with it. Specifically, we consider proposals having IoU > 0.7 with the ground truth as positive examples for the corresponding phrase, while proposals with IoU < 0.3 are marked as negative background regions. These threshold numbers can be tuned and chosen differently in practice, especially when there are not enough positive or negative regions.

As our experiments will demonstrate, positive region augmentation improves recall scores. It helps to improve the model’s robustness since ground truth regions are not available at test time, and is consistent with the way object detection is typically evaluated (a detection does not need to perfectly overlap the ground truth box to be considered correct).

3.3.2 Baselines and Comparisons

Our experiments systematically evaluate multiple components of our models, including network structure, sampling of the training set, and different components of the loss function for the embedding network. The full list of variants used in our comparisons is as follows.

Network Architecture. We are interested in how our networks benefit from being able to learn a nonlinear mapping in each branch. For this, we compare two variants:

- **Linear branch structure**: only keeping the first layers in each branch (i.e. the ones with parameters $W_1$, $V_1$, as shown in Fig. (3.1)) immediately followed by $L2$

\footnote{Consistent with [12], for plural entities associated with multiple boxes, we form one big bounding box containing all the instances. We also exclude non-visual phrases, i.e., phrases that do not correspond to a bounding box.}
• **Nonlinear branch structure**: branches consisting of two FC layers with ReLU, batch normalization and \(L_2\) normalization.

**Selecting Positive Pairs.** We evaluate how positive example augmentation contributes to the performance of phrase localization. We compare the vanilla scheme without augmentation to our scheme described in Section 3.3.1:

• **Single positive**: using the ground truth region for a phrase as the single positive example.

• **Augmented positive**: augmenting ground truth regions with other regions having larger IoUs. The selection of positive threshold depends on the quality of region proposals. It can be tuned and changed in practice.

**Embedding Loss Functions.** In principle, phrase localization is a single-directional task of retrieving image regions given a query phrase, so we want to know whether we can derive an additional benefit by using a bi-directional loss function:

• **Single-directional**: only using the phrase-to-region loss from Eq. (3.6). This is done by setting \(\lambda_1 = 0, \lambda_2 = 1, \lambda_3 = 0, \lambda_4 = 0\).

• **Bi-directional**: using the bi-directional loss from Eq. (3.6). This is done by setting \(\lambda_1 = 1, \lambda_2 = 6, \lambda_3 = 0, \lambda_4 = 0\). These parameter values have been tuned on our validation set.

**Neighborhood Sampling and Constraints.** Flickr30K Entities dataset includes 130K pairs of region-phrase correspondences, with 70K unique phrases and 80K unique regions. In general, one phrase can correspond to many regions and vice versa. We are interested in how multiple matches to the same region (resp. phrase) can help the task, as described in Section 3.1.3:

• **Neighborhood sampling**: using the sampling strategy of Section 3.1.3 to augment standard triplet sampling.

• **Neighborhood constraints**: using the full loss function of Eq. (3.6). This is done by setting \(\lambda_3 = 0.1, \lambda_4 = 0.1\). This requires the use of neighborhood sampling.
3.3.3 Implementation details

Following Rohrbach et al. [81] and Plummer et al. [45], we use Fast R-CNN features [102] from the VGG network [103] fine-tuned on the union of the PASCAL 2007 and 2012 train-val sets [104]. To be consistent with [45], we extract 4096D features from a single crop of an image region.

For phrases, we use the Fisher Vector (FV) encoding [105] as suggested by Klein et al. [2]. We start from 300-dimensional word2vec features [97] and apply ICA as in [2] to construct a codebook with 30 centers. The resulting FV representation has dimension $300 \times 30 \times 2 = 18000$. For simplicity, we only use the Hybrid Gaussian-Laplacian Mixture Model (HGLMM) from [2] rather than the combined HGLMM+GMM model. To save memory and training time, we perform PCA on these 18000-dimensional vectors to reduce the dimension to 6000. A disadvantage of HGLMM is that it is a complex and nonlinear hand-crafted text feature. However, as we showed in the conference version of this work [39], we can obtain very similar results on top of basic tf-idf features. In this chapter, our main focus is on the design and training of two-branch networks, so we omit the evaluation of different text features.

Both the embedding and similarity networks use the same configurations for the two branches. The image branch has two FC layers with weight matrices $W_1$ and $W_2$ having sizes $4096 \times 1024$ and $1024 \times 512$. The text branch has two FC layers with weight matrices $V_1$ and $V_2$ having sizes $6000 \times 1024$ and $1024 \times 512$. Thus, the embedding network projects image and text features into a 512-dimensional latent space. For the similarity network, the outputs of the two branches get combined by the element-wise product layer that doesn’t change the dimensionality, followed by three additional FC layers. The similarity network outputs a scalar score trained with logistic regression as described in Section 3.2.

We train our networks using the Adam optimizer [106] with an initial small learning rate. We use a mini-batch of 100 image-phrase pairs. Each epoch thus needs around 4000 iterations. Both our similarity (using single positive and using augmented positive) and embedding networks (no neighborhood terms or sampling) converge after 32000 iterations (around 8 epochs). It should be noted that, for our embedding network trained with bi-directional terms and neighborhood constraints (corresponding to the last two rows in Table (3.1)(b)), we adopt two-stage training to save time and have a fair comparison with single directional baselines: We first train the network for around 32000 iterations with single directional setting and neighborhood sampling, then add the neighborhood terms and the other direction term, and fine-tune the checkpoint model for another epoch. For comparisons, we list results of those single directional settings in Table (3.1)(b) by testing their 32000 iterations checkpoints.
3.3.4 Result Analysis

At test time, we treat phrase localization as the task of retrieving regions matching a query phrase (assumed to be present in the image) from a set of region proposals. For the embedding network, the query phrase and each candidate region are passed through the respective branches to compute their embedded representation, and Euclidean distance (equivalently, cosine similarity) is used as the similarity score. For the similarity network, the score is predicted directly using the logistic formulation. In both cases, we rank regions in decreasing order of similarity to the query and report Recall@K, or the percentage of queries for which the correct match has rank of at most K. A region proposal is considered to be a correct match if it has IoU of at least 0.5 with the ground-truth bounding box for that phrase. We follow the evaluation protocols provided by Plummer et al. [12].

Table (3.1) shows the results of our embedding and similarity networks, in comparison to several state-of-the-art methods. Among them, CCA [12], GroundeR [81] and MCB [80] are representative linear and deep models for this task. We also compare to the structured matching system of Wang et al. [42], which uses a single-layer version of our embedding network from [39] combined with global optimization to find a joint assignment of phrases to all image regions while satisfying certain relations derived from the sentence. As we can see from Table (3.1)(a), CCA, which is trained only on positive region-phrase pairs, already establishes a strong baseline. MCB gives the best results among all previous methods.

Table (3.1)(b) gives the ablation study results for the embedding network. Using a nonlinearity within each branch improves performance, and using a nonlinearity within each branch improves performance, and using bi-directional instead of single-directional loss function does not give too much difference. For example, $R@1$ increases from 50.18 to 51.00 but both $R@5$ and $R@10$ seem to be comparable.

Adding positive region augmentation improves $R@1$ by around 5%. We also observe that neighborhood sampling achieves better recall scores than standard triplet sampling when they are both trained with 32000 iterations in the single directional setting.

Finally, adding neighborhood constraints gives a slight drop in $R@1$ but further minor improvements in $R@5$ and comparable performance in $R@10$. Interestingly, contrary to our original expectations [39], it is the composition of the mini-batches, not the imposition of a specific neighborhood-preserving loss penalty during training, that seems to be responsible for most of the improvements in performance. However, image-sentence retrieval experiments in Section 3.4 will demonstrate that neighborhood constraints have a more noticeable effect on the accuracy of retrieval within the same modality, i.e., sentence-to-sentence retrieval as opposed to image-sentence retrieval.
Table 3.1: Phrase localization results on Flickr30K Entities. We use 200 EdgeBox proposals, for which the recall upper bound is $R@200 = 84.58$.

Table (3.1)(c) reports the accuracy of the similarity network with and without nonlinearity in each branch, with and without positive region augmentation. Consistent with the embedding network results, the nonlinear models improve $R@1$ over their linear versions by about 2%, but positive region augmentation gives an even bigger improvement of about 5%. The highest $R@1$ achieved by the similarity network is 51.05, which is almost identical to that is achieved by our best embedding networks. We also checked the performance of the similarity network with a different number of FC layers after the element-wise product, though we do not list the complete numbers in Table (3.1) to avoid clutter. With a single FC layer, we get a significantly lower $R@1$ of 36.61, and with two FC layers, we get 49.39, which is almost on par with three layers.

Fig. (3.2) shows examples of phrase localization results in three images with our best model (similarity networks with augmented positives) compared to the CCA model.

3.4 EXPERIMENTS ON IMAGE-SENTENCE RETRIEVAL

This section evaluates our networks on the task of bi-directional image-sentence retrieval. Given a query image (resp. sentence), the goal is to find corresponding sentences (resp.
images) from the dataset. In addition to the Flickr30K dataset, here we perform experiments on the larger COCO dataset [13], consisting of 123287 images (the combination of released train2014 and val2014 from COCO website [51]) with five sentences each. COCO does not include comprehensive region-phrase correspondence, so we can use it for image-sentence retrieval only.

3.4.1 Training Set Construction

For the embedding network, we start by randomly permuting the data into mini-batches consisting of 500 positive image-sentence pairs. Then for each positive image-sentence pair \((x, y)\), we enumerate triplets \((x, y, y')\) where \(y'\) is a sentence in the same mini-batch not associated with \(x\), as well as triplets \((y, x, x')\) where \(x'\) is an image in the mini-batch not associated with \(y\). In both cases, we keep at most \(K = 10\) triplets with highest nonnegative loss. For neighborhood sampling, we need to make sure that given a target image \(x\), a mini-batch has at least two triplets \((x, y_1, y'_1)\) and \((x, y_2, y'_2)\) where \(y_1\) and \(y_2\) are both sentences associated with \(x\). Because we typically cannot identify more than one image described by the same sentence, we cannot do the other direction of neighborhood sampling.

For the similarity network, for each positive pair \((x, y)\), we can only generate a negative pair \((x, y')\) by randomly sampling a sentence not associated with the image. Note, however, that we cannot guarantee that \(x\) and \(y'\) are semantically incompatible with each other, since for any given image, our image-sentence datasets probably contain a number of sentences not associated with it that could still describe it accurately. This is not a major issue for phrase localization, since we adopt the alternative strategy of sampling negative pairs \((x', y)\) with the “negative” regions \(x'\) constrained to have low overlap with \(x\) in the same image. It also doesn’t play as much of a role for the embedding network, since the triplet objective merely tries to make sure that the captions actually written for a given image are closer to it than other sentences, not to push down the similarity of the other sentences to the image to a fixed low target value. Based on this reasoning, we expect the similarity network to perform poorly for image-sentence retrieval, and will omit its comparisons in the following experiments. The focus of this section is to study how each component of embedding network will work for the image-sentence retrieval task.

