Nava Jyoti, International Journal of Multi-Disciplinary
Research Volume 4, Issue 1, August 2019

Image Captioning Using Deep Learning

Amrutha A[1], Indumathi S[2], Irene Getzi S[3]

[1]Student, Department of MCA, Jyoti Nivas College, Bangalore, India
[2] Student, Department of MCA, Jyoti Nivas College, Bangalore, India
[3] Faculty, Department of MCA, Jyoti Nivas College, Bangalore, India

ABSTRACT

Image Captioning refers to automatic generation of textual description by analyzing the content of an image. Automated caption generation of an image uses computer vision and natural language processing. Deep learning based techniques provide better results in analyzing the image contents. This article presents a multilayer neural network model that automatically learns and predicts the content of images. Our model consists of two sub-models, namely Convolutional Neural Network (CNN) that extracts pixel level features from the image, identify objects and recognize actions. The other model Recurrent Neural Network (RNN) helps in describing the objects and actions in a single sentence.

Keywords: Image Captioning, Deep Learning, Machine Learning, Convolutional Neural Network and Recurrent Neural Network

INTRODUCTION

Artificial intelligence techniques such as machine learning and deep neural networks gain momentum recently and has made considerable advancement in image processing. It provides better results in image classification and object detection. But, image understanding tasks is complex than image classification and object detection applications. The real challenge is to understand the whole image scenario by extracting the complete details of individual objects and their associated relationship from image and then description of the image. This problem involves two major artificial intelligence fields namely computer vision and natural language processing. Computer vision enables the system to identify, process and understand the images. Natural Language Processing (NLP) helps in analyzing, understanding and generating the languages that humans can understand.

A CNN is a feed-forward based artificial neural network that can be used for analyzing visual images and identify objects. It compares two images by the features it has extracted. RNN on the other hand, is designed to model sequences of data, which can be utilized to process words or sentences. For generating sentences a Long Short-Term Memory (LSTM) based Recurrent Neural Network is used. The LSTM models is best suitable for temporal sequences of varying lengths and trained using back propagation method. It replaces traditional artificial neuron with a memory cell containing long and short term nonlinear capabilities.

Visual Geometry Group (VGG) is a pre-trained deep neural network model used to interpret the content of the photos. This model is very powerful and flexible and produced better results in image classification.

In the last few years, convolutional models have changed the computer vision landscape drastically. Several methods are proposed for the underlying problem of caption generation

from the images. Most of them aim at providing solutions for visually-impaired. The methods differ in the underlying architectures used for CNN and for NLP.

In this article, a CNN based model that uses VGG-16 network is utilized to extract features. By replacing the last stage of CNN, with three state-of-the-art architectures such as RNN and LSTM, we find the VGG-16 performs best according to the Bilingual Evaluation Understudy (BLEU) score.

PROPOSED WORK

The aim of this work is to build a system that can analyze the content of the digital photos and generate a caption in English Language. The system could identify objects, actions or events in an image, understand the underlying relationship between the objects and create a description. The CNN, which acts as an encoder extracts the deep features out of the image and RNN acts as the decoder and translates the features and objects to a sentence.
Visual Geometry Group (VGG) is a pre-trained model used to interpret the content of the photos. The last layer which is used to predict a classification for a photo is replaced by RNN with LSTM. The model is trained in such a way that it maximize the likelihood of the caption generation when a training image is given. The workflow of VGC16 is described as follows:

- Initially load each photo, prepare it for VGG by resizing the image to 224×224 pixels, and collect the predicted features from the VGG model.

- The description file contains tokenized texts. It is cleaned by converting all words to lowercase, removing all punctuation, removing all words that are one character or less in length and remove all words with numbers in them. Once cleaned, the descriptions are saved to a new file.

- Train the data on all of the photos and captions in the training dataset.

- Use the corpus BLEU score to find the accuracy of each model.

- Finally evaluate a model by generate captions for entirely new photographs in the test dataset.

