NN_PJ3

YiFan Li 19307110499

May 2022

Contents

1 Introduction

Given an image, we can easily infer the main entities in it, and describe the scene effectively, such as, the location of objects, the attributions of objects, and, furthermore, the interaction between objects and other objects in a scene (running in a field, or being held by a person etc.). The task of visual description aims to develop visual systems that generate contextual descriptions about objects in images. Visual description is challenging because it requires recognizing not only objects (kids), but other visual elements, such as actions (playing) and attributes (young), and constructing a fluent sentence describing how objects, actions, and attributes are related in an image like a human (such as image NO.248703 in COCO dataset: young kids that are playing a game of soccer).
In addition to the problems described above, there is another difficulty in this task. Current visual description or image captioning models have worked quite well, but most of them can only describe objects seen in existing training datasets, and they require a large number of training examples to generate good captions. But in reality, we cannot have a large enough dataset to train our model to deal with all possible scenarios.
Recent works in object recognition through Convolutional Neural Networks (CNNs) can recognize hundreds of categories of objects. While object recognition models can recognize novel objects, description models cannot compose sentences to describe these objects correctly in context. So in this task, I will overcome this problem by building visual description systems which can describe new objects without pairs of images and sentences about these objects.
This task can reflect the generalization ability of models in understanding and captioning the semantic meanings of visual concepts and objects unseen in training set. Here I define this task more formally. Given MSCOCO dataset consisting of paired images and descriptions, we wish to learn how to describe objects unseen in paired image-sentence data. To do this we must build a model which can recognize different visual constituents (e.g., bottle, bus) and compose these in novel ways to form a coherent description. Below I will explain the implementation process of this task.

2 Data Prepare and Preprocess

2.1 Data Prepare

Before starting the task, to fimiliar with the dataset to be used this time, MSCOCO[1] using DCC split[2].
The original MSCOCO dataset (2014) has 164062 images, including 82783 training

images, 40504 validation images and 40775 test images. There are 1.5 million object instances, 80 object categories and 91 stuff categories.
DCC split made some changes to dataset construct. Author selected eight categories of objects, "bottle", "bus", "couch", "microwave", "pizza", "racket", "suitcase" and "zebra" as OOD objects. The training set excludes all paired image caption that describe at least one of the 8 novel objects. 50% of the paired image caption containing novel objects were selected from the original MSCOCO validation set for val set, and another 50% are reserved for testing. After split, the size of the dataset has also changed greatly. The validation set only contained 3018 verification images, and the test set only contained 3024 test images.
Each image in the dataset is matched with five captions. The annotation content has the following characteristics:

- Describe all important parts of the scene.
- Do not describe unimportant details.
- Do not describe what may happen in the future or in the past.
- Don't describe what a person might say.
- Exclusive names are not provided.
- These sentences should contain at least 8 words.

A group of kids that are playing soccer on a field.
A soccer player getting ready to kick the soccer ball.
Some teenagers are playing in a soccer game.
young kids that are playing a game of soccer
young males wearing soccer uniforms playing in a soccer field

Figure 1: No.248703 image and paired five captions

From the above features, we can see that the captions of MSCOCO dataset are relatively regular. Perhaps we can imitate this regularity and use templating to generate regular but natural captions. Therefore, I refer to template-based caption model, Neural Baby Talk[3], to complete this task.

2.2 Data Preprocess

Data preprocessing is divided into two parts: image processing and text processing.

2.2.1 Image Preprocess

Resize all images in training set size to 576×576, and then randomly resize to 512×512.
The images in the validation set and test set are resized to 512×512 directly. Then all images are normalized.

2.2.2 Text Preprocess

The task model is a mixture of visual and language models. The input of NLP model needs a series of preprocessing for the original text data. Text preprocessing can be divided into the following steps:

1. **Build word to id vocabulary.** This step is to obtain the embedding vector and recover the corresponding word from the embedding vector.
   It is necessary to weaken feature of words, whose frequency in the training set is lower than the preset threshold, by transforming them to $< UNK >$. The reason for this step is that the words with low frequency have little benefit on the training model because them carry less information.
   Another reason is that out of vocabulary (OOV) words are likely to appear outside the training set. In order for the model to operate normally while encountering these words, these OOVs need to be simulated in the training set, $< UNK >$ is the OOV words simulated in the training set.

2. **Record word stemming form.** The same word has different forms in different modes, which will increase the number of parameters that the model needs to learn. Words with different modality should have similar semantics in sentences. Word stemming helps to reduce complexity and retain the essence of word meaning.
   In the later language model, we can determine the rough label of words first, and then consider the plurality and finegrained class to restore.