3.4.2 Baselines and Comparisons

We demonstrate the impact of different components of our models by reporting results for the following variants.
Linear vs. Nonlinear Branch Structure. The same way as in the phrase localization experiments, we want to see the difference made by having one vs. two fully connected layers within each branch.

Embedding Loss Functions. Image-sentence retrieval is a bi-directional retrieval task, so we want to see whether bi-directional loss can give a bigger improvement over the single-directional loss than that on phrase localization task.

- **Single-directional**: in Eq.(3.6), only using the single direction (from image to sentences) by setting $\lambda_1 = 1, \lambda_2 = 0, \lambda_3 = 0, \lambda_4 = 0$.
- **Bi-directional**: in Eq.(3.6), set $\lambda_1 = 1, \lambda_2 = 1.5, \lambda_3 = 0, \lambda_4 = 0$. These parameter values are determined on the validation set.

Neighborhood Sampling and Constraints. In both Flickr30K and COCO datasets, each image is associated with five sentences. Therefore, we can try to enforce neighborhood structure on the sentence space. We cannot do it on the image space since in the Flickr30K and COCO datasets we do not have direct supervisory information about multiple images that can be described by the same sentence. Thus, in Eq.(3.6), we always have $\lambda_3 = 0$.

- **Neighborhood sampling**: using the neighborhood sampling strategy (see Section 3.1.3) to replace standard triplet sampling.
- **Neighborhood constraints**: using the full loss function as in Eq.(3.6). This is done by setting $\lambda_2 = 1.5, \lambda_4 = 0.05$. We always use neighborhood sampling in this case.

3.4.3 Implementation Details

To represent whole images, we follow the implementation details in [2, 12]. Given an image, we extract the 4096-dimensional activations from the 19-layer ImageNet-trained VGG model [103]. Following standard procedure, the original $256 \times 256$ image is cropped in ten different ways into $224 \times 224$ images: the four corners, the center, and their x-axis mirror image. The mean intensity is then subtracted from each color channel, the resulting images are encoded by the network, and the network outputs are averaged. The output dimensions of the two FC layers on the image side are 2048 and 512.

To represent sentences, we continue to rely on the same orderless HGLMM features as for phrase localization, PCA-reduced to 6000 dimensions, with output dimensions of the two FC layers on the text side also set to 2048 and 512. However, while we could be reasonably assured that these features do not lose much information when representing short phrases,
their suitability for longer sentences is less clear. Therefore, in this section we also evaluate a recurrent sentence representation learned by a LSTM [25]. We start with a one-hot encoding with vocabulary size of 11,263 (COCO) and 8,569 (Flickr30K), which are the numbers of words in the respective training sets. This input gets projected into a word embedding layer of dimension 256, and the LSTM hidden space dimension is 512. The hidden space output is used as the input to the text branch of our embedding network. Accordingly, we change the first FC layer of the text branch to accept 512-dimensional input. The output dimension of the embedding space is also 512. During training, the LSTM parameters are optimized jointly with the rest of the network parameters by back-propagating the embedding loss.

For the subsequent experiments, we train our networks using Adam with starting learning rate of 0.0001 for HGLMM features and 0.0002 for LSTM features. We use a Dropout layer after ReLU with probability = 0.5 (note that in the phrase localization experiments, we did not use Dropout).

3.4.4 Result Analysis

For evaluation of bi-directional image-sentence retrieval, we follow the same protocols as other recent works [2, 12]. Given the test set of 1000 images and 5000 corresponding sentences, we use our networks to score images given query sentences and vice versa, and report performance as Recall@K \((K = 1, 5, 10)\), or the percentage of queries for which at least one correct ground truth match was ranked among the top \(K\) matches. For Flickr30K, we use the same random split as Plummer et al. [12]. For COCO, like [1, 2], we randomly generate the splits that contain 113287 images with their corresponding sentences for training, 1000 images and their corresponding sentences for testing and the remaining images and their corresponding sentences for validation.

Results on the Flickr30K and COCO datasets are given in Tables (3.2) and (3.3), respectively. Parts (a) of the tables list the numbers reported by recent competing methods. The most relevant baseline for our embedding network is CCA (HGLMM) [2, 12], since it uses the same underlying feature representations for images and sentences. Parts (b) of the tables give results for our embedding networks, and the trends are largely similar to those of Table (3.1). Going from single-directional to bi-directional constraints improves the accuracy by a bigger amount for sentence-to-image retrieval. Neighborhood sampling is effective and can generally improve over conventional triplet sampling around in R@1 across the table, and adding neighborhood constraints does not show significant further improvements. In Table (3.3)(b), adding neighborhood constraints improves the R@1 in both directions but shows a small drop for R@10. However, we will show in Section 3.4.5 that adding neighborhood
<table>
<thead>
<tr>
<th>Methods on Flickr30K</th>
<th>Img-to-sen</th>
<th>Sen-to-img</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td>(a) State of the art</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deep CCA [57]</td>
<td>27.9</td>
<td>56.9</td>
</tr>
<tr>
<td>mCNN(ensemble) [43]</td>
<td>33.6</td>
<td>64.1</td>
</tr>
<tr>
<td>m-RNN-vgg [22]</td>
<td>35.4</td>
<td>63.8</td>
</tr>
<tr>
<td>Mean vector [2]</td>
<td>24.8</td>
<td>52.5</td>
</tr>
<tr>
<td>CCA (FV HGLMM) [2]</td>
<td>34.4</td>
<td>61.0</td>
</tr>
<tr>
<td>CCA (FV GMM+HGLMM) [2]</td>
<td>35.0</td>
<td>62.0</td>
</tr>
<tr>
<td>CCA (FV HGLMM) [12]</td>
<td>36.5</td>
<td>62.2</td>
</tr>
<tr>
<td>Two-way Nets [44]</td>
<td>49.8</td>
<td>67.5</td>
</tr>
<tr>
<td>(b) Embedding Network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 3.2: Bi-directional retrieval results. The numbers in (a) come from published papers, and the numbers in (b-c) are results of our embedding network. Note that the Deep CCA results in [57] were obtained with AlexNet [83].

constraints can consistently improve within-view retrieval.

Parts (c) of Tables (3.2) and (3.3) give the results for our full embedding network with LSTM sentence encoding, which turns out to be comparable to, or slightly worse than, the HGLMM feature.

3.4.5 Sentence-to-sentence Retrieval

Our experiments on the embedding network both for phrase localization and image-sentence retrieval have shown that neighborhood sampling can give some improvements even without adding neighborhood constraint terms to the triplet loss. However, it is still unclear how neighborhood constraints change the latent embedding space. Therefore, in this section, instead of only looking at cross-modal retrieval, we show how neighborhood constraints can improve performance for the within-view task of sentence-to-sentence retrieval: given a query sentence, retrieve other sentences that correspond to the same image. For the evaluation metric, we still use R@K. We also use the same training/val/testing splits as in the previous section. Results on Flickr30K and COCO datasets are listed in Table (3.4). It
Table 3.3: Bi-directional retrieval results on the COCO 1000-image test set.

<table>
<thead>
<tr>
<th>Methods on COCO</th>
<th>Image-to-sen</th>
<th>Sen-to-image</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R@1</td>
<td>R@5</td>
</tr>
<tr>
<td>(a) State of the art</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean vector [2]</td>
<td>33.2</td>
<td>61.8</td>
</tr>
<tr>
<td>CCA (HGLMM) [2]</td>
<td>37.7</td>
<td>66.6</td>
</tr>
<tr>
<td>CCA [2]</td>
<td>39.4</td>
<td>67.9</td>
</tr>
<tr>
<td>DVSA [4]</td>
<td>38.4</td>
<td>69.9</td>
</tr>
<tr>
<td>m-RNN-vgg [22]</td>
<td>41.0</td>
<td>73.0</td>
</tr>
<tr>
<td>mCNN(ensemble) [43]</td>
<td>42.8</td>
<td>73.1</td>
</tr>
<tr>
<td>LayerNorm [107]</td>
<td>48.5</td>
<td>80.6</td>
</tr>
<tr>
<td>OrderEmbedding [108]</td>
<td>46.7</td>
<td>-</td>
</tr>
<tr>
<td>Two-way Nets [44]</td>
<td>55.8</td>
<td>75.2</td>
</tr>
<tr>
<td>(b) Embedding Network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(nonlinear, bi-directional)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>(c) Embed.(LSTM)</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

can be seen that adding neighborhood constraints on top of neighborhood sampling provides a more convincing gain for within-view retrieval than for cross-view retrieval. This behavior can be useful for practical multi-media systems where both tasks are required at the same time.

### 3.5 CONCLUSION

This chapter has studied state-of-the-art two-branch network architectures for region-to-phrase and image-to-sentence matching. Our first architecture, the embedding network, works by explicitly learning a non-linear mapping from input image and text features into a joint latent space in which corresponding image and text features have high similarity. This network works well for both image-sentence and region-phrase tasks, though its objective consists of multiple terms and relies on somewhat costly and intricate triplet sampling. We investigated triplet sampling within mini-batches in detail and showed that the way it is done can have a significant impact on performance, even without changing the objective function. Our second architecture, the similarity network, tries to directly predict whether an input image and text feature are similar or dissimilar. We analyze that this network can serve as an attractive alternative to the embedding network for region-phrase matching, but not for image-sentence retrieval, revealing an interesting difference between the two tasks. This also indicates that, with a better designed negative sampling strategy for similarity
Table 3.4: Sentence-to-sentence retrieval on Flickr30K and COCO datasets.

network, its performance on retrieval tasks might have the potential to be improved in the future.
A group of eight campers sit around a fire pit trying to roast marshmallows on their sticks.

Figure 3.1: Taking the phrase localization task as an example, we show the architectures of the two-branch networks used in this chapter. Left column: given the phrase “a fire pit” from the image caption, sets of positive regions (purple) and negative regions (blue) are extracted from the training image. The positive regions are defined as ones that have a sufficiently high overlap with the ground truth (dashed white rectangle). $X$ and $Y$ denote the feature vectors describing image regions and phrases, respectively. Middle: the embedding network. Each branch consists of fully connected (FC) layers with ReLU nonlinearities between them, followed by L2 normalization at the end. We train this network with a maximum-margin triplet ranking loss that pushes positive pairs closer to each other and negative pairs farther (Section 3.1). Right: the similarity network. As in the embedding network, the branches consist of two fully connected layers followed by L2 normalization. Element-wise product is used to aggregate features from two branches, followed by several additional fully connected (FC) layers. The similarity network is trained with the logistic regression loss function, with positive and negative image-text pairs receiving labels of “+1” and “-1” respectively (Section 3.2).
A little girl stands on the fence while peeking through it to look at the horse.