IMPLEMENTATION AND RESULTS

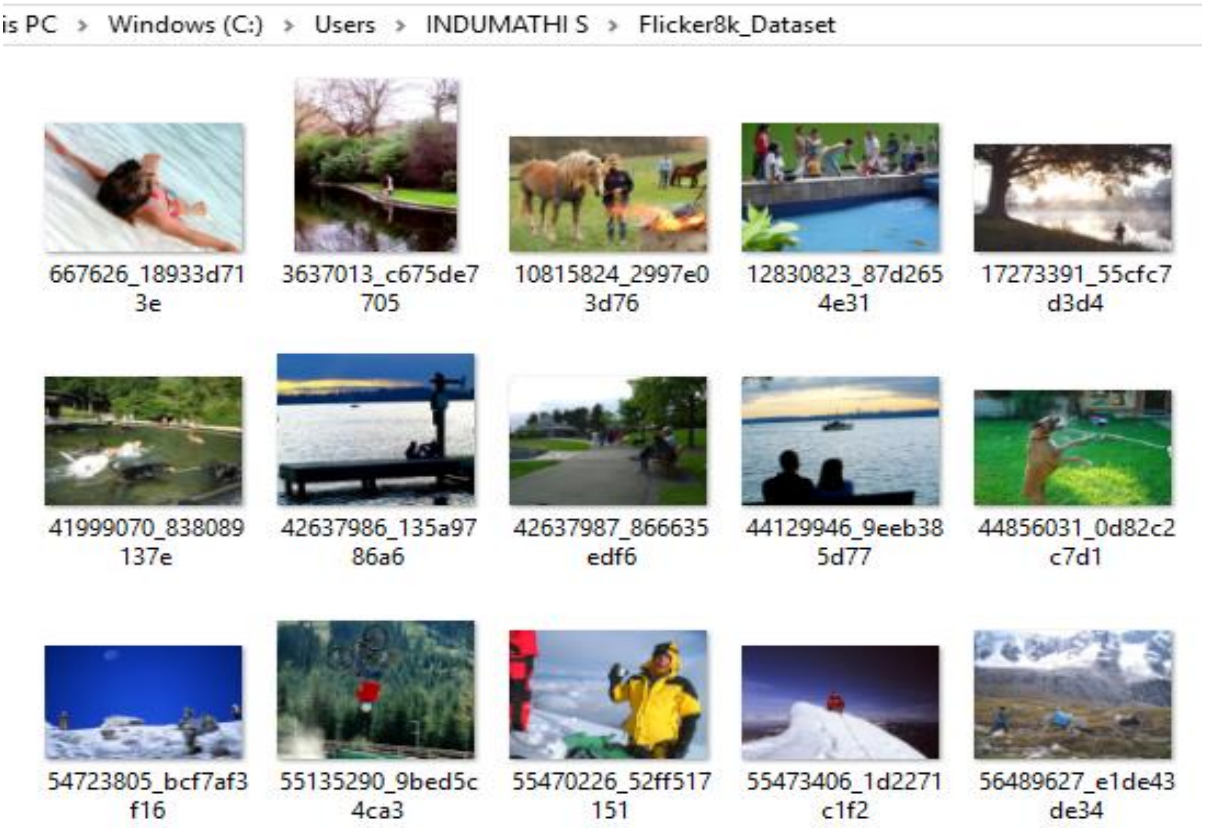
Environmental Setup: Python SciPy environment installed ideally with Python 3.6.3, Keras with TensorFlow, and other libraries like Scikit-learn, Pandas, NumPy, and Matplotlib is used. Minimum of 8GB RAM is required to train the machine with large volume of photos. VGG16 algorithm is used for feature extraction.

Dataset: For training and testing Flickr8k dataset have been used. Flilckr8K contains 8,000 images and descriptions of the image salient entities and events are contained in a text file. Each image is provided with five different captions. The images were chosen from six different Flickr groups, to depict a variety of scenes and situations. The following are the details of the data set used along with the sample images and text descriptions.

- **Flicker8k_Dataset:** Contains 8092 natural images in jpeg format.
- **Flickr8k_text:** Text files that provides descriptions for the images.

The dataset has a training samples of 6000 images, 1000 for validation and 1000 images for testing. The following images depicts the sample images and captions from the dataset.