3. **Word embedding**. Convert word in captions into vector form of model input. In order to generate captions about diverse categories of objects outside the image-caption training data, we take advantage of external data sources. Specifically, we need an external corpus to help us pretrain embedding vectors. Here, I use the pretrained 300-dim GloVe vector[4].

3 Model Structure

Model basic idea: the object detector is used to detect the objects in the image (visual words), and then at the generation time of each word, the model can decide whether to select text word or visual word.
The following figure is a simple schematic diagram of the model structure:
The specific implementation method is as follows:

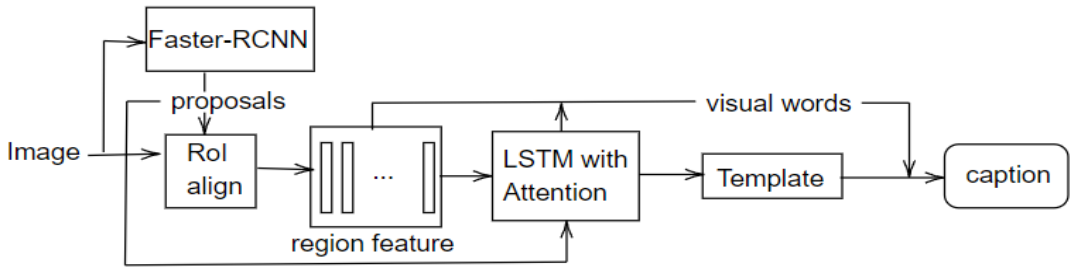

Figure 2: Model structure

Using the Top-Down attention model, learn two groups of word probability: text words and visual words. The probability of text words is basically consistent with the attention model; The visual words probability uses the correlation between the current hidden state and region features to learn region weights (i.e., attention). Each region corresponds to a word.
When generating captions, first use LSTM as the model to generate template, which uses feature maps output by ResNet101 as its input. In addition, the model learns from the practice of adaptive attention to learn a sentry gate to determine whether text or visual word should be used at the current time, which is realized by scaling the probability of visual words. Select a word at each time step by finding the word with maximum probability. Finally, a template shaped as "a <region-2> is laying on the <region-4> near a <region-7>" is generated.
After the above template is generated, the specific vocabulary of each slot can be predicted according to the feature v_i of the region corresponding to each slot and the current hidden state h_i. For novel object words, simply replace the new object's word embedding with an existing one which belongs to the same super category in training dataset (e.g., bus $\leftarrow$ car).
After the basic word is generated, it still need to be transformed to adapt to the current text context, such as plural, morphology, etc. The author considers two transformations: plural (such as dog and dogs) and fine-grained tags of categories (such as dog can be subdivided into puppy, etc.). The two transformations are

realized by learning two classifiers respectively. The plural classifiers use binary classifiers, and the fine-grained classifiers use multiple classifiers.

4 Evaluation Metrics and Results

4.1 Hyper-parameters

The following are some parameter settings during training:

rnn_hidden_size	1024	**rnn_layers**	1
rnn_input_size	512	**fc_feat_size**	248
att_feat_size	248	**epochs**	3
batch_size	20	**grad_clip**	0.1
drop_prob_lm	0.5	**optimizer**	Adam
learning_rate	5e-4	**learning_rate_decay_start**	1
learning_rate_decay_every	3	**learning_rate_decay_rate**	0.8
optim_alpha	0.9	**optim_beta**	0.999
weight_decay	0	**cnn_finetune_block**	1
cnn_optimizer	Adam	**cnn_optim_alpha**	0.8
cnn_optim_beta	0.999	**cnn_learning_rate**	1e-5
cnn_weight_decay	0		

In addition, the following parameters need to be set during the test:

beam_size	3	**cbs_mode**	novel
cbs_tag_size	3	**det_oracle**	True

4.2 Training Process

The following is the visualization of loss during training:

4.3 Evaluation Metrics

Compared with other evaluation indicators, CIDEr[5] is closer to the principle that human beings judge whether two sentences are similar. It uses TF-IDF to give different weights to different n-grams. Intuitively, the weight of frequently occurring phrases is lower, while the infrequent phrases are more special (with greater weight). People will pay more attention to these special words. CIDEr is calculated as follow:

$$CIDEr_n(c_i, S_i) = \frac{1}{m} \sum_j \frac{g^n(c_i) \cdot g^n(s_{ij})}{||g^n(c_i)|| \cdot ||g^n(s_{ij})||}$$