A large yellow dog leaps into the air to catch his Frisbee.

A man is using a chainsaw to carve a wooden sculpture.

Figure 3.2: Example phrase localization results. For each image and reference sentence, phrases and best-scoring corresponding regions are shown in the same color. The first row shows the output of the CCA method [12] and the second row shows the output of our best model (similarity network trained with augmented positive regions). In the first example, our method finds a partially correct bounding box for the horse while CCA completely misses it; in the second (middle) example, our method gives a more accurate bounding box for the frisbee. In the third (right) example, our method gives marginally better boxes for the chainsaw and wooden sculpture.
Two young wet boys playing in the sand on a beach.
Two little boys play with sand on a beach.
Two children playing on a beach in the sand with the ocean in the background.
Two children are playing with sand on a beach near the ocean.
A woman her children and a dog are on a beach.

Figure 3.3: Examples of image-to-sentence retrieval for some test images in Flickr30K. For each query image, we show the top five sentences retrieved by our best model. Correct matches are shown in blue. In the last example, all five retrieved sentences are incorrect.

A group of people are rock climbing on a rock climbing wall.
A man in a plaid shirt and blue jeans stands on scaffolding.
Two women one in a fur coat walk down a sidewalk with Umbrellas.

Figure 3.4: Examples of sentence-to-image retrieval on the Flickr30K test set. For each sentence query, we show the top five retrieved images. The ground-truth images are marked with blue borders.
CHAPTER 4: GENERATING IMAGE DESCRIPTION USING A VARIATIONAL AUTO-ENCODER WITH AN ADDITIVE GAUSSIAN ENCODING SPACE

In Chapter 3, we introduced the two-branch neural networks for joint image-language tasks. Our model mapped images and language information onto the shared space and cross-modality search was performed by finding the nearest neighbors in the joint space. However, this approach can not generate sentences from the encoding space. In this chapter, we present a deep generative model that is able to learn the latent encoding space, from which sentences can be decoded conditioned on the input image content.

Our starting point is the work of Jain et al. [109], who trains a “vanilla” CVAE to generate questions given images. At training time, given an image and a sentence, the CVAE encoder samples a latent $z$ vector from a Gaussian distribution in the encoding space whose parameters (mean and standard deviation) come from a single fixed Gaussian prior (zero mean, unit variance). This $z$ vector is then fed into a decoder that uses it, together with the features of the input image, to generate a question. The encoder and the decoder are jointly trained to maximize (an upper bound on) the likelihood of the reference questions given the images. At test time, the decoder is seeded with an image feature and different $z$ samples, so that multiple $z$ vectors result in multiple sentences.

For the task of image captioning, we find that having a single fixed Gaussian prior for all the $z$ vectors limits the expressiveness and the accuracy of the generative model. Especially, this single Gaussian prior can make the learning of VAE models easily collapse to single mode. Instead, we create a set of $K$ Gaussian priors on the latent $z$ space with different means and standard deviations, corresponding to different “modes” or types of image content.

Concretely, we identify these modes with specific object categories, such as “dog” or “cat”. If both “dog” and “cat” are detected in an image, we encourage the generated captions to capture both aspects of “dog” and “cat”.

To achieve this goal, we propose two different ways of modeling the latent $z$ space in this chapter. We propose to use a Gaussian Mixture prior instead of the original single Gaussian prior. The Gaussian mixture model (GMM) can capture more than one semantic mode since it represents the distribution of $z$ as a weighted sum of multiple Gaussian distributions. However, due to the intractability of Gaussian mixtures in the VAE framework, we are encouraged to propose another novel prior, which we call Additive Gaussian prior (AG). This prior adds multiple semantic aspects in the $z$ space directly, lending itself to a tractable learning scheme. Our CVAE formulation with this additive Gaussian prior (AG-CVAE) is able to model a richer, more flexible encoding space, resulting in captions
Figure 4.1: Example output of our proposed AG-CVAE approach compared to an LSTM baseline (see Section 4.3 for details). For each method, we show top five sentences following consensus re-ranking [38]. The captions produced by our method are both more diverse and more accurate.

that are simultaneously more diverse and more accurate than those produced by a strong LSTM baseline, as illustrated in Fig. (4.1). Furthermore, the additive prior gives us an interpretable mechanism for controlling the captions based on the image content, as shown in Fig. (4.5). Experiments of Section 4.3 will show that our approach outperforms LSTMs and other CVAE baselines on the challenging COCO dataset [13].

4.1 CONDITIONAL VARIATIONAL AUTO-ENCODERS

Our proposed framework for image captioning extends the standard variational autoencoder [110] and its conditional variant [111]. We briefly set up the necessary background here.

Variational auto-encoder (VAE): Given samples $x$ from a dataset, VAEs aim at mod-
eling the data likelihood $p(x)$. To this end, VAEs assume that the data points $x$ cluster around a low-dimensional manifold parameterized by embeddings or encodings $z$. To obtain the sample $x$ corresponding to an embedding $z$, we employ the decoder $p(x|z)$ which is often based on deep nets. Since the decoder’s posterior $p(z|x)$ is not tractably computable we approximate it with a distribution $q(z|x)$ which is referred to as the encoder. Taking together all those ingredients, VAEs are based on the identity

$$\log p(x) - D_{KL}[q(z|x), p(z|x)] = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{KL}[q(z|x), p(z)], \quad (4.1)$$

which relates the likelihood $p(x)$ and the conditional $p(z|x)$. It is hard to compute the KL-divergence $D_{KL}[q(z|x), p(z|x)]$ because the posterior $p(z|x)$ is not readily available from the decoder distribution $p(x|z)$ if we use deep nets. However, by choosing an encoder distribution $q(z|x)$ with sufficient capacity, we can assume that the non-negative KL-divergence $D_{KL}[q(z|x), p(z|x)]$ is small. Thus, we know that the right-hand-side is a lower bound on the log-likelihood $\log p(x)$, which can be maximized w.r.t. both encoder and decoder parameters.

**Conditional variational auto-encoders (CVAE):** In tasks like image captioning, we are interested in modeling the conditional distribution $p(x|c)$, where $x$ are the desired descriptions and $c$ is some representation of content of the input image. The VAE identity can be straightforwardly extended by conditioning both the encoder and decoder distributions on $c$.

Training of the encoder and decoder proceeds by maximizing the lower bound on the conditional data-log-likelihood $p(x|c)$, i.e.,

$$\log p_\theta(x|c) \geq \mathbb{E}_{q_\phi(z|x,c)}[\log p_\theta(x|z,c)] - D_{KL}[q_\phi(z|x,c), p(z|c)], \quad (4.2)$$

where $\theta$ and $\phi$, the parameters for the decoder distribution $p_\theta(x|z,c)$ and the encoder distribution $q_\phi(z|x,c)$ respectively.

In practice, the following stochastic objective is typically used:

$$\max_{\theta,\phi} \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(x|z^i,c) - D_{KL}[q_\phi(z|x,c), p(z|c)], \quad \text{s.t. } \forall i \ z^i \sim q_\phi(z|x,c).$$

It approximates the expectation $\mathbb{E}_{q_\phi(z|x,c)}[\log p_\theta(x|z,c)]$ using $N$ samples $z^i$ drawn from the approximate posterior $q_\phi(z|x,c)$ (typically, just a single sample is used). Backpropagation through the encoder that produces samples $z^i$ is achieved via the reparameterization trick [110], which is applicable if we restrict the encoder distribution $q_\phi(z|x,c)$ to be, e.g., a
4.2 GAUSSIAN MIXTURE PRIOR AND ADDITIVE GAUSSIAN PRIOR

Our key observation is that the behavior of the trained CVAE crucially depends on the choice of the prior \( p(z|c) \). The prior determines how the learned latent space is structured, because the KL-divergence term in Eq. (4.2) encourages \( q_\phi(z|x,c) \), the encoder distribution over \( z \) given a particular description \( x \) and image content \( c \), to be close to this prior distribution.

Motivated by the above considerations, we could encourage the latent \( z \) space to have a multi-modal structure composed of \( K \) modes or clusters, each corresponding to different types of image content. In our current work, for concreteness, we identify these clusters with a set of object categories that can be reliably detected automatically, such as “car,” “person,” or “cat.” The COCO dataset, on which we conduct our experiments, has direct supervision for 80 such categories. Note, however, our formulation is general and can be applied to other definitions of modes or clusters, including latent topics automatically obtained in an unsupervised fashion.

Given an image \( I \), we assume that we can obtain a distribution \( c(I) = (c_1(I), \ldots, c_K(I)) \), where the entries \( c_k \) are nonnegative and sum to one, representing the topics or clusters that are present in the image. Then we can model \( p(z|c) \) as a Gaussian mixture with weights \( c_k \) and components with means \( \mu_k \) and standard deviations \( \sigma_k \). This model will be called GMM-CVAE in the following and the details of this model are listed below.

**GMM-CVAE:** by choosing the \( p(z|c) \) as a Gaussian mixture distribution, we have

\[
p(z|c) = \sum_{k=1}^{K} c_k N \left( z | \mu_k, \sigma_k^2 \right), \quad (4.3)
\]

where \( c_k \) is defined as the weights above. \( \mu_k \) represents the mean vector of \( k \)-th component. In practice, for all components, we use the same standard deviation \( \sigma \).

It is not directly tractable to optimize Eq. (4.2) with the above GMM prior. We have to approximate the KL divergence term stochastically [112]. In each iteration during training, we first draw a discrete component \( k \) according to the cluster probability \( c(I) \), and then sample \( z \) from the resulting Gaussian component. And therefore, the intractable KL divergence term in Eq. (4.2) is now approximated by the following:
Figure 4.2: Sampling from a standard VAE with a Gaussian mixture prior (a,b) is contrasted to our prior formulation (c,d) which is suitable in case multiple clusters can co-occur. GMM-CVAE switches from one cluster center to another, while our AG-CVAE encourages the embedding $z$ for an image to be close to the center of its objects’ means.

$$D_{KL}[q_\phi(z|x, c_k), p(z|c_k)] = \log \left( \frac{\sigma_k}{\sigma_\phi} \right) + \frac{1}{2\sigma^2} \mathbb{E}_{q_\phi(z|x, c_k)} \left[ \|z - \mu_k\|_2^2 \right] - \frac{1}{2}$$

$$= \log \left( \frac{\sigma_k}{\sigma_\phi} \right) + \frac{\sigma^2_\phi + \|\mu_\phi - \mu_k\|_2^2}{2\sigma^2_k} - \frac{1}{2}. \quad \forall k \quad c_k \sim c(I) \quad (4.4)$$

By plugging the KL term into the Eq. (4.2), we have the following approximate stochastic objective function. In practice, we find this strategy to be stable and to converge well.

$$\max_{\theta, \phi} \frac{1}{N} \sum_{i=1}^{N} \log p_\theta(x|z^i, c) - D_{KL}[q_\phi(z|x, c_k), p(z|c_k)] \quad (4.5)$$

where $c_k \sim c(I)$ and $z^i \sim q_\phi(z|x, c_k)$. Only one $z^i$ ($N = 1$) is sampled in this chapter.

