Redefine the Diversity of Image Captioning

Chuan Cen
chuancen@umich.edu

Yuan Liang
yualiang@umich.edu

Xuanyu Wang
olivwang@umich.edu

Fei Yi
feiyi@umich.edu

Ruoyao Wang
ruoyaow@umich.edu

Abstract

An image can be described by various sentences in different styles. While most previous works assumed independence between style and content, we argue that the captions can also be diverse w.r.t the content. Specifically, we decompose the diversity into three dimensions: the form, the level of detail, and the content itself. Then we propose an image-captioning model that can generate various captions that varies in these three aspects, with the latter two controllable.

1 Introduction

Automatically describing the content of an image is a fundamental problem in artificial intelligence that connects computer vision and natural language processing. This field has grown fast since the Deep Neural Network (DNN) was introduced. The first model that was based on Deep CNN and RNN was proposed in (Vinyals et al., 2014), where they took the deep features of CNN as the initial hidden state of a LSTM which then generates a caption sequentially.

Most works afterwards that followed this structure used paired image-text dataset and trained the RNN model using word-wise Cross-entropy loss. But the mapping from image to text is essentially "one-to-many", that is, an image can usually be described in various ways. (Dai et al., 2017) addressed this problem by aligning image content and texts in the semantic embedding space instead of hard supervision on words, and designing a LSTM-based conditional GAN so that a noise vector can be applied to generate sentences of diverse "styles". In this way, they assume the style independent from the content. In other words, the generated sentences always express the same meaning in different ways.

Figure 1: This sample image can be described as "a man dressing a white T-shirt is playing frisbee". In a different sentence structure we can say: "a man plays frisbee wearing a white T-shirt". In a different level of detail it can be: "a man plays frisbee". With the same level of detail but different content: "a man wearing a cropped pants stands on the grass."

We argue that there are other two dimensions lying in the diversity of the captions, that is, the amount of the information, and the information itself. Take Figure 1 for example, the caption can vary in three ways as demonstrated in the figure caption. We list the three dimensions below:

1. **Form** is the element of style that is independent from content. It's the variety of the sentence structure and the word choices under the constraint of expressing the same meaning.

2. **Level of detail** is measuring the amount of information, or the richness of content of the caption. It's usually reflected by the length of the generated sentences.

3. **Content** is just "what" to describe in our captions. We can describe an image differently by emphasizing different objects, regions or illustrating it from different perspectives. The interesting objects, regions, or perspectives subject to change for different people, situations or queries.

In this project, we propose a model that can generate various captions varying in these three aspects. To our best knowledge, there has no existing work achieving such thing. (Senina et al., 2014) generated descriptions in 3 levels of details for videos, but they need ground truth labels for each level, and didn't cover other two dimensions. Many other previous works only tried to give captions in as much details as possible such as (Krause et al., 2016)(Yu et al., 2016)(Zhang et al., 2019)(Johnson et al., 2016).

The high-level idea of our method is simple. Assume we have a pre-trained CNN model able to propose a set of region boxes, we can just randomly sample from those boxes by their importances, and generate captions strictly describing those sampled boxes. The number of sampled boxes determines the level of detail of caption, and the combination of boxes determine the information we describe. We add noise to LSTM decoder for "form" diversity as did in (Dai et al., 2017). The key challenge here is to generate sentences that precisely describe the boxes given. In other words, every word can find a source from the given boxes, and in turn every box's content needs to be covered by the generated caption.

2 Related Works

Image Captioning The majority of image captioning models proposed in recent years adopted Deep CNN and RNN architecture, starting from (Vinyals et al., 2014). Many adapted this architecture by applying attention mechanism(Karpathy and Fei-Fei, 2015)(Xu et al., 2018)(You et al., 2016)(Anderson et al., 2018). Some tried to generate captions in as much details as possible (Krause et al., 2016)(Yu et al., 2016)(Zhang et al., 2019). (Johnson et al., 2016) generate a short caption for each region box detected resulting in dense captions. (Dai et al., 2017) deal with diversity of captions and was able to generate diverse captions by using LSTM-based conditional GAN and supervision on semantic level. Our work further explores the definition of "diversity" and propose a method that generates captions various in three different aspects.

