Prediction of a Caption from an Image

Riya Shende , ShabduliDurge , GunjanDhurde
Department of Computer Engineering
Cummins College of Engineering for Women
Nagpur, Maharashtra, India

Prof HarshwardhanKharpate
Department of Computer Engineering
Cummins College of Engineering for Women
Nagpur, Maharashtra, India

Abstract - Automatically generating the caption from an image is a very challenging task itself. Predicting a caption from an image requires good knowledge of natural language processing as well as image processing. In this paper, we have discussed about how the growth in the field of machine learning and object detection has helped in creating an image captioning model. We have perform different methods on Flickr8k dataset which contains 8k images. These captions generated form an image can also be used for image classification.

Keywords:Image Captioning, Image Processing, Caption Bot, Convolution Neural Network, Recurrent Neural Network.

1. INTRODUCTION

In the past few years, computer vision in image processing area has made a significant progress like image classification and object detection. With the growth in the field of machine learning and advancement in object detection and image classification, it is possible to generate a sentence automatically to understand the visual content of an image, which is called as Image Captioning. Captioning of image is a very important research field. In order to produce qualitative images descriptions, image captioning system must not only understand what objects are presented but also relationships between them. More than that it must generate human like sentences based on that information. Thanks to the feature of image captioning system can be used in a wide range of practical tasks, such as image search, human computer interaction and help to visually impaired people.

Fig 1 :A small boy playing football

Let's take an example, if you were asked to describe the above image in one single sentence, what would you say? Firstly, you would look at the image and take a note of different objects like a small boy, football, green grass etc. Then on the basis of how objects were placed in a picture, some of people may say "A small boy playing with football", some people may say "Blue top boy spotted playing football on grassy ground" and some others might even say "A small kid playing football on ground" and there might be some other captions also. By using a combination of relation of objects with each other and sequence of text, we can generate a caption, and that's exactly how the model of CNN and RNN architecture works.

2. METHODOLOGY

For predicting a caption, the model can be divided into two modules as Image based module and Language based module. Image based module is used for extracting features which is mainly done by CNN algorithm i.e. Convolutional Neural Network. CNN applies set of filters to each pixel of an input image. Language based module is used for translating features and objects given by the image based module to a sentence which is done by RNN algorithm i.e. Recurrent Neural Network. The job of RNN is to decode the process vectors and turn it into sequence of words by applying various image processing technique to find a pattern in an input image. Hence captioning model is a combination of two separate architecture i.e. CNN and RNN. The input image will be processed by CNN and will connect the output of CNN to input of RNN which will allow us to generate a descriptive text. The procedure that we followed while executing the code was that we initially loaded the datasets and provided paths to them. We stored the dataset paths in variables to be used in further functionality. Then after importing the libraries we loaded the captions in a file and calculated the length of the file, we created a dictionary having the image name as the key and the captions as values. Then we started performing the cleaning phase. In this phase we applied lowercasing,punctuation removal and removed words less than length as 1. Modified all the captions and wrote the clean descriptions to another text file. After loading the newly created description file, the phase of 'Understanding the data' was carried out.

Unique words in the vocabulary were provided as output and frequency of these words was also calculated. Threshold value was used to determine which words occurred less frequently. Complete length was calculated and output came out to be 1845. The next phase was data pre-processing wherein we trained and tested our dataset. We stored the trained data in a dictionary named 'train_descriptions' which consisted of two words added in them that were 'startseq' and 'endseq'. Encoding vectors for both train and test images were found out with the help of Resnet50 model. The encodings were saved on a disk and loaded. The word embedding were calculated with the help of two dictionaries word_to_idx and idx_to_word the latter of which did the vice versa process of ensuring that each word got its index. Even the startseq and endseq words were assigned indices and the vocabulary size was determined. The creation of a matrix to determine the shape of our embedding helped us furthermore to the process of image feature extraction. Here use of functions Embed, Dropout and LSTM also helped a lot. The last steps of our work consisted of creation of model weights and saving it, as well as creation of a function that would help to predict the captions in front of images as output where we select 20 images at random.

3. ALGORITHMS

3.1 CONVOLUTION NEURAL NETWORK (CNN)

Let's first understand what is CNN and why is it important. They are a kind of Deep Neutral Network which were designed from biologically driven models. So what researches found was how a mammal/ a human receives an image into the brain in different layers and that's how Convolution Neural Network was designed.