Flicker8k_Dataset



Flickr8k_text

Flickr8k.lemma.token.txt

```
1305564994_00513f9a5b.jpg#0 A man in street racer armor be examine the tire of another racer 's motorbike
1305564994_00513f9a5b.jpg#1 Two racer drive a white bike down a road .
1305564994_00513f9a5b.jpg#2 Two motorist be ride along on their vehicle that be oddly design and color .
1305564994_00513f9a5b.jpg#3 Two person be in a small race car drive by a green hill .
1305564994_00513f9a5b.jpg#4 Two person in race uniform in a street car .
1351764581_4d4fb1b40f.jpg#0 A firefighter extinguish a fire under the hood of a car .
1351764581_4d4fb1b40f.jpg#1 a fireman spray water into the hood of small white car on a jack
1351764581_4d4fb1b40f.jpg#2 A fireman spray inside the open hood of small white car , on a jack .
1351764581_4d4fb1b40f.jpg#3 A fireman use a firehose on a car engine that be up on a carjack .
1351764581_4d4fb1b40f.jpg#4 Firefighter use water to extinguish a car that be on fire .
1358089136_976e3d2e30.jpg#0 A boy sand surf down a hill
1358089136_976e3d2e30.jpg#1 A man be attempt to surf down a hill make of sand on a sunny day .
1358089136_976e3d2e30.jpg#2 A man be slide down a huge sand dune on a sunny day .
1358089136_976e3d2e30.jpg#3 A man be surf down a hill of sand .
1358089136_976e3d2e30.jpg#4 A young man in short and t-shirt be snowboard under a bright blue sky .
```

Implementation Steps:

Step 1 – Extract Features from Images

VGG16 is a pre-trained model used to interpret the content of the photos. Keras provides this pre-trained model in Python library. The last layer of VGG that is used for classification of a photo is removed. The model is used to extract the images features and are saved in a file. The extracted feature sample is as shown below.

features.txt

```
{'1000268201_693b08cb0e': array([[2.5076475, 0.       , 0.       , ..., 0.       , 0.       ,
        0.       ]], dtype=float32), '1001773457_577c3a7d70': array([[0.       , 0.       , 0.4941089, ..., 0.       , 0.       ,
        0.       ]], dtype=float32), '1002674143_1b742ab4b8': array([[1.4937081 , 0.        , 0.53568316, ..., 2.315415  , 3.7418408 ,
        0.        ]], dtype=float32), '1003163366_44323f5815': array([[0., 0., 0., ..., 0., 0., 0.]], dtype=float32), '1007129816_e79441
       array([[0.        , 0.09227684, 0.        , ..., 0.        , 0.        ,
       0.06529188]], dtype=float32), '1007320043_627395c3d8': array([[0.       , 0.       , 0.       , ..., 0.       , 3.3386438,
        0.       ]], dtype=float32), '1009434119_febe49276a': array([[2.0962925 , 2.1193182 , 3.5624356 , ..., 0.64264125, 2.7146504 ,
        0.        ]], dtype=float32), '1012212859_01547e3f17': array([[0.        , 0.        , 0.9873712 , ..., 0.        , 1.4932494 ,
       0.86128545]], dtype=float32), '101511866l_98073541lb': array([[1.4410403 , 0.28292155, 0.        , ..., 0.        , 2.4591994 ,
        0.        ]], dtype=float32), '1015584366_dfcec3c85a': array([[0.7002893, 0.       , 1.256156 , ..., 0.       , 0.       ,
        0.       ]], dtype=float32), '101654506_8eb26cfb60': array([[0.       , 0.       , 0.7495302, ..., 0.       , 2.2005944,
```

Step 2 – Clean the Text Descriptions

The dataset contains multiple descriptions for each photo and the text of the descriptions requires some minimal cleaning. The file containing the descriptions of all the images is loaded and cleaned to reduce the size of the vocabulary of words. The cleaning process includes:

- To convert all words to lowercase
- To remove all punctuation
- To remove all words that are one character or less in length
- To remove all words with numbers in them.