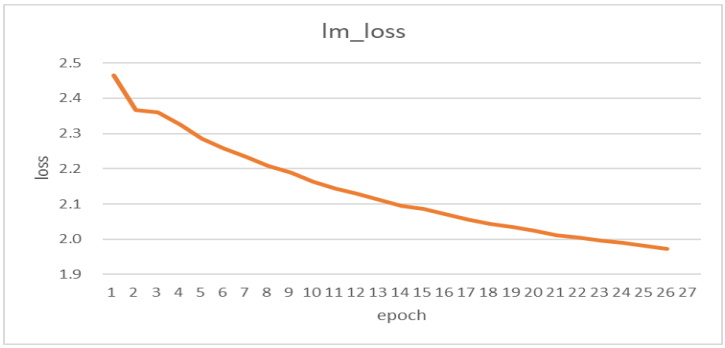

Figure 3: lm_loss in training process

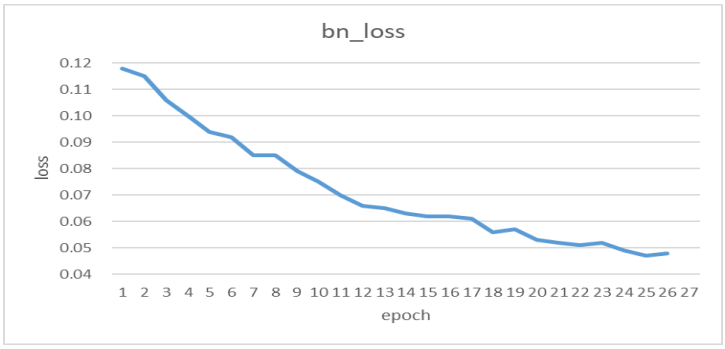

Figure 4: bn_loss in training process

Where c_i, S_{ij} are candidate caption and reference captions of image i, $g^n(c_i)$ is n-gram of c_i, $g^n(s_{ij})$ is n-gram of S_{ij}, m is the number of reference captions of image i.
CIDEr-D is a modified version of CIDEr, which aims to prevent the gaming of evaluation indicators. The gaming problem is that for a certain evaluation index, the score given by human is very low, while the score given by the evaluation index is very high. CIDEr-D mainly consists of the following changes:

1. The step of converting the words in the sentence to the original form is removed.

2. In the CIDEr evaluation index, the higher the CIDEr score is for the long sentences obtained from some repeated words. In order to reduce this effect, CIDEr-D adds a gaussian factor to punish the case where the length difference between candidate and reference sentence is large.

3. Some sentences that repeat words with high confidence (i.e. sentences with more information, such as a fish in the picture) are punished.

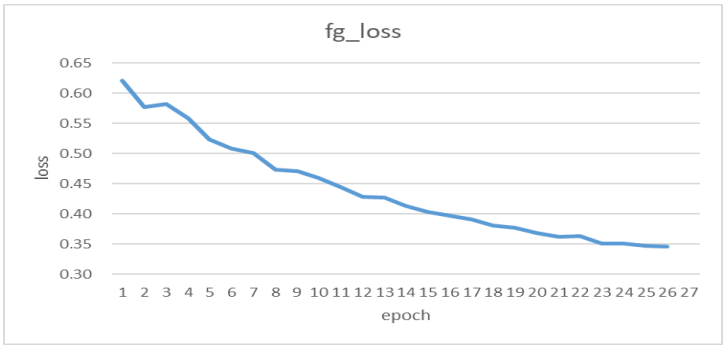

Figure 5: fg_loss in training process

The second evaluation index is METEOR[6], which calculates the quality of captioning by matching candidate caption and the best reference caption. It calculates the accuracy rate P and recall rate R of the two sentences by unigram, and uses the number of chunks formed by the continuously aligned words in the two sentences to calculate the penalty factor. Therefore, it takes into account the accuracy and recall rate based on the whole corpus, the fluency of sentences and the influence of synonyms on semantics. The calculation formula is as follows:

$$
\begin{aligned}
Pen &= \gamma(\frac{ch}{m})^{\theta} \\
F_{mean} &= \frac{P_m R_m}{\alpha P_m + (1\alpha) R_m} \\
P_m &= \frac{m}{\sum_k h_k(c_i)} \\
R_m &= \frac{m}{\sum_k h_k(s_i j)} \\
METEOR &= (1 - Pen) F_{mean}
\end{aligned}
$$

Where M is the total pair matched by candidate caption and reference caption, ch is the total chunk, and h_k calculates the length of sentence.

