International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Image Caption Generator by using CNN and LSTM

Dr. S. Pasupathy

Associate Professor, Department of Computer Science and Engineering, Annamalai University

Abstract

In this article, we systematically analyze a deep neural networks-based image caption generation method. Image Captioning aims to automatically generate a sentence description for an image. Our article model will take an image as input and generate on English sentence as output, describing the contents of the image. It has attracted much research attention in cognitive computing in the recent years. The task is rather complex, as the concepts of both computer vision and natural language processing domains are combined together. We have developeda model using the concepts of a Convolutional Neural Network (CNN) and long Short-Term Memory (LSTM) model and build a working model of Image caption generator by implementing CNN and LSTM. After the caption generation phase, we use BLEU Scores to evaluate the efficiency of our model. Thus, our system helps the user to get descriptive caption for the given input image.

Keywords: Convolutional Neural Network (CNN) and long Short-Term Memory (LSTM), BLEU (BiLingual Evaluation Understudy)

INTRODUCTION

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 needs to find which objects are contained in the image and also need to be able to express their relationships in a natural language such as English. Recent work also achieves the presence of attention, which can store and report the information and relationship between some most salient features and clusters in the image. In our article, 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 to understand the world in images. In addition, we introduced an 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 dominates the whole data in the world.

AIM OF THE ARTICLE

Image caption generator is a task that involves computer vision and natural language processing concepts to recognize the context of an image and describe them in a natural language like English. In this article, we will be implementing the caption generator using CNN and LSTM. The image features will be extracted from exception which is a CNN model trained on the image net dataset and then we feed the features into the LSTM model which will be responsible for generating the image captions. The scope of our article is to learn the concepts of a CNN and LSTM model and build a working model of Image caption

generator by implementing CNN with LSTM.

LITERATURE REVIEW

[1] A Deep Learning Approach (2018) Lakshmi narasimhan Srinivasan, Dinesh Sreekanthan and A.L Amutha. In this research they have used the tensor flow backend for the keras framework to evaluate the model. Using selected evaluation metrics suitable for the type of the problem helped in knowing how accurately the model has predicted. In this paper they have conducted mathematical computations on the confusion matrix and analyzed theresult.

[2] Image description using visual dependency representations. D. Elliott and F. Keller. In this research the major challenges are recognizing theobjects in an image and their attributes are difficult computer visionproblems; while determining how the objects interact, which relationshipshold between them. Automatic image description presents challenges on anumber of levels. The authors used CNN to train the model in different levels (layers) to make the model perform well. [3] Image Caption Generator Using Deep Learning Technique. C. Amritkar and V. Jabade. In this paper the model is trained in such a way that if aninput image is given to the model. it generates captions which nearly describes the image. The accuracy of model and smoothness or command of language model learned from image descriptions are tested on different datasets. These experiments show that the model is frequently giving accurate descriptions for an input image. [4] Deep Learning based Automatic Image Caption Generation. V. Kesavan, V. Muley and M. Kolhekar. The paper aims at generating automated captions by learning the contents of the image. In this paper, they systematicallyanalyze different deep neural network-based image caption generation approaches and pre-trained models to conclude on the most efficient model with fine-tuning. They analyzed models containing both with and without attention concepts to optimize the caption generating ability of themodel. All the models are trained on the same dataset for concrete comparison. [5] Detection and Recognition of Objects in Image Caption Generator System: A Deep Learning Approach, N. K. Kumar, D. Vigneswari, A. Mohan, K. Laxman and J. Yuvaraj. The aim of this paper is to detect, recognize and generate worthwhile captions for a given image using deep learning.Regional Object Detector (ROD) is used for the detection, recognition and generating captions. The proposed method focuses on deep learning to further improve upon the existing image caption generator system. Experiments are conducted on the Flickr 8k dataset using python languageto demonstrate the proposed method.