In order to model the $z$ space directly to reflect the additive nature of semantic concepts, we propose a novel conditioning mechanism with an additive prior, named as “Additive Gaussian (AG) Prior”.

**AG-CVAE:** If an image contains several objects with weights $c_k$, each corresponding to means $\mu_k$ in the latent space, we want the mean of the encoder distribution to be close to the linear combination of the respective means with the same weights:

$$p(z|c) = N \left( z \left| \sum_{k=1}^{K} c_k \mu_k, \sigma^2 I \right. \right), \quad (4.6)$$

where $\sigma^2 I$ is a spherical covariance matrix with $\sigma^2 = \sum_{k=1}^{K} c^2_k \sigma^2_k$. We call the resulting model AG-CVAE. Fig. (4.2) illustrates the difference between the GMM-CVAE and AG-CVAE
Figure 4.3: Illustration of our encoder (left) and decoder (right). For the encoder, we omit those lines that sum over all intermediate time step states $h$. See text for details.

models.

In order to train the AG-CVAE model using the objective of Eq. (4.2), we need to compute the KL-divergence $D_{KL}[q_\phi(z|x,c), p(z|c)]$ where $q_\phi(z|x,c) = \mathcal{N}(z | \mu_\phi(x,c), \sigma^2_\phi(x,c)I)$ and the prior $p(z|c)$ is given by Eq. (4.6). Its analytic expression can be derived as

$$D_{KL}[q_\phi(z|x,c), p(z|c)] = \log \left( \frac{\sigma}{\sigma_\phi} \right) + \frac{1}{2\sigma^2_\phi} \mathbb{E}_{q_\phi} \left[ \left\| z - \sum_{k=1}^K c_k \mu_k \right\|^2 \right] - \frac{1}{2}$$

$$= \log \left( \frac{\sigma}{\sigma_\phi} \right) + \frac{\sigma^2_\phi + \| \mu_\phi - \sum_{k=1}^K c_k \mu_k \|^2}{2\sigma^2} - \frac{1}{2}$$

According to Eq. (4.2) and Eq. (4.6), we obtain the following objective:

$$\max_{\theta, \phi} \frac{1}{N} \sum_{i=1}^N \log p_\theta(x|z^i, c) - D_{KL}[q_\phi(z|x,c), \mathcal{N}(z | \sum_{k=1}^K c_k \mu_k, \sigma^2)], \quad \forall i \ z^i \sim q_\phi(z|x,c), \quad (4.7)$$

which we optimize w.r.t. the parameters $\theta, \phi$ using stochastic gradient descent (SGD). In principle, the prior parameters $\mu_k$ and $\sigma_k$ can also be trained, but we obtained good results by keeping them fixed (the means are drawn randomly and all standard deviations are set to the same constant, as will be further explained in Section 4.3).

The aforementioned procedure is generally applicable to any task with co-occurring conditions. Since we are interested in sampling language descriptions for natural images, we now need to define our specific architectures for the encoder and decoder, which are shown in Fig. (4.3).
The **encoder** uses an LSTM to map a given caption, its corresponding image $I$ as well as the respective object detections, *i.e.*, the cluster or object vector $c(I)$, into a point in the latent space. More specifically, the LSTM receives the image feature in the first step, the cluster vector in the second step, and then the caption words one by one. We sum together all time step hidden states $h$. And this summed hidden state is transformed (by embedding matrix $W_{ck}$ in Fig. (4.3)) into $K$ mean vectors, $\mu_{\phi k}$, and $K$ log variances, $\log \sigma_{\phi k}^2$, using a linear layer for each. For AG-CVAE, the $\mu_{\phi k}$ and $\sigma_{\phi k}^2$ are then summed with weights $c_k$ and $c_k^2$ respectively to generate the desired $\mu_{\phi}$ and $\sigma_{\phi}^2$ outputs. The encoder is used at training time only, and the input cluster vectors are produced from ground truth object annotations.

The **decoder** uses a different LSTM that receives as input first the image feature, then the cluster vector, then a $z$ vector sampled from the conditional distribution of Eq. (4.6).

Next, it receives a “start” symbol and proceeds to output a sentence word by word until it produces an “end” symbol.

During training, its $c(I)$ inputs are derived from the ground truth, same as for the encoder, and the log-loss is used to encourage reconstruction of the provided ground-truth caption. At test time, ground truth object vectors are not available, so we rely on automatic object detection, as explained in Section 4.3.

### 4.3 EXPERIMENTS ON IMAGE CAPTIONING

#### 4.3.1 Implementation Details

We test our methods on the COCO dataset [13], which is the largest “clean” image captioning dataset available to date. The current (2014) release contains 82,783 training and 40,504 validation images with five reference captions each, but many captioning works repartition this data to enlarge the training set. We follow the train/val/test split released by [22]. It allocates 118,287 images for training, 4,000 for validation, and 1,000 for testing.

**Features.** As image features, we use 4,096-dimensional activations from the VGG-16 network [103]. The cluster or object vectors $c(I)$ are 80-dimensional, corresponding to the 80 COCO object categories. At training time, $c(I)$ consist of binary indicators corresponding to ground truth object labels, rescaled to sum to one. For example, an image with labels “person”, “car” and “dog” results in a cluster vector with weights of $1/3$ for the corresponding objects and zeros elsewhere. For test images $I$, $c(I)$ are obtained automatically through object detection. We train a Faster R-CNN detector [113] for the COCO categories using our train/val split by fine-tuning the VGG-16 net [103]. At test time, we use a threshold
of 0.5 on the per-class confidence scores output by this detector to determine whether the image contains a given object (i.e. all the weights are once again equal).

**Baselines.** LSTM-based captioners such as NeuralTalk2 [24] and Google Show and Tell [84] are strong baselines for our approach. Our LSTM baseline is obtained by deleting the \( z \) vector input from the decoder architecture shown in Fig. (4.3). To generate different candidate sentences using the LSTM, we use beam search with a width of 10. Our second baseline is given by the “vanilla” CVAE with a fixed Gaussian prior following [109]. For completeness, we report the performance of our method as well as all baselines both with and without the cluster vector input \( c(I) \).

**Parameter settings and training.** For all the LSTMs, we use a one-hot encoding with vocabulary size of 11,488, which is the number of words in the training set. This input gets projected into a word embedding layer of dimension 256, and the LSTM hidden space dimension is 512. We found that the same LSTM settings worked well for all models.

For our three CVAE models, we put a trade-off parameter between encoder and decoder terms when it is necessary for training. We tuned the dimension of the \( z \) space on the validation set and use 150 for CVAE, GMM-CVAE, and AG-CVAE. We wanted the dimensionality of \( z \) space to be at least equal to the number of categories so that each \( z \) vector corresponds to a unique set of cluster weights.

The means \( \mu_k \) of clusters for GMM-CVAE and AG-CVAE are randomly initialized on the unit ball and are not changed throughout training. The standard deviations \( \sigma_k \) are set to 0.1 at training time and tuned on the validation set at test time (the values used for our results are reported in the tables).

All networks are trained with SGD with a learning rate that is 0.01 for the first 5 epochs, and is reduced by half every 5 epochs.

4.3.2 Results

A big part of the motivation for generating diverse candidate captions is the prospect of being able to re-rank them using some other method. Because the performance of any re-ranking method is upper-bounded by the quality of the best candidate caption in the set, it makes sense to evaluate different methods assuming an oracle that can choose the best sentence among all the candidate predictions. For a more realistic evaluation, we also use a consensus re-ranking approach [38] to automatically select one of the sentences for evaluation. We compare caption quality using five metrics: BLEU [28], METEOR [114], CIDEr [32], SPICE [31], and ROUGE [115]. These are calculated using the COCO caption evaluation tool [13] augmented by the author of SPICE [31]. Finally, we assess the diversity
of the generated captions using uniqueness and novelty metrics.

**Oracle evaluation.** Table (4.1) reports caption evaluation metrics in the oracle setting, i.e., taking the maximum of each relevant metric over all the candidates. For the LSTM baseline, we report the scores attained among 10 candidates generated using beam search (as suggested in [22]). The high-level trend we see from this table is that the CVAE baseline performs worse than the basic LSTM, while the upper-bound performance for GMM-CVAE and AG-CVAE considerably exceeds that of the LSTM given the right choice of sampling parameters. This upper bound can also be improved by drawing more $z$ samples (100 instead of 20), which is much more efficient than increasing beam width for the LSTM.

In more detail, the top two lines of Table (4.1) compare performance of the LSTM with and without the additional object (cluster) vector input, and show that it does not make a dramatic difference. That is, to achieve a significant improvement over the LSTM baseline, it is not sufficient to add strong side information as an input without changing the structure of the model. For CVAE, GMM-CVAE and our AG-CVAE, we sample a fixed number of $z$ vectors from the corresponding prior distributions (the numbers of samples are given in the table). Just as with the LSTM, we can see that using the object vector as additional conditioning information in the encoder and decoder can improve performance somewhat, but does not account for all of the power of our method. One thing we observed about the