Multimodal Similarity To generate diverse captions, hard supervision on words is not preferred since we don't have enough diverse captions for each image to train. Instead we seek to align text and image on semantic level, which allows the output to diverse yet still match with the image content. Doing this involves dealing with multimodal similarity between text and image. (Frome et al.,) simply learns a linear mapping between pre-trained CNN features and word-embeddings to build alignment between images and class names. (Xu et al., 2018)(Karpathy and Fei-Fei, 2015)(You et al., 2016)(Johnson et al., 2016)(Krause et al., 2016) designed various attention mechanisms to learn to measure similarity between an image and a sentence. In our work we proposed a new attention learning algorithm that build tight connection between image and text in word-level and allow the generated captions to precisely describe the interesting regions given, which is a pre-requisite for the diversity we want to achieve.

3 Proposed Methods

3.1 Problem Formulation

For the "form" part of the style, we solve it in the same way (Dai et al., 2017) did it, that is to feed a noise vector to the LSTM decoder both when training and testing. So the remaining problem is to achieve the diversity in "level of detail" and "content". As explained earlier, the key challenge for the two diversities is to make the generated caption precisely describe what's in the region boxes we provide, with no absence and no redundancy. After this, we can randomly sample the proposed region boxes by their importances which will finally result in a diversity of "level of detail" and "content".

Formally, suppose for an image we extracted out K region boxes $B \in \mathbb{R}^{K\times C\times H_r\times W_r}$ where each is represented as a feature tensor b_k of the same size $C \times H_r \times W_r$, and has a score $q_k \in \mathbb{R}$. In some way we sample out M boxes, conditioned on which we want to generate a caption e precisely describing the M boxes B_M. Suppose we have a semantic similarity measure $SimiMetric$ between f and e, then our objective is:

$$\max\{\sum_{b_k\in B_M} SimiMetric(b_k, e) - \sum_{b_k\in B\setminus B_M} SimiMetric(b_k, e)\} \quad (1)$$

3.2 Method Overview

Figure 1 gives our model architecture. This model consists of four main modules: 1) a CNN with a Fully Convolutional Localization Network (FCLN)

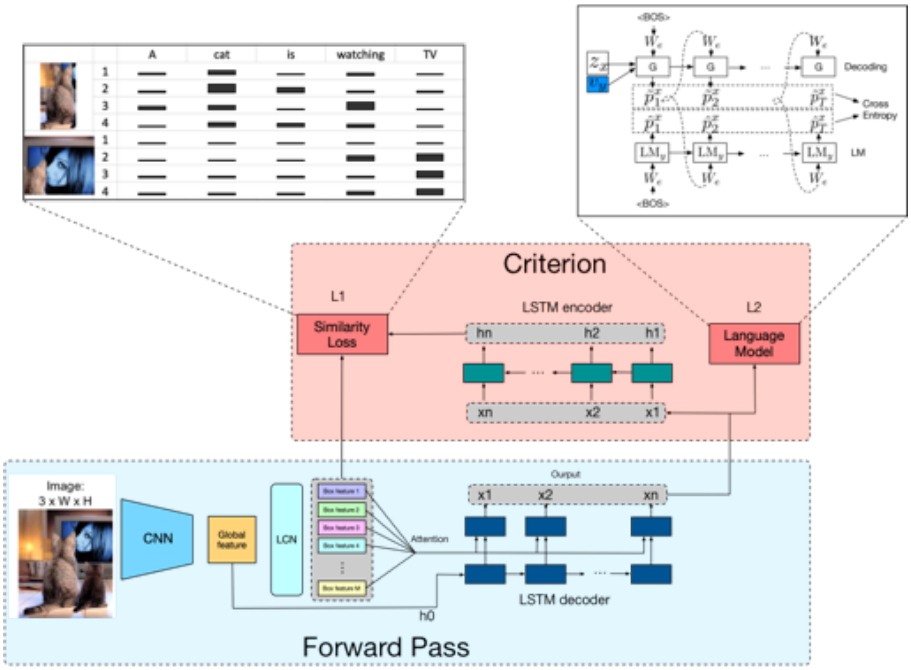


Figure 2: Model architecture.

which serves as an image feature extractor and region proposer. 2) a LSTM decoder which take the selected region box features as well as a global feature tensor as input, and generate a caption. 3) a LSTM encoder which will be trained to be a multimodal similarity evaluator and a criterion for the training of the LSTM decoder. 4) a language model (LM) which is another criterion for training LSTM decoder in order to make the sentence more fluent.

