Image Caption Generator using Deep Learning on Flickr8K dataset

Project Report

Submitted By

Akriti Jaiswal (216102108), Sujeet Kumar Singh (214363009)

Indian Institute of Technology Guwahati
Asaam, India-781039
May, 2022

Table of Contents

List of Figures

1 Introduction and Problem Statement

Automatically generating captions to an image shows the understanding of the image by computers, which is a fundamental task of intelligence. For a caption model it not only need to find which objects are contained in the image and also need to be able to expressing their relationships in a natural language such as English. Recently work [1] also achieve the presence of attention, which can store and report the information and relationship between some most salient features and clusters in the image [2]. In Xu's work [3], it describe approaches to caption generation that attempt to incorporate a form of attention with two variants: a "hard" attention mechanism and a "soft" attention mechanism. In our project, we do image-to-sentence generation. This application bridges vision and natural language. If we can do well in this task, we can then utilize natural language processing technologies understand the world in images. In addition, we introduced attention mechanism, which is able to recognize what a word refers to in the image, and thus summarize the relationship between objects in the image. This will be a powerful tool to utilize the massive unformatted image data, which dominate the whole data in the world.

2 Image Caption Generation with Attention Mechanism

2.1 Extract features

The input of the model is a single raw image and the output is a caption y encoded as a sequence of 1-of-K encoded words.

y = $y_1, ..., y_C$, y_i $\mathcal{E}R^k$

Where K is the size of the vocabulary and C is the length of the caption. To extract a set feature vectors which we refer to as annotation vectors, we use a convolutional neural network [3].

a = $a_1, ..., a_L$, a_i $\mathcal{E}R$

The extractor produces L vectors and each element corresponds to a part of the image as a D-dimensional representation. The feature vectors was extract from the convolutional layer before the fully connected layer. We will try different layers such such as convolutional layers to compare the result and try to choose the best layers to produce feature vectors that contains most precise in formation about relationship between salient features and clusters in the image.

2.2 Caption generator

The model use a long short-term memory (LSTM) network that produces a cation. At every time step , we will generate one word conditioned on a context vector, the previous hidden state and the previously generated words.

The model define a mechanism that computes z^t from annotation vectors $a_i, i = 1, ..., L$ corresponding to the features extracted at different image locations. And z^t is a representation of the relevant part of the image input at time t4. For each location i, the mechanism generates a positive weight $-i$ that can be interpreted as the relative importance to give to location i in blending the i's together. The model compute

the weight i by attention model f_att for which the model use multilayer perceptron conditioned on the previous state $ht1$.

2.3 Loss Function

We use a word-wise cross entropy as the basic loss function l_0. Further more, to encourage the attention function to produce more expressive output, we define l_1, l_2 as the variace of t along the seqence axis and spacial axise correspondingly. Then define the overall loss function as $l = l_0 + _1 l_1 + _2 l_2$, where $_1$ and $_2$ are hyperparameters.

3 Architecture

CNN features have the potential to describe the image. To leverage this potential to natural language, a usual method is to extract sequential information and convert them into language.. In most recent image captioning works, they extract feature map from top layers of CNN, pass them to some form of RNN and then use a softmax to get the score of the words at every step. Now our goal is, in addition to captioning, also recognize the objects in the image to which every word refers to. In other word, we want position information. Thus we need to extract feature from a lower level of CNN, encode them into a vector which is dominated by the feature vector corresponding to the object the word wants to describe, and pass them into RNN. ai is the feature vector from CNN. We get these feature map from the inception-5b layer of google net, which means we have 6x6 feature vectors with 1024 dimensions. Firstly, we use function finit, h and finit, c to generate initial hidden state and cell state for the LSTMs. Input of LSTM0 is word embeddings. Input of LSTM1 is h0, which is the output of LSTM0, concatenated with attention vector z, which is an weighted average over ai. The weight alpha is computed from the combination of h0, representing information of current word, and each of ai, representing position information. Our labels are only captions of the images. But to get a better caption, the network must force alpha to extract as much information as possible from ai at each step, which means alpha should put more weights on the area of the next word. This alpha is exactly the attention we want. Here for finit, h, finit, c, we use multilayer perceptrons(MLP). For fatt, we use a CNN with 1x1 filters. To further reduce the influence of the image information as a whole and thus put more weight on attention information, we build a new model where we send ai directly to the first input of LSTM0 throug a MLP, and initialize h and c as 0. An even more extreme model is to only use z as information source from the image.

4 Dataset and Features

A new benchmark collection for sentence-based image description and search, consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events.The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations.

For the sentences, we build a word-index mapping, add a “START” and “END”

symbol to its both ends, and add "NULL" symbol to make them the same length. Because some words in the sentences are very sparse, when we generate a vocabulary from the sentences we need to set a threshold to decrease the classification error. The threshold should not only remove the spares words, but also avoid producing too many unknown words when predict the sentences. Thus, we observe the curve of vocabulary size – threshold and the total word size – threshold. The previous one is exponential and the latter one is linear, so we choose 10 as our threshold. For images, we preprocessing them by cropping them to 224x224 and subtract the mean. Because they are already a large number of data, we don't do any data augmentations.

5 Experiment and Result

5.1 Experiment

In our architecture there are 2 parts. One is CNN encoder to map image to features, and the other is LSTM decoder with attention functions, which is a small CNN with 1x1 filters. We didnt finetune the encoder part and only trained the decoder part. To train the decoder, we used adam update. We tried learning rate from 1 to 1e-5 and found 5e-4 with decay rate 0.995 produce a best learning curve. Because feaure maps are smaller than images, and only decoder part was trained, so to make best of GPU memory, we used a large minibatch size of 512 samples.