<table>
<thead>
<tr>
<th></th>
<th>obj</th>
<th>#z</th>
<th>std</th>
<th>beam</th>
<th>B4</th>
<th>B3</th>
<th>B2</th>
<th>B1</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>0.413</td>
<td>0.515</td>
<td>0.643</td>
<td>0.790</td>
<td>1.157</td>
<td>0.597</td>
<td>0.285</td>
<td>0.218</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>0.428</td>
<td>0.529</td>
<td>0.654</td>
<td>0.797</td>
<td>1.202</td>
<td>0.607</td>
<td>0.290</td>
<td>0.223</td>
</tr>
<tr>
<td>CVAE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.261</td>
<td>0.381</td>
<td>0.538</td>
<td>0.742</td>
<td>0.860</td>
<td>0.531</td>
<td>0.246</td>
<td>0.184</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>-</td>
<td>0.312</td>
<td>0.421</td>
<td>0.565</td>
<td>0.733</td>
<td>0.910</td>
<td>0.541</td>
<td>0.244</td>
<td>0.176</td>
</tr>
<tr>
<td>GMM-CVAE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.371</td>
<td>0.481</td>
<td>0.619</td>
<td>0.778</td>
<td>1.080</td>
<td>0.582</td>
<td>0.274</td>
<td>0.209</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>2</td>
<td>0.449</td>
<td>0.553</td>
<td>0.666</td>
<td>0.813</td>
<td>1.216</td>
<td>0.617</td>
<td>0.298</td>
<td>0.233</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>0.494</td>
<td>0.597</td>
<td>0.719</td>
<td>0.856</td>
<td>1.378</td>
<td>0.659</td>
<td>0.325</td>
<td>0.261</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>0.527</td>
<td>0.625</td>
<td>0.740</td>
<td>0.865</td>
<td>1.430</td>
<td>0.670</td>
<td>0.329</td>
<td>0.263</td>
</tr>
<tr>
<td>AG-CVAE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.431</td>
<td>0.537</td>
<td>0.668</td>
<td>0.814</td>
<td>1.230</td>
<td>0.622</td>
<td>0.300</td>
<td>0.235</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>-</td>
<td>0.451</td>
<td>0.557</td>
<td>0.686</td>
<td>0.829</td>
<td>1.259</td>
<td>0.630</td>
<td>0.305</td>
<td>0.243</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>2</td>
<td>0.471</td>
<td>0.573</td>
<td>0.698</td>
<td>0.834</td>
<td>1.308</td>
<td>0.638</td>
<td>0.309</td>
<td>0.244</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>-</td>
<td>0.532</td>
<td>0.631</td>
<td>0.749</td>
<td>0.876</td>
<td>1.478</td>
<td>0.682</td>
<td>0.342</td>
<td>0.278</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>0.557</td>
<td>0.654</td>
<td>0.767</td>
<td>0.883</td>
<td>1.517</td>
<td>0.690</td>
<td>0.345</td>
<td>0.277</td>
</tr>
</tbody>
</table>

Table 4.1: Oracle (upper bound) performance according to each metric. Obj indicates whether the object (cluster) vector is used; #z is the number of $z$ samples; std is the test-time standard deviation; beam is the beam width if beam search is used. For the caption quality metrics, C is short for Cider, R for ROUGE, M for METEOR, S for SPICE.
models without the object vector is that they are more sensitive to the standard deviation parameter and require more careful tuning (to demonstrate this, the table includes results for several values of $\sigma$ for the CVAE models).

**Consensus re-ranking evaluation.** In real applications, we need the caption system to output a single best description for any input image. And there will not be any ground-truth captions available to calculate oracle scores. In this case, we need to find one way to rank the generated sentences and choose the best one as output. In this chapter, we will use consensus re-ranking [22, 38] to rank sentence candidates.

Specifically, for a given test image, we first find its $K$ nearest neighbors in the training set. We take the union of their ground-truth captions to create a set $C$ of $n$ candidate sentences. In COCO dataset, since each image has around five sentences, we have $n = 5K$.

The target is to choose the best generated sentence given the “reference” set $C$. We first review the consensus re-ranking formulas that are used in the work of [38]. The consensus caption in [38] is defined as the one with highest average lexical similarity scores to other captions in the set $C$. This can be formulated as follows:

$$c^* = \arg\max_{c \in C} \sum_{c' \in C} \text{Sim}(c, c')$$

The $\text{Sim}$ here is any similarity score between two captions. In this thesis, we use BLEU-4 [28] and CIDEr [32] scores. In order to smooth the scores and reduce outlier effects, the following is used in [38]:

<table>
<thead>
<tr>
<th></th>
<th>obj</th>
<th>#z</th>
<th>std</th>
<th>beam</th>
<th>B4</th>
<th>B3</th>
<th>B2</th>
<th>B1</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>0.286</td>
<td>0.388</td>
<td>0.529</td>
<td>0.702</td>
<td>0.915</td>
<td>0.510</td>
<td>0.235</td>
<td>0.165</td>
</tr>
<tr>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>0.292</td>
<td>0.395</td>
<td>0.536</td>
<td>0.711</td>
<td>0.947</td>
<td>0.516</td>
<td>0.238</td>
<td>0.170</td>
</tr>
<tr>
<td>CVNE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.245</td>
<td>0.347</td>
<td>0.495</td>
<td>0.674</td>
<td>0.775</td>
<td>0.491</td>
<td>0.217</td>
<td>0.147</td>
</tr>
<tr>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>0.265</td>
<td>0.372</td>
<td>0.521</td>
<td>0.698</td>
<td>0.834</td>
<td>0.506</td>
<td>0.225</td>
<td>0.158</td>
</tr>
<tr>
<td>GMCMNE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.271</td>
<td>0.376</td>
<td>0.522</td>
<td>0.702</td>
<td>0.890</td>
<td>0.507</td>
<td>0.231</td>
<td>0.166</td>
</tr>
<tr>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>2</td>
<td>0.289</td>
<td>0.394</td>
<td>0.538</td>
<td>0.718</td>
<td>0.932</td>
<td>0.516</td>
<td>0.238</td>
<td>0.170</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>-</td>
<td>0.292</td>
<td>0.402</td>
<td>0.552</td>
<td>0.728</td>
<td>0.972</td>
<td>0.520</td>
<td>0.241</td>
<td>0.174</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>0.307</td>
<td>0.413</td>
<td>0.557</td>
<td>0.729</td>
<td>0.986</td>
<td>0.525</td>
<td>0.242</td>
<td>0.177</td>
<td></td>
</tr>
<tr>
<td>AGNECVAE</td>
<td>✓</td>
<td>20</td>
<td>0.1</td>
<td>-</td>
<td>0.287</td>
<td>0.394</td>
<td>0.540</td>
<td>0.715</td>
<td>0.942</td>
<td>0.518</td>
<td>0.238</td>
<td>0.168</td>
</tr>
<tr>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>-</td>
<td>0.286</td>
<td>0.391</td>
<td>0.537</td>
<td>0.716</td>
<td>0.953</td>
<td>0.517</td>
<td>0.239</td>
<td>0.172</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>20</td>
<td>2</td>
<td>2</td>
<td>0.299</td>
<td>0.402</td>
<td>0.544</td>
<td>0.716</td>
<td>0.963</td>
<td>0.518</td>
<td>0.237</td>
<td>0.173</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>-</td>
<td>0.301</td>
<td>0.410</td>
<td>0.557</td>
<td>0.732</td>
<td>0.991</td>
<td>0.527</td>
<td>0.243</td>
<td>0.177</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>100</td>
<td>2</td>
<td>2</td>
<td>0.311</td>
<td>0.417</td>
<td>0.559</td>
<td>0.732</td>
<td>1.001</td>
<td>0.528</td>
<td>0.245</td>
<td>0.179</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Consensus re-ranking using CIDEr. See caption of Table 4.1 for legend.
\[ c^* = \arg \max_{c \in C} \max_{M \subseteq C} \sum_{c' \in M} \text{Sim}(c, c') \quad (4.8) \]

\( M \) is chosen as a subset of \( C \). And the inner maximization is over all subsets in \( C \) with the size of \( m \). In this chapter, we use the source code released by [22] to calculate consensus scores. The difference between our consensus re-ranking and the above Eq. (4.8) used in [38] is that, we treat each generated sentence as the hypothesis. In other words, we calculate the similarity scores between the generated sentences and ground truth captions of the nearest neighbor images, which is different from Eq. (4.8).

To find the nearest neighbor images, we first train two-branch embedding network using the MatConvNet code released from [39]. In order to train this embedding network, we extract both visual and textual features following instructions of [39], which are not the same with those used in Table (4.1). After the embedding network is trained, we map all images onto the learned shared embedding space. Given a test image, we search for its nearest neighbors in the low dimensional embedding space instead of doing that in original VGG feature space. This is inspired by the approach from mRNN [22]. Then we take the corresponding ground-truth captions as the set \( C \) and calculate consensus scores between them and the generated sentences. Since CIDEr seems to capture more human correlations [32, 33], we do the consensus re-ranking using CIDEr scores.

Table (4.2) shows the evaluation based on the single top-ranked sentence for each test image. While the re-ranked performance cannot get near the upper bounds of Table (4.1), the numbers in Table (4.2) follow a similar trend, with GMM-CAVE and AG-CVAE achieving better performance than the baselines in almost all metrics, sometimes by a significant margin. It should also be noted that, while it is not our goal to outperform the state of the art in absolute terms, our performance is actually better than some of the best methods to date [22, 89], although [89] was trained on a different split.

AG-CVAE tends to get slightly better performances than GMM-CVAE in all metrics in oracle Table (4.1). This means that our AG-CVAE is able to generate more accurate captions than GMM-CVAE.

When using CIDEr as the similarity score, AG-CVAE seems to get consistent better performances than GMM-CVAE and other VAE variants in all metrics in Table (4.2). However, we later found GMM-CVAE and AG-CVAE got closer performance when using B-4 as the similarity score for consensus re-ranking. The consensus re-ranking results convey messages that: (1) there are still big gaps between oracle performances and consensus re-ranking performances; (2) how to efficiently re-rank the generated sentences might be an interesting direction in the future.
Table 4.3: Diversity evaluation. For each method, we report the percentage of unique candidates generated per image by sampling different numbers of z vectors. We also report the percentage of novel sentences (i.e. sentences not seen in the training set) out of (at most) top 10 sentences following consensus re-ranking. It should be noted that for CVAE, there are 2,466 novel sentences out of 3,006. For GMM-CVAE and AG-CVAE, we get roughly 6,200-7,800 novel sentences.

**Diversity evaluation.** To compare the generative capabilities of our different methods we report two indicative numbers. One is the average percentage of unique captions in the set of candidates generated for each image. This number only makes sense for the CVAE models, where we sample candidates by drawing different z samples, and multiple z’s can result in the same caption. For LSTM, the candidates are obtained using beam search and are by definition distinct. From Table (4.3), we observe that CVAE has very little diversity, GMM-CVAE is better, while the corresponding numbers for AG-CVAE are dramatically higher.

Similar to [27], we also report the percentage of all generated sentences for the test set that have not been seen in the training set. It only really makes sense to assess novelty for sentences that are plausible, so we compute this percentage based on (at most) top 10 sentences per image after consensus re-ranking. Based on this novelty ratio, the CVAE baseline does well. However, since it generates fewer distinct captions per image, the absolute numbers of novel sentences are much lower than for the GMM-CVAE and AG-CVAE (see Table (4.3) caption for details).