Now, let's understand how CNN works. There is an input image X. This input image goes through bunch of layers. It will take a patch from the input image and pass to Convolution Layer which apply set of filters. The Convolution Layer as the name say is the core block of the CNN and create different activation features in that input image. Therefore, Convolution Layers are a set of filters that are applied to the given input image. Once the set of filters are applied to input and the output is generated from the convolution which will have the same spatial dimension as the input and the depth of the output will be same as the number of filters. The spatial dimension is also depend on one more factor called stride. Stride decides how you want to process your pixels. So, if the stride value is 1 then it is processing each and every pixel and the dimension of output will be same as the input. But if the value of stride is 2 then it is processing every other pixel in the image. The output is generated from the convolution which will W/2 and H/2. Therefore, stride will act as down sampling. Another layer is Relu which is nonlinear activation layer. Its function is to bring non linearity into our network. Everything which is less than zero will be made zero i.e. all negatives will be made zero and all positives stay as it is. And that's brings non linearity into your data set and removes over fitting probability which makes it more adaptable to real case.

The processed filter that activated data goes into another layer which is called Pooling Layer. Pooling is another nonlinear layer. Pooling will get input from the previous layer i.e. convolution layer. Its main purpose is to gradationally reduce the spatial size of the processed image to reduce the amount of parameter and computation. The same patch will go down in the Pooling Layer. There are different aspects of Pooling Layer. There are different ways of doing pooling. The most common way is doing max pooling or average pooling. For example if there is 2*2 window. After applying max pooling on the 282 input data, will have the max value picked out. So this brings linearity and down sample. Now, why is down sampling important? Since CNN is known for its commute heavy and memory heavy issues. And with the help of Pooling, we are able to reduce the size and hence reduce the computation complexity and the memory complexity. Similar process keeps going on but a continuation of Convolution - Pooling - Convolution - Pooling takes place. Therefore, again the patch will go through bunch of activation filters in Convolution Layer and will get more information from that image. Then the down sampled it further. And once it done down sampled, at the end of the network, there are generally layers which are Fully Connected layers. As the term implies, fully connected Layer are the layers where each and every node is connected to the next node in the coefficients. The objective of fully connected layer is to identify or detect the final output categories. So it's a heavy data driven load where there are lot of coefficients which are loaded to support each and every node in the Pooling data. Hence, there will be multiple sets of output from the Pooling Layer and it will collectively draw top 3, top 5 best cases for the object into the consideration.

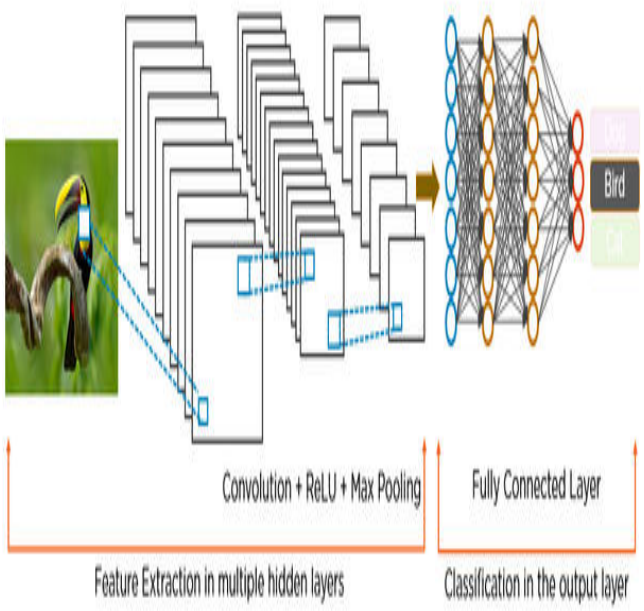

Fig 2 : Architecture of CNN

CNN can be into two phases i.e. training phase and inference phase. Training phase is the more involved phases where we are taking lot of data as input and going through feedback pattern and creating the filter data sets which are then used for inference phase.

3.2 RECURRENT NEURAL NETWORK (RNN)

Let's first understand the problem with our traditional Feed Forward Neural Network. Feed Forward Neural Networks are simply not good enough for time series data because Feed Forward Neural Network needs fixed size input and they give us fixed output. Outputs are independent to previous outputs. Therefore feed forward cannot be used for predicting next word in a sentence. They do not capture the sequences or time series information. They don't even account for memory.