SPICE[7] uses graph based semantic representation to encode objects, attributes, and relationships in caption. It first parses candidate caption and reference captions into synchronous dependency trees with Probabilistic Context-Free Grammar (PCFG) dependency parser, and then maps dependency trees into scene graphs with rule-based method. Finally, calculate the F-score values of objects, attributes and

relationships in the caption to be evaluated. The calculation formula is as follows:

$$SPICE(c, S) = F_1(c, S) = \frac{2P(c, S)R(c, S)}{P(c, S) + R(c, S)}$$
$$P(c, S) = \frac{|T(G(c)) \otimes T(G(S))|}{|T(G(c))|}$$
$$R(c, S) = \frac{|T(G(c)) \otimes T(G(S))|}{|T(G(S))|}$$

Where c is candidate caption, S is the set of reference captions. $G(\cdot)$ convert text into a scene graph, and $T(\cdot)$ convert a scene graph into a set of tuples.

4.4 Beam Search and Best result

When evaluating the model, we need to generate the caption first. The most direct method is to send the image features to trained LSTM, and obtain the sentence with the highest joint probability according to greedy search.
Here, I use beam search to improve greedy search. It can expand the search space, so it is easier to get the global optimal solution. Beam search keeps the N sequences with the highest scores at each time step, and continues to generate words with these N candidate sequences in the next step.
Obviously, if the probabilities of all generated words are correct, the larger the beam size, the greater probability of finding the global optimal solution. But the biggest problem of brute force method is that the time complexity is unacceptable. Beam search achieves a balance between greedy search and brute force method.
The following problem is how to select the beam size of beam search. The previous conclusion, the larger the beam size the better the search, is based on the correct probability of all generated words. However, in the deep learning, our model only learns the empirical value approaching the real value. Such errors will also lead to error. The larger the beam size, the higher the bias. Therefore, it is necessary to balance both search space size and search validity. The beam size should not be too large or too small. Here, the performance of the model is tested when the beam size is 2 and 3.

generate method	**CIDEr**	**METEOR**	**SPICE**
Greedy Search	63.5	23.6	16.1
CBS with beam size 2	69.5	23.7	18.3
CBS with beam size 3	73.1	24.0	18.9

Table 1: Three standard automatic evaluations metrics

generate method	bus	bottle	couch	microwave	pizza	racket	suitcase	zebra
greedy search	65.55	13.07	41.05	65.19	40.79	11.61	42.75	88.52
beam size 2	90.41	66.42	89.32	91.08	92.33	84.56	88.62	96.6
beam size 3	91.24	78.02	88.92	90.41	94.73	93.37	84.65	96.7

Table 2: F1 score of the performance of captioning eight novel concepts

generate method	average
greedy search	46.07
CBS with beam size 2	76.17
CBS with beam size 3	89.76

Table 3: Average F1 score

Greedy search can also be regarded as beam search with beam size of 1. It can be seen that the beam size ranges from 1 to 3, the overall evaluation metrics have increased, and a few indicators (such as some F1 score of novel objects) have decreased.

Figure 6: No.480076 Image

Captions generated from this image:

- After 7 epochs with greedy search: a black and white photo of a person sitting on a chair
- After 22 epochs with greedy search: a bedroom with a wooden floor and a

wooden table

- After 28 epochs with greedy search: a bedroom with a guitar and a guitar
- After 30 epochs with beam search: a cat laying on a chair in a room

It can be seen that although the sentences generated by the model become more and more fluent with the training process, beam search is still the best choice to find a better solution combining image and text information.

5 Future Works

In the process of model training, I have found that with the progress of training, the metric evaluation is gradually increasing, but the F1 score is declining very severely. It is understandable that the F1 score is high at the beginning, because the visual model ResNet101 used is pretrained, but when the model is adjusted on the coco image caption data set, it seems that it will gradually forget what it has seen before. Perhaps in the future, I can use joint training strategies to solve the problem of forgetting. Specifically, the model is divided into different components, and all components share parameters and train together. During the training, each batch of input contains some images with labels, a different set of images and captions, and some common sentences. These three inputs are used to train different components of the network. Because the parameters are shared among the three components, the network is jointly trained to identify objects in the image, add captions to the image and generate captions. Perhaps this joint training can help the network overcome the forgetting problem and enable the model to generate descriptions for many new object classes.

References

[1] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.

[2] L. A. Hendricks, S. Venugopalan, M. Rohrbach, R. J. Mooney, K. Saenko, and T. Darrell, "Deep compositional captioning: Describing novel object categories without paired training data," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–10, 2016.

[3] J. Lu, J. Yang, D. Batra, and D. Parikh, "Neural baby talk," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7219–7228, 2018.

[4] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[5] R. Vedantam, C. L. Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4566–4575, 2015.

[6] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *IEEvaluation@ACL*, 2005.

[7] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," in *ECCV*, 2016.