At the beginning, we can overfit a small dataset with 500 samples. But when we went to full dataset of 8000 samples, we cannot overfit it even we increase the number of hidden units and depth of attention function. Then we adopted a gradually tuning method: train the model on dataset with size of 500, 1000, 2000 and gradually pick our hyperparameters. Finally we got a good model with 3000 training samples, LSTM hidden size of 512 and MLP hidden size for [1024, 512, 512, 512], which generalize decently well.

5.2 Result

We can see that the generated sentences expressed the pictures quite well. The main parts of the images can be recognized and shown in the sentence, and also of the minor parts are also encoded. As for attention, our model is only able to recognize the most important part of the images. That is, the attentions at each step are the same. There are 2 major reasons. Firstly, since the features are input at the first step of LSTM, the overall information of the image has been feed into the decoder, which is enough to generate a decent sentence, and thus the following inputs can be coarser. This is exactly the motivation of our other models. They are potential to work better given more fine tune. Secondly, the receptive field of inception 5 is quite large (139 x 139). So to focus on the main part of image is enough to get a good sentence. To address this problem, we can use lower level features from CNN with more expressive fatt, i.e., to deepen the fatt CNN and enlarge the number of hidden units in each layer. We can observed that it is generating image captions from the given Flickr8K dataset which is shown in Fig.1, 2, 3. In that figures we can observed that it is showing predicted caption and original caption and it is almost equal to original captions. While training our model we got accuracy of 53 %.

Training Results:

After 500 iterations: Cost = 11.360715341567992 and Accuracy: 26.191071930030983

After 1000 iterations: Cost = 8.357436231772105 and Accuracy: 40.116378565629326

After 1500 iterations: Cost = 6.502959014972051 and Accuracy: 50.99914977947871

After 2000 iterations: Cost = 5.692384207248688 and Accuracy: 50.12823591629664

After 2500 iterations: Cost = 4.3318730163077515 and Accuracy: 55.33779750267664

After 3000 iterations: Cost = 4.402987347046534 and Accuracy: 53.16447198390961

Optimization finished!

Let's check...

Figure 1: Image

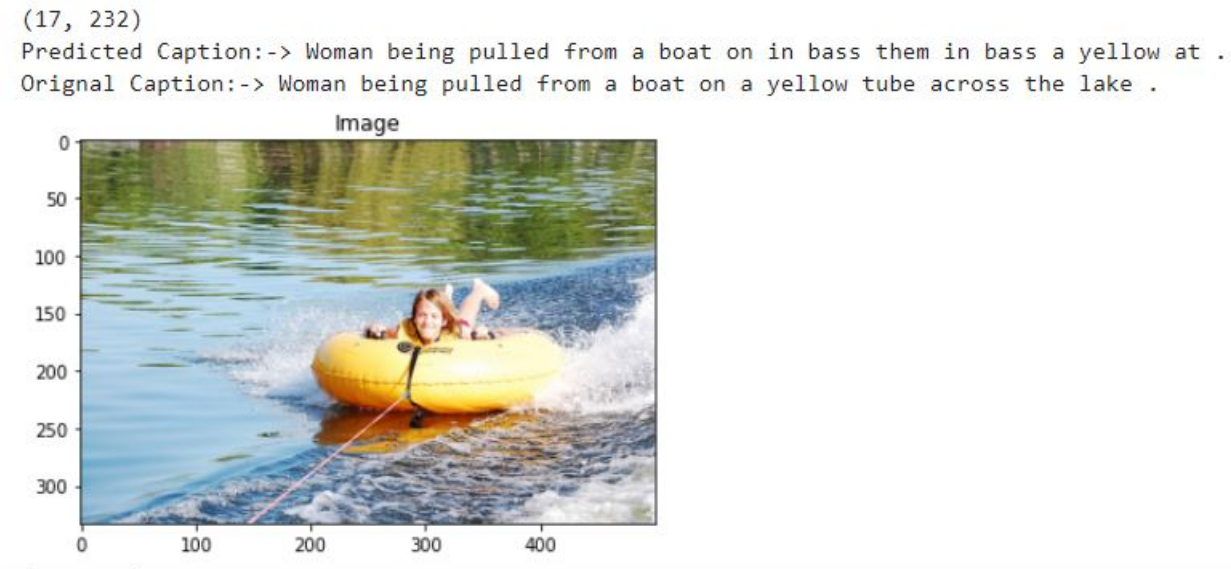

Figure 2: Image

Figure 3: Image

6 Conclusion

In this way using CNN, RNN and LSTM, we got our predicted output which generates image captioning using various deep learning model. We got accuracy of 53 percentage in 3000 iteration. From the result we notice that the attention coefficient are evenly distributed, which means that the model takes the whole picture information to generate the next time step hidden layer via LSTM. But we expect that we can highlight specific part of the picture related to the certain word. To achieve this goal we can use hard attention, which restricts information extraction from image as whole. We can also use a sharper activation function instead of softmax to produce a suitable attention distribution. Moreover, we can label more detailed captions to force the mode to attend smaller parts.

References

[1] Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing. Convolutional image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5561–5570, 2018.

[2] MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CsUR)*, 51(6):1–36, 2019.

[3] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4651–4659, 2016.