In training time, there are two steps to train the whole model. The first step is to train the LSTM encoder and the LM, which are two sources of the loss when training LSTM decoder. The second step is to train the LSTM decoder: we sample some boxes, feed them into the decoder, the output of which is fed into pretrained LM and LSTM encoder model to calculate losses. The loss from LSTM encoder measures the distance between sentence embedding and the sampled boxes.

In test time, the trained LSTM decoder just take the sampled boxes as input and generate a sentence precisely describing them.

3.3 Modules

CNN + FCLN The image feature extractor is borrowed from (Johnson et al., 2016). This model will take an image as input and generate K region boxes $B \in \mathbb{R}^{K\times C\times H_r\times W_r}$, each of which is represented as a feature tensor b_k of the same size $C \times H_r \times W_r$, $k \in \{1, ..., K\}$, along with a score $q_k \in \mathbb{R}$. We also got a global feature tensor $g \in \mathbb{R}^{C\times H'\times W'}$. Since later when training we'd treat g parallel with b_k, we downsample it to become $b_{-1} \in \mathbb{R}^{C\times H_r\times W_r}$.

In order to measure the similarity between image feature and text, we need further map B and b_{-1} to the semantic embedding space $\mathbb{R}^D$, where D is the dimension of the hidden state of the LSTM we use. To make a fine-grained similarity measure, we are not simply map a box feature b_k into a D dimension vector, instead we treat every b_k as $H_r \times W_r$ vectors of dimension C, each representing a sub region of a box, then do:

$$v_{k,i,j} = W \cdot b_{k,i,j}, \quad W \in \mathbb{R}^{D\times C} \tag{2}$$

where $k \in \{-1, 0, ...M\}, i \in \{0, ...H_r\}, j \in \{0, ...W_r\}$. So now we got a "visual-semantic" feature tensor $V \in \mathbb{R}^{(M+1)\times H_r\times W_r\times D}$ which will play a role both in caption generation and similarity loss calculation.

LSTM decoder The LSTM decoder is for caption generation. The global feature tensor b_{-1} will be first linearly mapped to a vector $h_{-1} \in \mathbb{R}^D$. The decoder then takes h_{-1} as initial hidden state, and a $START$ token as initial input x_0. Every time we got a new hidden state $h_t, t \in \{0, ..., T-1\}$, it is treated as a query for the attention on V. The attention mod-

ule then returns a vector z_t, which is then concatenated with h_t and passed through a linear and Softmax layer to obtain the output probability vector y_t. Formally, we have:

$$\begin{aligned} h_t, c_t &= f(h_{t-1}, c_{t-1}, x_t) \\ a_t &= Softmax(\tilde{V} \cdot h_t) \\ z_t &= \tilde{V}^T \cdot a_t \\ y_t &= Softmax(W_o \cdot [h_t,\ z_t]) \\ x_{t+1} &= y_t \end{aligned} \tag{3}$$

Here we use $\tilde{V} \in \mathbb{R}^{((M+1)H_r \cdot W_r) \times D}$, a reshaped version of V. The "[ ]" denotes concatenation operation. The initial cell state vector $c_{-1} = 0$.

Note here we assign a probability vector y_t to next time step input x_{t+1}. This is because, if we use adversarial-like training scheme, using discrete samples as output hinders gradients propagation, as suggested by (Shen et al., 2017)(Dai et al., 2017). Although sampling-based gradient estimator such as REINFORCE (Williams, 1992) can be adopted, training with these methods can be unstable due to the high variance of the sampled gradient. Instead, we retain the probability vector as output to approximate the discrete training (Hu et al., 2017)(Lamb et al., 2016). This output vector will be taken as next input, and also be fed into the LSTM encoder and LM model for loss calculations.

LSTM Encoder and Similarity Loss The LSTM encoder will be first trained with paired data $\{v_i, x_i\}$, where $v_i \in \mathbb{R}^{H_r \times W_r \times D}$ is a transformed region feature tensor and x_i is a short text description. At test time the encoder will be used to measure the similarity between the decoder output y and *a set* of boxes $\{v_k\}$. Below we always assume we have multiple boxes for loss computing, and the case of single box will fit naturally.