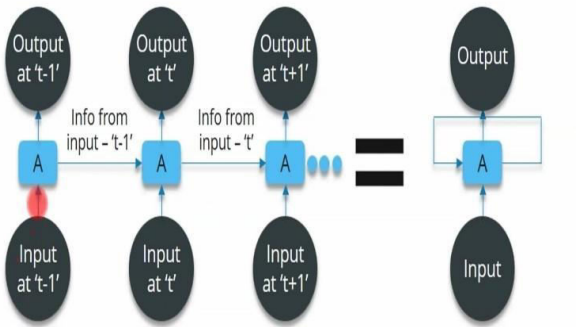

Fig 3 : Architecture of RNN

We have input at 't-1' will feed it to our network then we will get output at 't-1'. Then at the next time that is, at time 't' we have input at time 't' that will be given to a network along with the information from the previous time i.e. 't-1t and this will help us to get the output at 't'. Similarly as output for 't+1' we have two inputs, one is new input that we give and another is the information coming from the previous time i.e. 't' in order to get the output at time 't+1' and so on. There is a loop where the information from the previous timestamp is flowing and this is how we can overcome the problem faced in Feed Forward Neural Network.

Now let's understand what is RNN? The Recurrent Neural Network captures information about sequences/ time series data. They can take variable size input and gives variable size output. A real life example where RNN works is Gmail. Gmail has Recurrent Neural Network embedded into it. Therefore, when you type few words, it will auto complete it which saves your time. Another example of RNN is Google Translator where you can translate a sentence into one or another language easily.

4. IMAGE PREPROCESSING

An Image is an input X to our model. Any input image to a model will be given in the form of a vector. We need to convert every image into a fixed sized vector. Hence we use InceptionV3 model. Here, we do not want to classify the image but get an informative vector of fixed size from an image. This process is called automatic feature engineering.

5. DATA PREPROCESSING

Let's first understand why pre-processing is needed? Data in its original form will always be contaminated with different kinds of impurities or inconsistence. So it is our job that before we input into other system, we need to first purify it in proper format. Data will always be contaminated by incompleteness (lacking attributes), noise (deviate from actual result) and inconsistency (discrepancies in code). When we deal with words or sentence, we perform some basic data cleaning like lower casing all the words (else "world" and "World" would be considered as two separate words), removing punctuation marks as well as special characters like &,*,$ etc and lastly eliminating words which contain numbers like fish675etc. Captions are the end output of our model which we want to predict. Captions would be the target variables (Y) at the time of training period. But the predicting entire caption of given image does not take place at once. Therefore, captions are going to be predicted word by word.

6. VOCABULARY

The vocabulary was found out uniquely from the dictionaries that were created to store cleaned captions. Size of which came out to be 8424. Also the total number of words in the vocabulary was 373837. Moreover, the threshold value was set to be 10, the words below which were discarded as they did not occur even 10 times. And finally the complete length of this vocabulary was 1845. Main purpose of this was to find out the words in the captions, their frequency and the size of the total bag of words. The creation of vocabulary is meant for making the model robust to outliers and make sure that the model does not make mistakes.

7. WORD EMBEDDING

The two dictionaries namely 'word_to_idx' and 'idx_to_word' had been created that were used to assign an index to each word present in the captions. And vice-versa to ensure that the same word generates upon checking the index. Also, 'startseq' and endseq' had been added to this process so that the complete size of the vocabulary comes out to be 1848. As we add 0s also. (Originally it should be 1847 as two more words only have been assigned the indexes). The length of the largest caption generated out of this vocabulary was 35. These dictionaries were then used further in the data generator which is used because the dataset is huge and we need to perform batch processing on our captions.

```
[ ] # vocab_size is total vocabulary len +1 because we will append 0's as well.

    vocab_size = len(idx_to_word)+1
    print(vocab_size)

    1848
```

```
    all_captions_len = []

    for key in train_descriptions.keys():
        for cap in train_descriptions[key]:
            all_captions_len.append(len(cap.split()))

    max_len = max(all_captions_len)
    print(max_len)

    35
```

Fig 4 : Word Embedding

8. FEATURE EXTRACTION

After the CNN part where features were extracted, RNN(Recurrent neural network) plays the crucial role which is used to decode the entire images that generates sentences. The model was trained for 30 epochs with the learning rate of 0.001 and 3 pictures per batch. However, after 20 epochs, the learning rate was reduced from 0.001 to 0.0001 and the model was trained on 6 pictures per batch.