MODULE DESCRIPTION

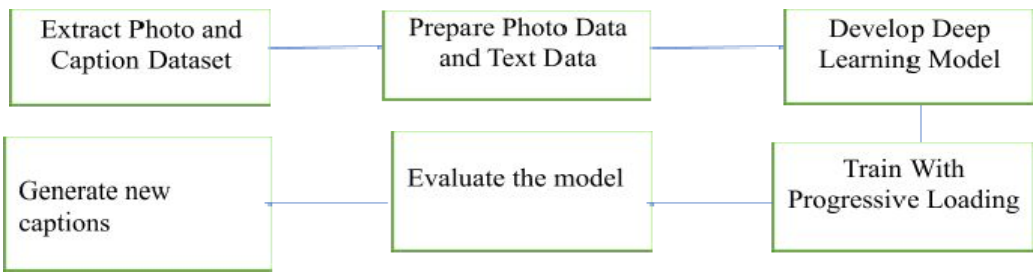

Figure 1: Architecture Diagram

Design Phase

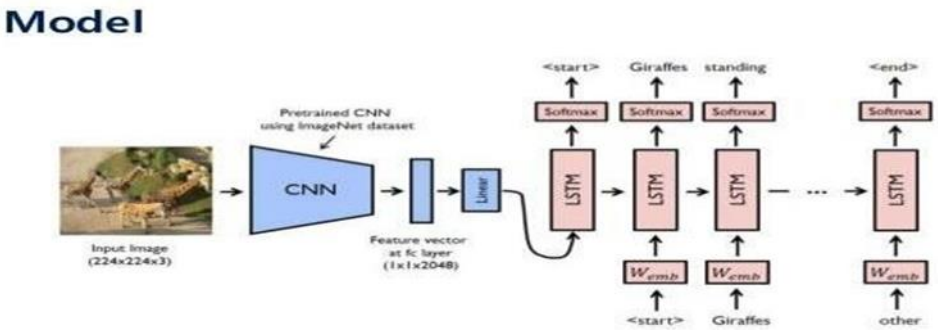

Figure 2 Work flow model.

So, to make our image caption generator model, we will be merging thesearchitectures. It is also called a CNN-RNN model. CNN is used for extracting features from the image. We will use thepre-trained model exception. LSTM will use the information from CNN to help generate a description of the image

Data Flow Diagram

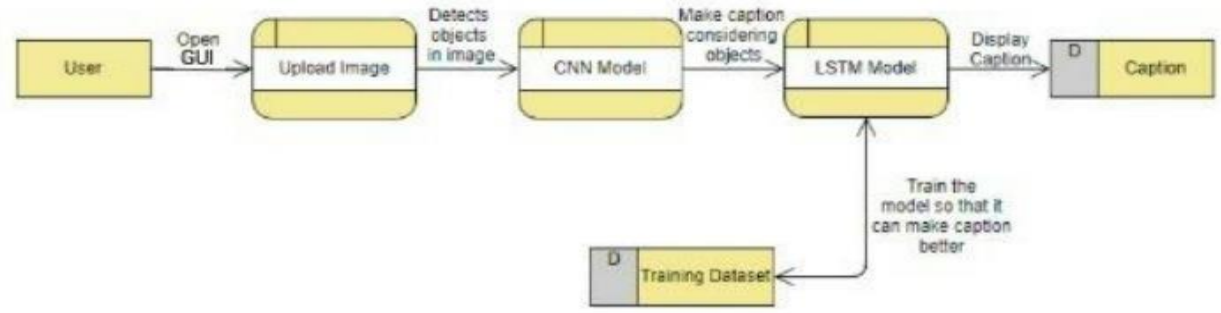

Figure 3: Data Flow Diagram

IMPLEMENTATION AND TESTING

ARTICLE FILE STRUCTURE

Flicker8k_Dataset – Dataset folder which contains 8091 images. Flickr_8k_text – Dataset folder which contains text files. The below files will be created by us while making the article. Models – It will contain our trained models. Descriptions.txt – This text file contains all image names and theircaptions after preprocessing. Features.p – Pickle object that contains an image and their featurevector extracted from the exception pre-trained CNN model. Tokenizer.p – Contains tokens mapped with an index value. Model.png – Visual representation of dimensions of our article. Testing_caption_generator.py – Python file for generating a caption of any image. Training_caption_generator.ipynb – Jupyter notebook in which wetrain and build our image caption generator.

EXTRACTING THE FEATURE VECTOR FROM ALL IMAGES

This technique is also called transfer learning, we don't have to do everything on our own, we use the pre-trained model that has been already trained on large datasets and extract the features from these models

and use them for our tasks. Weare using the exception model which has been trained on an image net dataset that had 1000 different classes to classify. We can directly import this model from the keras.applications . Make sure you are connected to the internet as the weights get automatically downloaded. Since the Xception model was originally built for imagenet, we will make little changes for integrating with our model. One thing to notice is that the Xception model takes 299*299*3 image size as input. We will remove the last classification layer and get the 2048 feature vector.

model = Xception(include_top=False, pooling='avg')
The function extract_features() will extract features for all images and we will map image names with their respective feature array. Then we will dump the features dictionary into a "features.p" pickle file.