Consider the pair $\{\{v_k\}, y\}, k \in \{0, ..., M-1\}$, where y is either a set of words of a set of probability vectors output by decoder. The LSTM encoder convert y into a sequence of hidden vectors $e_i \in \mathbb{R}^D, i \in \{0, ..., T-1\}$. So we got matrices $e \in \mathbb{R}^{T \times D}$ and $v \in \mathbb{R}^{M \times H_r \times W_r \times D}$. Reshaping v into 2 dimensions we got $\tilde{v} \in \mathbb{R}^{(M \cdot H_r \cdot W_r) \times D}$. First we calculate the similarity matrix for all possible pairs of words in the sentence and sub-regions in the image by:

$$s = e \cdot \tilde{v}^T \in \mathbb{R}^{T \times (M \cdot H_r \cdot W_r)} \tag{4}$$

where $s_{i,j}$ is the dot-product similarity between the i^{th} word and the j^{th} sub-region.

Now we want the text closely related to the M boxes. In other words, we want every word can find its source from the boxes, and in turn every box should be described in the text. This result in maximizing a bi-directional similarity measure. The first direction is from each word. Specifically, we use each word as a query to give all the sub-regions attention. We have:

$$\begin{aligned} \tilde{v}'_i &= \sum_{j=0}^{M \cdot H_r \cdot W_r} \alpha_{i,j} \tilde{v}_j, \\ \alpha_{i,j} &= \frac{\exp(s_{i,j})}{\sum_{k=0}^{M \cdot H_r \cdot W_r} \exp(s_{i,k})} \end{aligned} \tag{5}$$

where $\alpha \in \mathbb{R}^{T \times (M \cdot H_r \cdot W_r)}$. Here we define relevance for word i to the M boxes, i.e., $R_e(\tilde{v}'_i, e_i) = (\tilde{v}_i'^T e_i)/(||\tilde{v}'_i||||e_i||)$.

For the other direction, we do similar thing but the calculation of attention vector $\beta \in \mathbb{R}^{M \times T}$ is slightly different. Notice that β has dimension $M \times T$, meaning that it has an attention vector for each "box", instead of each "sub-region". This is because we can not expect every sub-region to find its corresponding words. So we do it only on box level. How can we obtain β? We normalize the attention weights w.r.t each box. Formally, we have:

$$\begin{aligned} e'_m &= \sum_{i=0}^{T} \beta_{m,i} e_i, \\ \beta_{m,i} &= \sum_{u \in Reg^{(m)}} \frac{\exp(s_{i,u})}{\sum_{l \in Reg^{(m)}} \sum_{k=0}^{T} \exp(s_{k,l})} \end{aligned} \tag{6}$$

where $Reg^{(m)} = \{i | m \cdot (H_r \cdot W_r) < i < (m+1) \cdot (H_r \cdot W_r) - 1\}$ denotes the set of sub-regions for m^{th} box. Then we define the relevance for each box w.r.t the sentence, i.e., $R_v(v_m, e'_m) = \frac{1}{H_r \cdot W_r} \sum_{u \in Reg^{(m)}} (\tilde{v}_u^T e'_m)/(||\tilde{v}_u||||e'_m||)$. Note the difference between v an $\tilde{v}$ here.

The matching score between the the set of boxes (Q) and the whole text description (D) is then defined as:

$$\begin{aligned} R(Q, D) = \log(\frac{1}{T} \sum_{i=0}^{T-1} \exp(R_e(\tilde{v}'_i, e_i))) + \\ \log(\frac{1}{M} \sum_{m=0}^{M-1} \exp(R_v(v_m, e_m))) \end{aligned} \tag{7}$$

Finally, we define the loss for a batch of boxes-text pairs $\{(Q_i, D_i)\}_{i=1}^N$, the conditional probability of sentence D_i matching with image Q_i is:

$$P(D_i|Q_i) = \frac{\exp(R(Q_i, D_i))}{\sum_{j=1}^N \exp(R(Q_i, D_j))} \quad (8)$$

We can define $P(Q_i|D_i)$ in a mirrored way. Note there is a trick when calculating the denominator of $P(Q_i|D_i)$. That is, when summing over mismatching boxes, we can consider also the boxes proposed in the same image but not overlapped with the boxes we select as the mismatching boxes.