This usually makes sense because during the later stages of training, since the model is moving towards convergence, we must lower the learning rate so that we take smaller steps towards the minima. Also increasing the batch size over time helps your gradient updates to be more powerful. The terminology such as epoch, learning rate, batch are known as hyperparameters.

Hyperparameters are parameters whose values are set before learning process. By contrast, values of other parameters are derived via training. Different model training algorithm require different parameters.

Epochs: It is defined as a single pass through your entire training set. Total epoch means total cycles.

Initial learning rate: It is defined as the amount that weights and are updated during training is referred as step size/learning rate

Batch: It is defined as the process in which training samples are used to compute loss function. It is distinct from 'Online' and 'Mini-batch' learning

9.SENTENCE GENERATION ARCHITECTURE

Finally, for the image we are passing as an input one function named predict_caption is used which will concert the image into word and word into image. Basically it will train the model for generating a captions. And after this process is done our model will be able to convert it into final captions.

We will generate the caption iteratively, one word at a time as follows:

Iteration 1: Input: Image vector + "startseq" (as partial caption): Expected Output word: "the"

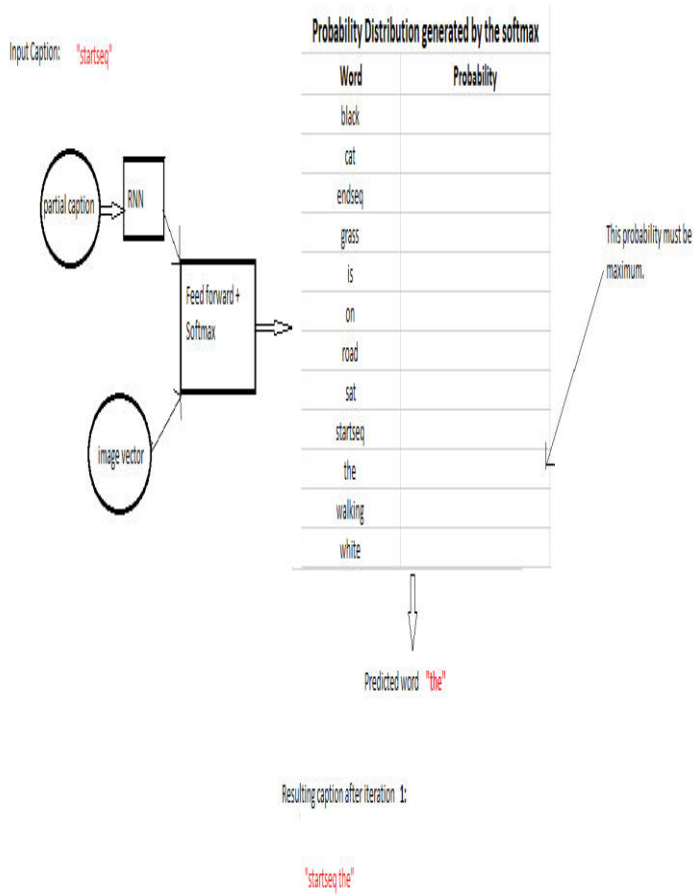

Fig 5 : Sentence Generation

Likewise our bot can generate a sentence by completing complete eight iterations.

10. MODEL ARCHITECTURE

Since the image consists of two parts, an image vector and a partial captions .Sequential model cannot be used. So for that we create custom. Initially features are extracted through CNN. Secondly, RNN will used to decode the entire images after CNN process is done. By doing this it will no get changed during the backpropagation. These partial captions are given to RNN so that it can understand and create a sequence of it. Lastly the backpropagation algorithm will leads to update the weights of the model and the model will learn to output a word. And it will fetch the correct word for the particular caption. And image vector is passed through Feed Forward ending with softmax this will help us predictwhat is the next caption according to the image we have given. So now let us take an overview of this diagram below:

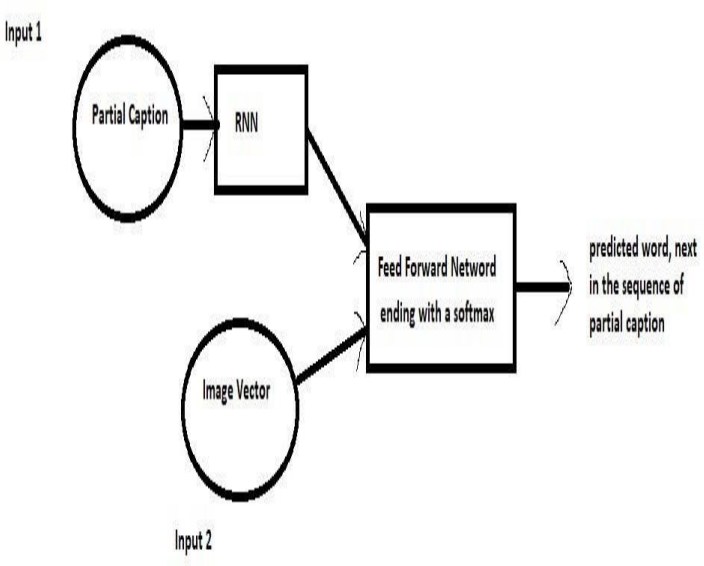

Fig 6 : Model Architecture

So we used LSTM(Long Short Term Memory) is nothing but a Special type of RNN (Recurrent Neural Network)to process the sequence input that is partial captions. As we had created an embedding matrix from a pre-trained Glove model which need to be include before starting the training. However, we need to freeze pre-trained embedding layer (trainable = False), before training the model.

```
model.layers[2].set_weights([embedding_matrix]
)
model.layers[2].trainable = False
```

Finally model is compiled using the adam optimizer:

```
model.compile(loss='categorical_crossentropy', optimizer='adam')
```

11. CONCLUSION AND RESULT

Here are some of the images for which our model has generated the caption.

Fig 7 : Output 1

Fig 8 : Output 2

Fig 9 : Output 3

Fig 10 : Output 4

12. FUTURE SCOPE

The requirement of the society and the progress in technology today has resulted in the development of Machine Learning and we can solve the difficulty of blind person by designing a Caption Bot that will help him/her carry out his/her task easily. We can create a product for the blind person which will guide them travelling without the support of anyone else. We can do this by first converting the image into text and then the text to speech. Elimination of the walking stick is a major achievement for the blind people. Therefore, the model will aimed at the well-being of the blind person. Also we can introduce Devanagari Script for the Speech Module so that people can use it easily.

ACKNOWLEGEMT

We would like to express our sincere gratitude and indebtedness to our guide Prof HarshwardhanKharpate for his valuable suggestion, constant supervision, timely guidance, keen interest and encouragement throughout our project and research paper. It would not be possible for us to complete the same without his sincere and affectionate help.

We would also like to express our gratitude and thanks to industry person Mr. Abhishek Pathak for giving us time as well as attention despite of his professional commitments.

Finally, we would like to thanks Dr. Bhushan Joshi for providing us amenities required in the successful completion.

REFERENCES

[1] Turbo learning for Caption Bot and Drawing Bot. https://papers.nips.cc/paper/7881-turbo-learning-for-captionbot-and-drawingbot.pdf .

[2] Turbo learning for Caption Bot and Drawing Bot. https://www.researchgate.net/publication/325283055_Turbo_Learning_for_Captionbot_and_Drawingbot

[3] Caption Bot, a picture worth a thousand words – Microsoft. https://www.captionbot.ai/

[4] Turbo Learning for Caption Bot and DrawingBot.https://deeplearn.org/arxiv/55789/turbo-learning-for-captionbot-and-drawingbot .

[5] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3128–3137.

[6] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Showand tell: A neural image caption generator.arXiv preprintarXiv:1411.4555, 2014.

[7] C. Matuszek*, N. FitzGerald*, L. Zettlemoyer, L. Bo, andD. Fox. A Joint Model of Language and Perception forGrounded Attribute Learning. InProc. of the 2012 Interna-tional Conference on Machine Learning, Edinburgh, Scotland, June 2012.

[8] M. Yatskar, L. Vanderwende, and L. Zettlemoyer. See noevil, say no evil: Description generation from densely la-beledimages.Lexical and Computational Semantics, 2014.

[9] A. Barbu, A. Bridge, Z. Burchill, D. Coroian, S. Dickin-son, S. Fidler, A. Michaux, S. Mussman, S. Narayanaswamy,D. Salvi, et al. Video in sentences out.arXiv preprintarXiv:1204.2742, 2012.

[10] K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D. M.Blei, and M. I. Jordan. Matching words and pictures.JMLR,2003.

[11] S. Gould, R. Fulton, and D. Koller. Decomposing a sceneinto geometric and semantically consistent regions. InCom-puter Vision, 2009 IEEE 12th International Conference on,pages 1–8. IEEE, 2009.

[12] https://cs.stanford.edu/people/karpathy/cvpr2015.pdf

[13] https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/