**Qualitative results.** Finally, Fig. (4.4) compares captions generated by AG-CVAE and the LSTM baseline on four example images. The AG-CVAE captions tend to exhibit a more diverse sentence structure with a wider variety of nouns and verbs used to describe the same image. Often this yields captions that are more accurate (“open refrigerator” vs.
“refrigerator” in (a)) and better reflective of the cardinality and types of entities in the image (in (b), our captions mention both the person and the horse while the LSTM tends to mention only one). Even when our method does not manage to generate any correct candidates, as in (d), it still gets the right number of people in some candidates. A shortcoming of AG-CVAE is that detected objects frequently end up omitted from the candidate sentences if the LSTM language model cannot accommodate them (“bear” in (b) and “backpack” in (c)). On the one hand, this shows that the capacity of the LSTM decoder to generate combinatorially complex sentences is still limited, but on the other hand, this provides robustness against false positive detections.

**Controllable sentence generation.** To show how the output of GMM-CVAE and AG-CVAE models can change when we change the input object vectors, we illustrate examples in Fig. (4.5). We observe that for the same number of $z$ samples, AG-CVAE produces more unique candidates than GMM-CVAE. For the first example showing a cat, when we add the object label “chair”, AG-CVAE is able to generate some captions mentioning a chair, but GMM-CVAE is not. Similarly, in the second example, when we add the concepts of “sandwich” and “cake”, only AG-CVAE can generate some sentences that capture them. Still, the controllability of AG-CVAE leaves something to be desired, since, as observed above, it has trouble mentioning more than two or three objects in the same sentence, especially in unusual combinations.

### 4.4 CONCLUSION AND FUTURE WORK

This chapter shows that both our proposed GMM-CVAE and AG-CVAE approaches can generate image captions that are more diverse and more accurate than standard LSTM baselines. While GMM-CVAE and AG-CVAE have very similar bottom-line accuracies according to Table (4.2), AG-CVAE has a clear edge in terms of diversity (unique captions per image) and controllability.

To date, CVAEs have been used for image question generation [109], but as far as we know, our work is the first to apply them to captioning. A mixture of Gaussian prior is used in [116] for colorization. Their approach is essentially similar to our GMM-CVAE, though it is based on mixture density networks [117] and uses a different approximation scheme during training.

Our CVAE formulation has some advantages over the conditional GAN (CGAN) approach adopted by other recent works aimed at the same general goals [26, 27]. GANs do not expose control over the structure of the latent space, while our additive prior results in an interpretable way to control the sampling process. While we represent the $z$ space as a simple
vector space with multiple modes, it is possible to impose on it a more general graphical model structure [118], though this incurs a much greater level of complexity.

Finally, from the viewpoint of inference, our work is also related to general approaches to diverse structured prediction, which focus on extracting multiple modes from a single energy function [119]. This is a hard problem necessitating sophisticated approximations, and we prefer to circumvent it by cheaply generating a large number of diverse and plausible candidates, so that “good enough” ones can be identified using simple re-ranking mechanisms.

Future work

We would like to investigate more general formulations for the conditioning information $c(I)$, not necessarily relying on object labels whose supervisory information must be provided separately from the sentences. These can be obtained, for example, by automatically clustering nouns or noun phrases extracted from reference sentences, or even clustering vector representations of entire sentences. We are also interested in other tasks, such as question generation, where the cluster vectors can represent the question type (“what is,” “where is,” “how many,” etc.) as well as the image content. Control of the output by modifying the $c$ vector would in this case be particularly natural.

In this chapter, we evaluate the diversity mainly using the percentage of unique sentences and the number of novel sentences that are not in the training set. In the future, it is also interesting to discover other evaluation metrics which can measure the diversity from a more comprehensive view. For example, we can record the locations of some meaningful words (e.g. head-nouns) and plot the histograms of their positions. We can also try to measure the distinctiveness of n-grams. With the more comprehensive understanding of “diversity”, we believe it will encourage more advanced approaches to appear in the future to this field.
Figure 4.4: Comparison of captions produced by our AG-CVAE method and the LSTM baseline. For each method, top five captions following consensus re-ranking are shown.
Figure 4.5: Comparison of captions produced by GMM-CVAE and AG-CVAE. The captions are generated via the control of input object vectors. For both models, we get 20 samples and list all unique captions. The number of unique captions tells the generative strength of a deep generative model.
CHAPTER 5: TRAINING WITH REINFORCEMENT LEARNING AND RE-RANKING

In Chapter 4, we introduced our method for diverse image captioning using the conditional variational auto-encoder (CVAE) framework. The whole network is trained according to the objective function of Eq. (4.7), which essentially has two terms, the maximum likelihood term given the samples of $z$, and the KL divergence term for fitting the prior of $z$.

The generated captions are evaluated using multiple syntactic metrics including BLEU [28], METEOR [29], ROUGE [30], SPICE [31] and also CIDEr [32]. Optimizing these syntactic metrics could be the most direct way to improve the system performance. However, it is not easy since none of these metrics are differentiable. Recently, there have been a number of works [33, 34, 120] using policy gradient to maximize the expected rewards which consist of above evaluation scores. In this scenario, the image captioning task is treated as a reinforcement learning problem, where the recurrent network (e.g. LSTM) decoder needs to choose an action in each time step and this action space is as large as the codebook dictionary size which can reach 10K in most settings.

As described in Chapter 4, we expect our approach to generate captions that are both accurate and diverse. Unlike previous approaches relying on conventional LSTM decoder that only take image feature as the input, the decoder from our CVAE framework needs to take additional $z$ vectors, as shown in Fig. (5.1). Before diving deeper in this chapter, we are still unsure about how these additional $z$ vector inputs can affect the performance. And especially, we want to ask several questions: (1) Can we further improve the captioning accuracy of our CVAE framework using reinforcement learning? (2) What should the reinforcement signal look like? (3) Will policy gradient training destroy the diversity in generating sentences?

We try to answer the above questions in this chapter. In Section 5.1, we formulate an objective function that contains two terms. The first term is the maximum likelihood term (MLE loss), and the second is a surrogate reinforcement term. We experimentally find that using the combination of two losses can give improvements in caption accuracy over the pre-trained decoder from our CVAE framework in Chapter 4.

Given the improved caption decoder, we still face the core problem of outputting a single best sentence. In Chapter 4, we used consensus re-ranking to rank the generated sentences set by searching nearest neighbors of testing images in the training set. However, consensus re-ranking is unsatisfactory for two reasons: (1) It is very time and memory intensive. For example, in the testing case of Chapter 4, it can take up to hours to calculate all consensus scores. (2) It is necessary to keep the training set around at test time and thus heuristic.
Therefore, we need advanced ranking methods that can be much faster and less heuristic. In Section 5.2, we explore several deep rankers for this sentence ranking task. We try different triplet sampling schemes for our rankers and find that triplet sampling can affect re-ranking performances. Our best ranker can finish this sentence re-ranking task more quickly. But frustratingly, it still cannot outperform the heuristic consensus re-ranking approach. At the end of this chapter, we will give some discussions about current limitations and show future possible ways of dealing with this sentence re-ranking task.

Figure 5.1: Training CVAE decoder with the mixed reinforcement learning. This decoder is cropped from the pre-trained AG-CVAE framework in Fig.(4.3). The maximum likelihood estimator (ML) is a cross-entropy loss that imposes word level accuracy. The approximate reinforcement term is plotted in the dashed rectangular box. For each time step, there are three Monte Carlo rollouts that are used to estimate the expected rewards. The red box represents the end token of each sentence. While training, we feed ground-truth caption words into the main sequence path (boxes that are with LSTM cells). When doing inference, we decode the whole sequence word by word in the greedy way.

5.1 MIXED REINFORCEMENT TRAINING

In this section, we will introduce our approach of training caption decoder with a combination loss that contains two terms, one of which is the maximum likelihood term and the
other one is an approximate reinforcement term.

Fig. (5.1) shows the basic structure of our caption generator. It is the decoder part inherited from our CVAE framework in Chapter 4 (see the right sub-figure of Fig. (4.3)). We first train the AG-CVAE model to convergence by optimizing Eq. (4.7) in Chapter 4. Then, we fix the encoder of the AG-CVAE model and only update the decoder with the reinforcement training strategy that we are going to present in this chapter.

In the following, we will describe the two terms of the loss function we use in training the decoder.

**Maximum Likelihood (ML)**

Following the similar notations as in Chapter 4, we write the likelihood loss below. When using ML for training, it means that the model parameters \( \theta \) (referring to parameters of decoder) are optimized by minimizing the following,

\[
L^\theta_{\text{ML}} = -\log p_\theta(x^*|z, c) \\
= -\sum_{t=1}^{T} \log p_\theta(x^*_t|x_{1:t-1}^*, z, c),
\]  

(5.1)

where \( c \) is the current cluster vector and we omit the notation of image features. When \( t = 1 \), \( x^*_1 \) is the first word and there is no partial sequence of \( x^*_{1:t-1} \). \( x^*_{1:t-1} \) only makes sense when \( t > 1 \). For simplicity, we keep this notation of \( t \) across this chapter. \( z \) is sampled from the prior distribution of \( p(z|c) \). This is different from what the decoder does in Chapter 4 where \( z \) is sampled from the \( q(z) \) since we need to jointly train both encoder and decoder of our CVAE model. \( p(z|c) \) is the prior Gaussian distribution where the center is defined by the linear combination of cluster centers according to \( c \). While training, we keep the standard deviation of this \( p(z|c) \) Gaussian distribution to be very small (e.g. 0.001 in our experiments), which means \( z \) is almost sampled from the center of this Gaussian distribution. We only sample a single \( z \) for each data. \( x^* = (x^*_1, x^*_2, ..., x^*_T) \) represents one ground-truth caption of the current image. For simplicity, we drop notations of data indexes in this chapter. \( T \) represents the maximum time step of the sentence.

We use the same \( z \) sampling procedure for both MLE loss and the reinforcement learning term. From the above ML loss, we can easily see that it is trying to minimize word level prediction errors.

**Reinforcement Term**

We will first explain how we understand the image captioning problem as a reinforcement learning task and then go through formulas.
In Fig. (5.1), given the time step $t$, we need to choose a discrete action from the action space. Here, each action corresponds to one word choice in the dictionary. The action is chosen according to a stochastic policy, which is denoted as $\pi_\theta(x_t|x_{1:t-1}, z, c)$, where $x_{1:t-1}$ is the partial sequence of words chosen before time step $t$. This policy $\pi_\theta(x_t|x_{1:t-1}, z, c)$ is actually the softmax function over the vocabulary. Because $z$ and $c$ are only conditional vectors which depend on images, for simplicity, we omit notations of $z$ and $c$ in the following derivations.

The reward can only be calculated when the generated sequence reaches the maximum time step (which is denoted as $T$ in this section) or come across the end token before reaching the $T$-th step. Following recent works of [33, 120], we use BLEU-4 (4 grams) to calculate the reward of a complete sentence.