Then the loss is defined as:

$$\mathcal{L}_1' = -\sum_{i=1}^N P(D_i|Q_i) - \sum_{i=1}^N P(Q_i|D_i) \quad (9)$$

We add a constraint to make the attention vector orthogonal to each other among those boxes we select for each image. This is to make sure the phrases in a sentence describing each box are not overlapped to each other. Denote $\beta^{(i)} \in \mathbb{R}^{M^{(i)} \times T^{(i)}}$ as the attention matrix from the second direction for image i. Then for a batch data we have the regularization loss:

$$\mathcal{L}_{reg} = \sum_{i=1}^N \|\beta^{(i)}\beta^{(i)T} - diag(\beta^{(i)}\beta^{(i)T})\|_F \quad (10)$$

Note when the number of boxes considered for a sentence is 1, $\mathcal{L}_{reg} = 0$. This happens when we train the LSTM encoder.

The whole loss is then

$$\mathcal{L}_1 = \mathcal{L}_1' + \mathcal{L}_{reg} \quad (11)$$

When training LSTM encoder we will only use $\mathcal{L}_1$. For training LSTM decoder we need to also consider the language model loss below.

Language Model and LM Loss Inspired by (Yang et al., 2018), we use a language model loss to make the generated captions more fluent. The language model is simply a LSTM which takes a set of probability vectors y generated by the decoder, and calculate the probability of y being a proper sentence. It's trained with the full image caption dataset (Krause et al., 2016) in Visual Genome(Krishna et al., 2017). Then, when use it as a criterion training LSTM decoder, the loss is:

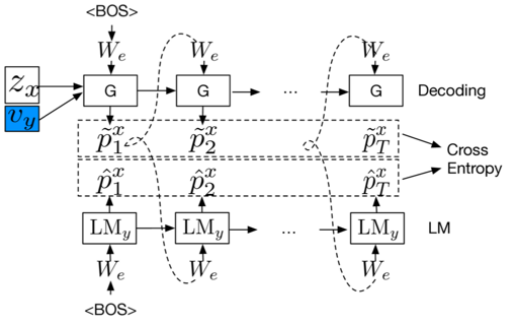


Figure 3: Language Model.

$$\begin{aligned}\mathcal{L}_{LM} &= \mathbb{E}_{y\sim Y}[-\log p_{LM}(y)] \\ &\approx \frac{1}{N}\sum_{i=1}^N -\log p_{LM}(y^{(i)})\end{aligned} \quad (12)$$

where $p_{LM}(y)$ denotes the probability of y measured by the language model.

The total loss for training the LSTM decoder is:

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_{LM} \quad (13)$$

Gumbel-softmax as a differentiable sampling method The similarity loss can't back-propagate gradients to the LSTM decoder if the outputs of the LSTM decoder were discretely sampled, e.g. using "argmax" on the predicted softmax probability vector. To solve this problem, we use a Gumbel-softmax (Jang et al., 2016) distribution as continuous approximation of the discrete samples instead. Let u be a categorical distribution with probabilities $\pi_1, \pi_2, ..., \pi_c$. Samples from u can be approximated using:

$$p_i = \frac{\exp\left(\log \pi_i + g_i\right)/\tau}{\sum_{j=1}^c \exp\left(\log \pi_i + g_i\right)/\tau} \quad (14)$$

During the training of decoder, at every time step a Gumbel-softmax probability vector p_i will be generated and be fed into the pretrained LSTM encoder for similarity loss calculation. It will also be used as the next-time-step input of the decoder.