LOADING DATASET FOR TRAINING THE MODEL

In our Flickr_8k_test folder, we have Flickr_8k.trainImages.txt file that contains a list of 6000 image names that we will use for training. For loading the training dataset, we need more functions: load_photos(filename) – This will load the text file in a string and will returnthe list of image names. load_clean_descriptions(filename, photos) – This function will create a dictionary that contains captions for each photo from the list of photos. We also append the <start> and <end> identifier for each caption. We need this sothat our LSTM model can identify the starting and ending of the caption. load_features(photos) – This function will give us the dictionary for image names and their feature vector which we have previously extracted from the Xception model.

TOKENIZING THE VOCABULARY

Computers don't understand English words, for computers, we will have to represent them with numbers. So, we will map each word of the vocabulary with a unique index value. Keras library provides us with the tokenizer function that we will use to create tokens from our vocabulary and save them to a **"tokenizer.p"** pickle file. Our vocabulary contains 7577 words. We calculate the maximum length of the descriptions. This is important for deciding the model structure parameters. Max_length of description is 32.

```
#calculate maximum length of descriptions
def max_length(descriptions):
    desc_list = dict_to_list(descriptions)
    return max(len(d.split()) for d in desc_list)

max_length = max_length(descriptions)
max_length
```

Create Data Generator

Let us first see how the input and output of our model will look like. To make this task into a supervised learning task, we have to provide input and output tothe model for training. We have to train our model on 6000 images and each image will contain a 2048 length feature vector and the caption is also represented as numbers. This amount of data for 6000 images is not possible tohold into memory so we will be using a generator method that will yield batches. The generator will yield the input and output sequence.

For example:
The input to our model is [x1, x2] and the output will be y, where x1 is the 2048 feature vector of that image, x2 is the input text sequence and y is the output text sequence that the model has to predict.

TRAINING THE MODEL

To train the model, we will be using the 6000 training images by generatingthe input and output sequences in batches and fitting them to the model using model.fit_generator() method.

TESTING THE MODEL

The model has been trained, now, we will make a separate file testing_caption_generator.py which will load the model and generate predictions. The predictions contain the max length of index values so we will use the same tokenizer.p pickle file to get the words from their index values.

RESULTS AND DISCUSSIONS

INPUT AND OUTPUT

Figure 4: Output Image

Figure: 5 Output Image

The system can generate sentences that are semantically correctaccording to the image. We also proposed a simplified version of GRU that has less parameters and achieves comparable results with the input image.

The strength of the method is on its end-to-end learning framework. The weakness is that it requires a large number of humans labeled data whichis very expensive in practice. Also, the current method still has considerable errors in both object detection and sentence generation.

CONCLUSION

Automatically image captioning is far from mature and there are a lot of ongoing research articles aiming for more accurate image feature extraction and semantically better sentence generation. We successfully completed what we mentioned in the article proposal, but used a smaller dataset (Flickr8k) due to limited computational power. The developed model is capable to autonomously view an image and generate a reasonable description in natural language with reasonable accuracy and naturalness. Further extension of the present model can bein regard to increasing additional CNN layers or increasing/implementing pre-training, which could improve the accuracy of the predictions. We analyzed and modified an image captioning method. To understand the method deeply, we decomposed the method to CNN, RNN, and sentence generation. For each part, we modified or replaced the component to see the influence on the final result. Another potential improvement is by training on a combination of Flickr8k, Flickr30k, and MSCOCO. In general, the more diverse training dataset the network has seen, the more accurate the output will be. We allagree this article ignites our interest in application of Machine Learning knowledge in Computer Vision and expects to explore more in the future.

References

1. P. Shah, V. Bakrola and S. Pati, "Image captioning using deep neural architectures," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, 2017, pp. 1-4, doi: 10.1109/ICIIECS.2017.8276124.

2. S. Han and H. Choi, "Domain-Specific Image Caption Generator with Semantic Ontology," 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), Busan, Korea (South), 2020, pp. 526- 530, doi:10.1109/BigComp48618.2020.00-12.

3. C. Amritkar and V. Jabade, "Image Caption Generation Using Deep Learning Technique," *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, Pune, India, 2018, pp. 1-4, doi: 10.1109/ICCUBEA.2018.8697360.

4. N. K. Kumar, D. Vigneswari, A. Mohan, K. Laxman and J. Yuvaraj, "Detection and Recognition of Objects in Image Caption Generator System: A Deep Learning Approach," *2019 5th International Conferenceon Advanced Computing & Communication Systems (ICACCS)*, Coimbatore, India, 2019, pp. 107-109,doi:10.1109/ICACCS.2019.8728516.

5. V. Kesavan, V. Muley and M. Kolhekar, "Deep Learning based Automatic Image Caption Generation," 2019 Global Conference for Advancement in Technology (GCAT), BENGALURU, India, 2019, pp. 1-6, doi: 10.1109/GCAT47503.2019.8978293.