We define the reward of a complete sequence $x_{1:T}$ as follows,

$$R(x_{1:T}) = \text{BLEU}(x_{1:T}; x^*), \quad (5.2)$$

where $x^*$ is the set of ground-truth captions for the current image and the $\text{BLEU}$ calculates score of hypothesis $x_{1:T}$ according to the reference set $x^*$. Specifically, when we are using COCO dataset [13], $x^*$ represents five ground-truth captions for the image. These ground-truth captions are used as reference sentences when calculating BLEU scores. Here, we should mention that our approach is not only restricted to BLEU scores. Recent work [33] also shows that combinations of BLEU, CIDEr and other metrics can better match human judgment.

When using reinforcement learning approaches to train the decoder (with parameter $\theta$), the goal is to maximize the expected reward (or minimize the negative of it) of the whole generated sentence, which is defined as follows:

$$L^\theta_{RL} = -E_{x \sim \pi_\theta}[R(x_{1:T})] \quad (5.3)$$

The difficulty here is, using BLEU scores or other evaluation metrics as rewards makes this loss function non-differentiable. In order to compute the gradient of $L^\theta_{RL}$, we use the policy gradient method following recent image captioning work of [33]. We follow the derivations of getting gradients in [33] and omit the details here.

We have the approximated gradient using Monte-Carlo sampling as:
\[ \nabla_{\theta} L^{\theta}_{RL} \approx -\frac{1}{M} \sum_{m=1}^{M} \sum_{t=1}^{T} \nabla_{\theta} \log \pi_{\theta}(x_t|x_{1:t-1}^m) \ast Q(x_{1:t-1}, x_t) \] (5.4)

where, \( Q(x_{1:t-1}, x_t) \) is defined as the expected future rewards of current partial sequence at time step \( t \), which is:

\[ Q(x_{1:t-1}, x_t) = E_{x_{t+1:T}}[R(x_{1:t-1}; x_t; x_{t+1:T})] \]

This is to follow [33], where we introduce \( Q \) function since the reward only makes sense when it is calculated at the end of the generated sentence. There is no easy way to calculate any intermediate rewards for a partial sequence.

Following [33, 121], we use \( K \) rollouts to approximate the function of \( Q(x_{1:t-1}, x_t) \). As shown in Fig.(5.1), at each time step \( t \), we sample \( K \) = 3 paths to append the partial sequence and compute the averaged path rewards as:

\[ Q(x_{1:t-1}, x_t) \approx \frac{1}{K} \sum_{k=1}^{K} R(x_{1:t-1}; x_t; x_{t+1:T}^k) \] (5.5)

In order to reduce the variance, we subtract the baseline term \( B_{\gamma}(x_{1:t-1}) \) from the \( Q \) function. And thus, we have:

\[ \nabla_{\theta} L^{\theta}_{RL} \approx -\frac{1}{M} \sum_{m=1}^{M} \sum_{t=1}^{T} \nabla_{\theta} \log \pi_{\theta}(x_t|x_{1:t-1}^m) \ast [Q(x_{1:t-1}, x_t) - B_{\gamma}(x_{1:t-1}^m)] \] (5.6)

In the above formula, \( B_{\gamma}(x_{1:t-1}) \) is the baseline function with model parameter \( \gamma \). We use a two-layer fully connected network (each layer is followed by a ReLU function) to minimize the loss, \( L_{\gamma} = \sum_{t} E_{x_{1:t-1}} E_{x_t} (Q(x_{1:t-1}, x_t) - B_{\gamma}(x_{1:t-1}))^2 \), which is following the baseline loss in [33]. In experiments, we pre-train \( B_{\gamma} \) for one epoch, then load the baseline and optimize it together with the decoder. \( M \) is set to 1 in the work of [33]. In Eq.(5.6), ideally, the Monte-Carlo paths \( x_{1:t-1}^m \) should be generated from the policy \( \pi_{\theta} \). But in our experiments, we approximate this by using the ground-truth partial sequence as \( x_{1:t-1} \), then sample \( x_t \) and its following \( K \) rollouts of \( \{x_{t+1:T}^k\}_K \) according to policy \( \pi_{\theta} \). To avoid clutter, we still use \( x_{1:t-1} \) in the following formula instead of introducing new notations to represent the ground truth partial sequence. Our implementation is based on Tensorflow, which has automatic differentiations. Therefore, we have the surrogate approximate reinforcement term as:
Mixed Reinforcement Training

As we have introduced both the maximum likelihood term and the approximate reinforcement term, we will now explain the mixed reinforcement training we use in our following experiments.

There are two reasons why we combine two loss terms together for training. First, the maximum likelihood loss encourages accurate “word level” predictions, while the approximate reinforcement term can take care of the sentence level structure by calculating the sentence reward signals (e.g. BLEU scores and other syntactic metrics). Therefore, the combination of two terms can encourage the decoder to generate descriptions that have both word level accuracy and meaningful sentence structures.

Second, due to a large action space (word dictionary size), we find in our experiment, the policy function needs a good guide. As we have explained earlier in this section, for each time step $t$, we sample the action from the policy $\pi_\theta$. While training, the gradients will be affected by both the ML term and the approximate reinforcement term. The idea of using this mixed training strategy is also explored in the recent work of [122], though the implementations can be different.

Therefore, by omitting data indexes, we define the mixed loss function as follows:

$$L^\theta_{RL} \approx -\sum_{t=1}^{T} \log \pi_\theta(x_t|x_{1:t-1}) \ast (Q(x_{1:t-1}, x_t) - B_\gamma(x_{1:t-1})).$$  \hspace{1cm} (5.7)

The first term is the likelihood term referring to Eq. (5.1) and the second term is the approximate reinforcement learning term of Eq. (5.7). $\lambda$ is the trade-off parameter. We set $\lambda$ to be 0.9 in experiments. Both terms have the same set of parameters $\theta$. As shown in Fig. (5.1), we only train the decoder part of AG-CVAE, which takes inputs of image features, cluster vectors $c$, $z$ samples and ground-truth caption words while training. $z$ is sampled from prior Gaussian distribution, with the mean calculated according to cluster vector $c$. In our experiments, we control $z$ to be sampled almost from the center of this distribution (sampling std = 0.001).
5.1.1 Experiments

In this section, we first give comparison baselines and then show that mixed reinforcement training will improve the decoder in our CVAE framework. We show performance improvements of generated captions in both oracle Table (5.1) and consensus re-ranking Table (5.2).

Baselines

- **AG-CVAE**: we train the AG-CVAE model by optimizing objective function of Eq. (4.7) in Chapter 4.

- **MLE FT**: by loading the pre-trained AG-CVAE model above, we keep the encoder fixed and fine-tune the decoder (with parameters $\theta$) using the maximum likelihood loss. This is implemented by choosing $\lambda = 0$ in Eq. (5.8).

- **Mix FT**: by loading the pre-trained AG-CVAE model above, we keep the encoder fixed and fine-tune the decoder (with parameters $\theta$) using the mixed reinforcement training. This is implemented by choosing $\lambda = 0.9$ in Eq. (5.8).

We follow the experiment settings in Chapter 4, using the same COCO train/val/test splits. For network structures and parameters initializations, we also keep using the same. But for the whole image features, we change to extract from ResNet-152 [123] with ImageNet pre-trained weights instead of VGG net [103] that are used in Chapter 4.

We show the training and testing procedures of “Mix FT” in detail as follows:

- **Training Procedure of “Mix FT”**

  1. Training the AG-CVAE framework as presented in Chapter 4.
  2. Load the pre-trained decoder as shown in Fig. (5.1) as the “generator”.
  3. Start training the decoder with the combined loss function of Eq. (5.8). For each image-sentence pair, we sample one $z$ vector from the prior distribution $p(z|c)$. The standard deviation of sampling $z$ is controlled to be very small (std = 0.001). While training, we use the ground-truth word for each time step in the main path as shown in Fig. (5.1), and sample three rollouts ($K = 3$ in Eq. (5.5)) according to the stochastic policy. We first pre-train the baseline model $B_\gamma$ in Eq. (5.7) with the learning rate of 0.001 for about one epoch. Then, we use the start learning rate of 0.0005 to train the mixed object function of Eq. (5.8) and update $B_\gamma$ together. We use Adam [106] optimizer to optimize this mixed surrogate loss. The batch size is set to be 256, and we observe the convergence after 10 epochs.
### Table 5.1: Oracle (upper bound) performance (calculated according to BLEU-4 scores).

<table>
<thead>
<tr>
<th></th>
<th>#z</th>
<th>std</th>
<th>B4</th>
<th>B3</th>
<th>B2</th>
<th>B1</th>
<th>C</th>
<th>R</th>
<th>M</th>
<th>Div</th>
</tr>
</thead>
<tbody>
<tr>
<td>AG-CVAE</td>
<td>20</td>
<td>2</td>
<td>0.463</td>
<td>0.566</td>
<td>0.681</td>
<td>0.803</td>
<td>1.197</td>
<td>0.600</td>
<td>0.285</td>
<td>0.727</td>
</tr>
<tr>
<td>MLE FT</td>
<td>20</td>
<td>2</td>
<td>0.468</td>
<td>0.569</td>
<td>0.681</td>
<td>0.802</td>
<td>1.223</td>
<td>0.605</td>
<td>0.286</td>
<td>0.823</td>
</tr>
<tr>
<td>Mix FT</td>
<td>20</td>
<td>2</td>
<td>0.489</td>
<td>0.587</td>
<td>0.695</td>
<td>0.809</td>
<td>1.226</td>
<td>0.606</td>
<td>0.287</td>
<td>0.770</td>
</tr>
</tbody>
</table>

Table 5.1: Oracle (upper bound) performance (calculated according to BLEU-4 scores). 

- **Inference Procedure**

  For inference, we use the same decoding strategy as in Chapter 4. As shown in Fig. (5.1), given the image feature input and cluster vector input \( c \), together with the sampled \( z \) vector, we generate the sentence word by word in a greedy way.

- **Results Analysis**

  We list oracle performance in Table (5.1). This table shows that our Mix FT is better than original pre-trained AG-CVAE. Especially, the Mix FT improves scores of BLEU-4 from 0.463 to 0.489.

  MLE FT is listed here for ablation study. Our Mix FT method gives further improvements by incorporating the mixed reinforcement training.

  Another observation from Table (5.1) is that, even with the mixed reinforcement training, the diversity (percentage of unique generated sentences per image) doesn’t decrease. This means that our new approach can improve the accuracy and keep the diversity at the same time. In Fig. (5.2), we show several comparison examples of generated sentences from AG-CVAE model and Mix FT model. Like (a) in this figure, the Mix FT model is able to generate more accurate descriptions of “bench” than AG-CVAE model. In (b), the Mix FT model even generates one sentence with “a large white and blue plane”, which describes the plane in the image accurately. (c) shows examples that Mix FT can make correct subjects “a man and a woman” instead of only describing one person as AG-CVAE. Likewise, in
(d), Mix FT generates the first sentence “a computer and two monitors” which precisely describes the content of the picture.