4 Experiments

4.1 Dataset

We perform our experiments using the Visual Genome (VG) region captions dataset (Krishna et al., 2017), which contained 94,313 images and 4,100,413 snippets of text (43.5 per image), each

		Image annotation				Image search			
data	model	R@1	R@5	R@10	Med r	R@1	R@5	R@10	Med r
train	DAMSM	0.041	0.162	0.249	68.3	0.037	0.140	0.239	69.5
	Our Model	**0.098**	**0.316**	**0.477**	**29.1**	**0.088**	**0.308**	**0.452**	**30.206**
test	DAMSM	0.032	0.153	0.213	74.9	0.027	0.132	0.225	76.3
	Our Model	**0.081**	**0.27**	**0.415**	**38.615**	**0.069**	**0.263**	**0.397**	**39.824**

Table 1: Image-Sentence ranking experiment results. R@K is Recall@K (high is good). Med r is the median rank (low is good). In the table, The size of sample training and testing data set are both 1000. And the statistics are obtained from 5 group of sample data

grounded to a region of an image. Example captions from the dataset include cats play with toys hanging from a perch, news- papers are scattered across a table, woman pouring wine into a glass, mane of a zebra, and red light.

We also leverage the full image captions for 19,551 images in the VG dataset which has been used in and open-sourced by (Krause et al., 2016). We use it both for language model training and LSTM encoder learning.

4.2 Image-text ranking

We investigate the quality of the inferred text and image similiarity with ranking experiment. We consider a withheld set of images and text and retrieve items in one modality given a query from the other by sorting based on the image-text score. We report the median rank of the closest ground truth result in the list and Recall@K, which measures the fraction of times a correct item was found among the top K results. The result of these experiments can be found in the following table. And the visualization of image-text score matrices is also shown in the table1. We now highlight some of the takeaway.

Our model outperforms the DAMSM model(Tao Xu et al., 2018). There are mainly three improvements compared with previous work. Firstly, we consider single text-multiple images pair. In the previous work, the DAMSM model only takes single text-single image pair into consideration, while in our model, we propose a new similarity loss metric to quantify the score between inferred text and multiple corresponding images. Secondly, we add the normalization with respect to both image and text, which improve the performance of image annotation and image searching. Finally, we add regularization part in loss computation.

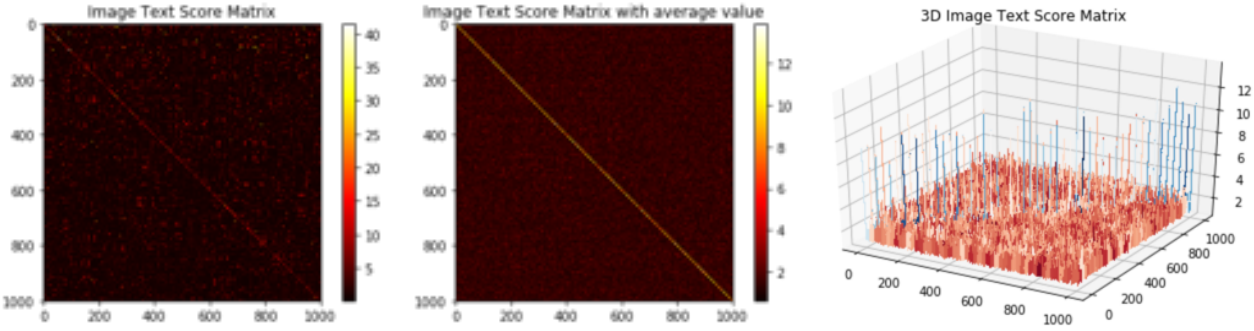
4.3 Language Model Collapse in Training

When training the decoder, the language model failed to guide the decoder to generate fluent sentences in our experiments. We observed from our experiments that the decoder tends to generate repeated token sequences. These tokens are usually the most frequently used words in the training set. We call this problem language model collapse.

We did an experiment to explain this phenomenon. We gave a sequence of tokens to the language model and observed the next word suggestions that the language model gave. We used two different input sequences. One was a random sentence from the training set, the other was a random token sequence. We tested both input sequences 1000 times respectively and recorded the most frequent words. Figure 5 shows the top 30 frequent words generated by the language model. We concluded from the experiment results that our language model tends to suggest the next words being high frequency words when given a random input sequence. This is the reason for language model collapse, because at the start of training, the decoder generated random sentences, as a result, the language model will guide the decoder to converge to high frequency word sequences.

4.4 Image captioning

Experiment set-up We use 500 random images from VISUALGENOME(VG) for this experiment. We generate 100 test queries by repeated sampling captions from some images. And To confirm the diversity of the image captioning,we generate different captions in terms of content, detail and from. For each image, we use different number of feature boxes(2,4) to generate different detail's captions. Different feature boxes have been used to generate different content's captions. And every time we generate captions from the same image, their form will be different.



(a) (b) (c)

Figure 4: Image text score matrix, the brighter diagonal line represents the better performance of model

Method	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	ROUGE-L	METEOR
Train	1.20E-03	9.52E-11	2.09E-13	1.08E-14	1.46E-02	1.18E-02	6.39E-03
Test	6.35E-03	6.77E-11	1.64E-13	8.93E-15	8.17E-03	6.73E-03	4.44E-03

Table 2: Evaluation Scores based on Metrics BLEU(1-4), CIDEr, ROUGE-L and METEOR

Model Set-up We first pre-train an LSTM Encoder using similarity loss between boxes features and regional captions. After that we train our Language Model using 20,000 full captions from VisualGenome and 600,000 image annotations from MSCOCO.Finally we train our LSTM Decoder whose outputs were supervised by the pre-trained LSTM Encoder and Language Model.

Results Figure 6 gives a sample output from an image that showing diversity in three aspects. The words in these generated sentences were closely related to the red boxes given (as shown by the bold text). For example, the captions at top-right corner covers the key elements in the one red box given such as ”groups”, ”crowded”, ”woman” and ”enjoys”. And once we include other boxes containing buildings and trees such as two samples on the left, the keywords ”multistoried” and ”trees” will be included apart from the key words mentioned above.

Please check out Appendix A for more sample outputs.

However, the drawback is that the sentences are all broken. Failing to involve language model into our training is one main reason. Yet another good question to answer is, why we can’t learn fluent expression simply with similarity loss?

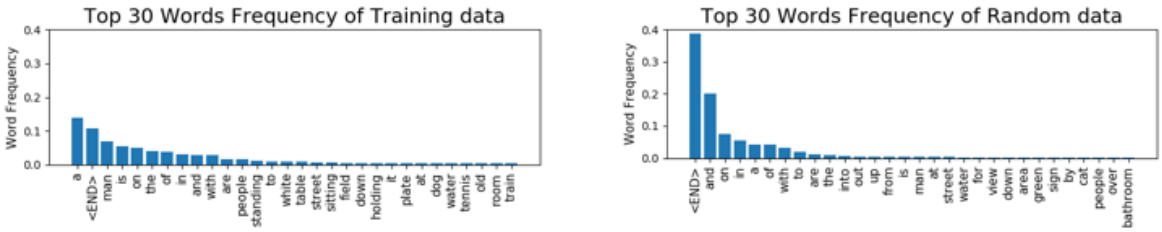
Why we can’t learn fluent captions with only similarity loss There used to be previous works successfully learned fluent image captions with only a similarity metrics, such as (Dai et al., 2017), where they also first pretrained a ”E-GAN” as a similarity metric measuring visual-text distances, and use it as the only criterion for decoder training. However, our model is different and we made the following hypotheses for future investigation about why our similarity loss can’t make the decoder learn fluent expression. In short, there are gaps between our designed objective and the expected objective, in the following three aspects:

1) The LSTM encoder was trained with short phrases from Visual Genome dataset but not full sentences. Since it has never seen what a full sentence looks like, we can’t expect the decoder to learn generating full sentence from encoder.

2) We trained the LSTM encoder on one-box-one-caption pairs, but when training decoder, we turn on the multi-box-one-caption option. We suspect this may be a gap making the performance worse.

3) We trained the LSTM encoder using discrete data, but since we use Gumbel-softmax to tackle gradient propagation issue, the encoder take continuous data as inputs during decoder training phase. Our experiments on language model training shows that taking continuous inputs at test time will confuse the LSTM and causing the output from LM to be random, indicating Gumbel-softmax may not be a perfect solution here.