Table (5.2) gives results of consensus re-ranking. We calculate consensus scores using BLEU-4 across the table. And we use the raw ResNet-152 features to find images nearest neighbors. From the table, we observe that our Mix FT can still give improvements on BLEU and CIDEr scores over the AG-CVAE method after consensus re-ranking.

5.2 TWO-BRANCH RANKER FOR RE-RANKING SENTENCES

Section 5.1 shows that the mixed reinforcement training can help to improve both the oracle performance and consensus re-ranking performance. However, in order to get the most accurate sentence from the generated candidate set, we still need the heuristic consensus re-ranking step to rank sentences by calculating similarity scores to a reference set of training sentences, as described in Chapter 4.

In this section, we study different variants of two-branch networks to rank the generated sentences. As shown in Fig. (5.3), we make use of this two-branch network structure from Chapter 3. The left branch takes the input of image features and the right one takes the input of language representations. Here, we use recurrent networks (i.e. LSTM) for language modeling. On both the image and language sides, we have two layers right before their merging into the embedding space and both branches have the L2 normalization.

5.2.1 Deep Rankers

We will discuss the loss function of our deep rankers. The main idea is the same as in Chapter 3. But the task is different. In Chapter 3, given the image, the task is to rank sentences in the testing pool, all of which are written by humans. For each test image, only five sentences in the test pool are ground-truth captions. By contrast, in this chapter, for each image, the task is to rank a set of very similar sentences that are generated by the same decoder (our AG-CVAE or Mix FT model). The ranker needs to find the top ranked sentence that would be similar to the one chosen by oracle evaluations (using BLEU or CIDEr).

The most important issue is how to select positive sentences and negative ones given the input image during training. We will discuss several variants of triplet sampling and show that how we sample the positive and negative pairs actually affects performance a lot.

Let us keep using the same notations with those in Chapter 3. For the \( i \)-th image \( x_i \), \( y_j \) is sampled from the positive set of sentences \( Y_i^{+} \) and \( y_k \) is sampled from the negative set of
sentences $Y_i^-$. We have the following distance constraint:

$$d(x_i, y_j) + m < d(x_i, y_k)$$
$$\forall y_j \in Y_i^+, \forall y_k \in Y_i^-.$$  \hspace{1cm} (5.9)

Note that here and in the following, $d(x, y)$ will denote the Euclidean distance between image and text features in the embedding space.

These ranking constraints can be converted into a margin-based loss function:

$$L(X, Y) = \sum_{i,j,k} \left[ m + d(x_i, y_j) - d(x_i, y_k) \right]_+$$  \hspace{1cm} (5.10)

where $[t]_+ = max(0, t)$. This ranking loss sums over all triplets (a target image, a positive sentence match, and a negative sentence match) defined in Eq. (5.9).

Then, the question is how to construct the positive set $Y^+$ and negative set $Y^-$? We investigate several sampling schemes here.

**Ranker Variants**

- **Ranker baseline**: for $i$-th image $x_i$, $Y_i^+$ comes from the ground-truth captions that are associated with this image. $Y_i^-$ comes from the irrelevant sentences (ground truth captions that come from other images).

- **aug-Pos-gen-Neg**: for $i$-th image $x_i$, $Y_i^+$ is constructed from two parts. One part is the ground-truth caption set from the dataset, the other part is from top generated sentences from the “MIX FT” trained decoder of Table (5.1). For example, for the $i$-th image in the training set, we use our trained decoder to generate 20 sentences. Among these, we calculate the BLEU scores of each sentence according to the reference set and choose the top K (K=4) best scoring sentences to augment the original positive set. The negative set is constructed by using those generated sentences which have M (M=8) smallest BLEU-4 scores.

- **aug-Pos-full-Neg**: The positive set is the same as in above “aug-Pos-gen-Neg”. The difference lies in the negative set $Y_i^-$, which is constructed from two parts. One part is randomly sampled irrelevant ground-truth captions of other images in the mini-batch and the other part is from the generated sentences with M (M=4) smallest BLEU-4 scores.
<table>
<thead>
<tr>
<th></th>
<th>z</th>
<th>std</th>
<th>B4</th>
<th>B3</th>
<th>B2</th>
<th>B1</th>
<th>C</th>
<th>R</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consensus Re-ranking</td>
<td>20</td>
<td>2</td>
<td>0.307</td>
<td>0.414</td>
<td>0.556</td>
<td>0.725</td>
<td>0.942</td>
<td>0.524</td>
<td>0.237</td>
</tr>
<tr>
<td>Ranker baseline</td>
<td>20</td>
<td>2</td>
<td>0.269</td>
<td>0.376</td>
<td>0.525</td>
<td>0.705</td>
<td>0.879</td>
<td>0.505</td>
<td>0.233</td>
</tr>
<tr>
<td>aug-Pos-gen-Neg</td>
<td>20</td>
<td>2</td>
<td>0.284</td>
<td>0.391</td>
<td>0.533</td>
<td>0.703</td>
<td>0.870</td>
<td>0.511</td>
<td>0.226</td>
</tr>
<tr>
<td>aug-Pos-full-Neg</td>
<td>20</td>
<td>2</td>
<td>0.292</td>
<td>0.400</td>
<td>0.544</td>
<td>0.716</td>
<td>0.930</td>
<td>0.515</td>
<td>0.235</td>
</tr>
</tbody>
</table>

Table 5.3: Re-ranking Results.

5.2.2 Experiments

In this section, we do experiments on re-ranking generated sentences with variants of deep rankers. As shown in Fig. (5.3), on the image side, we use the same visual features from Table (5.1); on the sentence side, we employ LSTM networks for language modeling. We train this two-branch network using Adam optimizer with an initial learning rate as 0.0002. We use the best performed “Mix FT” model from Table (5.1) to generate augmented sentences for training. For testing, we follow the procedure of Table (5.2), which is to generate 20 sentences for each test image using the pre-trained mixed RL model.

We show the results of our rankers in Table (5.3). We can see that our ranker “aug-Pos-full-Neg” so far gives the best performance among all other deep ranker variants. It outperforms “Ranker baseline” and “aug-Pos-gen-Neg” since both generated sentences and irrelevant sentences have been fed into the training stage. But frustratingly, even this best ranker cannot outperform consensus re-ranking.

If we consider the time cost of each re-ranking method, consensus re-ranking needs more than 10 minutes to run (including the cost for finding nearest neighbors and calculating consensus scores). Our best deep ranker takes merely a few seconds to calculate all scores. As mentioned earlier, our deep ranker variants don’t need to search in the training set and are less heuristic.

5.3 CONCLUSION AND FUTURE WORK

In this chapter, we first investigate the problem of training a caption decoder with the mixed reinforcement training. The improved decoder can generate more accurate captions without losing diversity. However, how to train the decoder and encoder in a joint framework under the reinforcement learning scheme still remains a question. In the future, we would like to explore more about training generative models with deep reinforcement learning.

We also study the challenging problem of re-ranking generated sentences in this chapter. If we compare the oracle performance of the mixed reinforcement learning model in Table (5.1) with the re-ranking performance in Table (5.2), we find that the gap between them
is still big. This means that, though our best trained model is able to generate some good quality sentences (i.e. with high BLEU or CIDEr scores), our rankers don’t know how to choose the best. We have tried two-branch rankers with various triplet sampling settings, but our best ranker still cannot outperform heuristic consensus re-ranking.

The success of consensus re-ranking motivates our future directions: (1) Consensus re-ranking constructs a “reference set” from the training dataset, which can help to reduce semantic ambiguity. (2) Consensus re-ranking calculates scores using evaluation metrics (e.g. BLEU), while our rankers only use embedding distances between images and sentences that are mapped onto the shared space. When we train two-branch rankers, there is no guarantee those ranking scores can preserve the order of generated sentences as in BLEU score space. (3) Another possible way of improving our rankers is to use adversarial training. We can use the ranker as a discriminator. During training, the caption generator tries to generate a sentence that is closer to the input image than ground-truth sentences, while the discriminator tries to do the opposite.
Figure 5.2: Examples of generated sentences from AG-CVAE model and the mixed reinforcement learning model (corresponding to “Mix FT” in Table (5.1)). For each method, we generate 20 sentences and rank them by oracle BLEU scores. We list the top five sentences of each image.
Figure 5.3: Two-branch Ranker for Ranking Sentences. The left branch takes input of image features, and the right branch uses LSTM to represent sentences.
CHAPTER 6: CONCLUSIONS

Joint vision-language tasks are vital to lots of real-world AI applications. Though only three tasks (image-sentence retrieval, phrase localization, and image captioning) have been explored, methods proposed by this dissertation have the potential to be used in many other tasks as well. We study the learning of joint image-language representations from both discriminative and generative views. In Chapter 3, we first propose the discriminative two-branch networks for cross-modality matching. To adapt it to a different task only requires designing the triplet sampling accordingly (e.g. choosing positive and negative pairs). In order to model the latent space with semantic priors, we introduce a deep generative model for image captioning in Chapter 4. We study the captioning problem from a new angle: we want to generate image descriptions that are both diverse and accurate. Our proposed conditional variational auto-encoder (CVAE) framework makes it possible to sample a diverse set of sentences from the semantic encoding space. In Chapter 5, we train our caption generator with a mixed reinforcement learning algorithm. The improved decoder can generate more accurate sentences without losing diversity. We also explore a challenging task of ranking generated sentences in Chapter 5. We try several deep ranking variants with different triplet sampling schemes. However, it turns out even our best ranker cannot outperform consensus re-ranking, which is a non-trained heuristic. At the end of Chapter 5, we discuss about future directions that can further improve both reinforcement training and sentence re-ranking tasks.
REFERENCES


[51] “Coco dataset website, howpublished = http://cocodataset.org/,”


and recurrent networks for sequence modeling,” arXiv preprint arXiv:1803.01271,
2018.

attention,” in CVPR, 2016.

“Bottom-up and top-down attention for image captioning and visual question answer-
ing,” CVPR.

[91] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, “Boosting image captioning with at-

visual question answering based on attributes and external knowledge,” IEEE Trans-
actions on Pattern Analysis and Machine Intelligence, 2017.


[94] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and
D. Batra, “Diverse beam search: Decoding diverse solutions from neural sequence

[95] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair,
A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in neural

[96] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with gumbel-softmax,”

[97] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed repre-
sentations of words and phrases and their compositionality,” in NIPS, 2013.

reducing internal covariate shift,” ICML, 2015.

[99] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” in Pro-


ECCV, 2014.