Evaluation We evaluate our generated captions based on four conventional metrics, including n-gram BLEU (Papineni et al., 2002), METEOR(Denkowski and Lavie, 2011), CIDEr(Vedantam et al., 2015) and ROUGE-L(Lin, 2004), which are adopted by a large number of previous works. They provide ways to measure the closeness of generated captions to one or multiple



(a) (b)

Figure 5: (a) Word frequency of top 30 frequent words using training sentences as input sequences (b) Word frequency of top 30 frequent words using random token sequences as input sequences

Figure 6: Sample outputs. From left to right the level of detail decreases (from 3 boxes to 1 box). From top to bottom the contents changed but the level of detail keeps the same (same number of boxes but different boxes). For a given set of boxes, we generate two captions in different forms (different word choices and sentence structure).

references. In particular, we use METEOR to compare performance of our model to that of Fully Convolutional Localization Network(FCLN) as it is found to be most highly correlated with human judgments in settings with a low number of references. And we only have one caption for each region. Generated captions of regions will be evaluated with ground truth of correlated regions of 500 images. We also give appropriate length penalty for the evaluation metrics. Below is the evaluation result.

Compared to the meteor score 0.305 of FCLN, our model gives quite low evaluation score around 0. It is quite as expected due to failure in learning fluent sentence expression from language model.

5 Conclusion

In this project we proposed a novel model that can generate image captions that are diverse in three aspects: the form, the level of detail, and the content. Our trained model was able to generate relevant words to the boxes selected, but not logical sentences due to LM failure. Our novel similarity loss achieved significantly better performance in image-text ranking task compared with referred previous work (DAMSM). We also investigated into the language model failure and found that the LM loss will collapse when fed with random or noisy sentences, hence not a ideal criterion for natural language generation.

6 Contribution Statements

Chuan Cen designed the model architecture. Yuan Liang and Chuan Cen designed and implemented similarity loss metric. Yuan Liang carried out image-text ranking experiments and visualization. Chuan Cen wrote the progress report. Chuan Cen implemented the feature extractor, dataloader and

encoder pre-training script. Xuanyu Wang designed and implemented the decoder module. Xuanyu Wang and Chuan Cen implemented the decoder training script. Chuan Cen trained the encoder and decoder. Ruoyao Wang designed and pre-trained the language model, implemented the language model loss. Chuan Cen located the language model collapse problem. Ruoyao Wang designed and carried out the experiment to explain the language model collapse. Fei Yi designed the LSTM encoder and Transformer to encode output sentences for similarity loss. Fei Yi also implemented validation mode of the train script and utilized several metrics to evaluate the generated captions. Fei Yi, Chuan Cen, Yuan Liang, Ruoyao Wang and Xuanyu Wang all contributed to the poster and the final report.

References

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

Bo Dai, Sanja Fidler, Raquel Urtasun, and Dahua Lin. 2017. Towards Diverse and Natural Image Descriptions via a Conditional GAN. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2989–2998, Venice, October. IEEE.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the sixth workshop on statistical machine translation*, pages 85–91. Association for Computational Linguistics.

Andrea Frome, Greg S Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, MarcAurelio Ranzato, and Tomas Mikolov. DeViSE: A Deep Visual-Semantic Embedding Model. page 11.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR. org.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Justin Johnson, Andrej Karpathy, and Li Fei-Fei. 2016. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4565–4574, Las Vegas, NV, USA, June. IEEE.

Andrej Karpathy and Li Fei-Fei. 2015. Deep Visual-Semantic Alignments for Generating Image Descriptions. pages 3128–3137.

Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. 2016. A Hierarchical Approach for Generating Descriptive Image Paragraphs. *arXiv:1611.06607 [cs]*, November. arXiv: 1611.06607.

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

Alex M Lamb, Anirudh Goyal Alias Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Anna Senina, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. 2014. Coherent Multi-Sentence Video Description with Variable Level of Detail. *arXiv:1403.6173 [cs]*, 8753:184–195. arXiv: 1403.6173.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2014. Show and Tell: A Neural Image Caption Generator. *arXiv:1411.4555 [cs]*, November. arXiv: 1411.4555.

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

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018.

Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1316–1324.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems*, pages 7298–7309.

Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. 2016. Image Captioning With Semantic Attention. pages 4651–4659.

Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593.

Mingxing Zhang, Yang Yang, Hanwang Zhang, Yanli Ji, Heng Tao Shen, and Tat-Seng Chua. 2019. More is better: Precise and detailed image captioning using online positive recall and missing concepts mining. *IEEE Transactions on Image Processing*, 28(1):32–44.

Appendices

A Sample outputs