The sample of the cleaned data is as shown in the following image.

descriptions.txt

```
1000268201_693b08cb0e child in pink dress is climbing up set of stairs in an entry way
1000268201_693b08cb0e girl going into wooden building
1000268201_693b08cb0e little girl climbing into wooden playhouse
1000268201_693b08cb0e little girl climbing the stairs to her playhouse
1000268201_693b08cb0e little girl in pink dress going into wooden cabin
1001773457_577c3a7d70 black dog and spotted dog are fighting
1001773457_577c3a7d70 black dog and tricolored dog playing with each other on the road
1001773457_577c3a7d70 black dog and white dog with brown spots are staring at each other in the street
1001773457_577c3a7d70 two dogs of different breeds looking at each other on the road
1001773457_577c3a7d70 two dogs on pavement moving toward each other
1002674143_1b742ab4b8 little girl covered in paint sits in front of painted rainbow with her hands in bowl
1002674143_1b742ab4b8 little girl is sitting in front of large painted rainbow
1002674143_1b742ab4b8 small girl in the grass plays with fingerpaints in front of white canvas with rainbow on it
1002674143_1b742ab4b8 there is girl with pigtails sitting in front of rainbow painting
1002674143_1b742ab4b8 young girl with pigtails painting outside in the grass
```

Step 3 - Train the data with Progressive Loading

The prepared photos and texts are progressively loaded so that it can be used to fit the model. The model is trained using all the data from the training samples and the model is saved after each training epoch. Each saved model after training is loaded again to evaluate and find the one with the lowest loss that can be used for prediction.

Step 4 - Evaluate Model

Once the model is fit, the skill of its predictions is evaluated on the holdout test dataset. We will evaluate a model by generating descriptions for all photos in the test dataset and evaluating those predictions with a standard cost function. Then evaluate a trained model against a given dataset of photo descriptions and photo features. The actual and predicted descriptions are

collected and evaluated collectively using the corpus BLEU score that summarizes how close the generated text is to the expected text.

Step 5 - Generate New Captions

Almost everything we need to generate captions for entirely new photographs is in the model file. We also need the Tokenizer for encoding generated words for the model while generating a sequence, and the maximum length of input sequences, used when we defined the model. We can hard code the maximum sequence length. The encoding of text and the tokenizer can be saved into a text file and can be loaded without using the Flickr8K image dataset.

The sample output obtained are tabulated as below. The first sample depicts the training phase and the rest of the samples depicts the few test cases. As seen from the results, our algorithm is able to predict the scene and generate the caption with good accuracy. The model is also verified quantitatively by using Bilingual Evaluation Understudy (BLEU) score. The algorithm yields a BLEU score of 59, as compared to human performance around 69.

```
(base) C:\Users\INDUMATHI S>python p6.py
Using TensorFlow backend.
2018-09-25 21:35:06.256809: I tensorflow/core/platform/cpu_feature_
s TensorFlow binary was not compiled to use: AVX AVX2
Output -
model 12 : startseq dog is running through the grass endseq
model 13 : startseq two dogs are playing with ball in the grass end
model 14 : startseq two dogs are playing in the snow endseq
```

```
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-11-15 11:15:30.307449: I T:\src\github\tensorflow\tensorflow\cor
cpu_feature_guard.cc:141] Your CPU supports instructions that this Te
nary was not compiled to use: AVX2

Output : startseq man is standing on the beach with his hands endseq

(base) C:\Users\MCA>
```

```
  from ._conv import register_converters as _register_converte
Using TensorFlow backend.
2018-11-15 11:23:59.504449: I T:\src\github\tensorflow\tensorf
cpu_feature_guard.cc:141] Your CPU supports instructions that t
nary was not compiled to use: AVX2

Output : startseq two people are standing on the beach endseq
```

```
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-11-15 11:27:28.795449: I T:\src\github\tensorflow\tensorflow\core\p
cpu_feature_guard.cc:141] Your CPU supports instructions that this Tenso
nary was not compiled to use: AVX2

Output : startseq man in red shirt is riding bike down dirt road endseq
```


```
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-11-15 11:29:05.454449: I T:\src\github\tensorflow\tensorflow\
cpu_feature_guard.cc:141] Your CPU supports instructions that this
nary was not compiled to use: AVX2

Output : startseq baseball player is running in the field endseq
```

CONCLUSION

This work presents a two-fold approach using state-of-art neural networks to learn the content of the images and generate appropriate captions. The VGG 16 architecture of CNN is used to extract the information of objects and their spatial locations in an image. RNN with LSTM is used to generate a description sentence. Each word of the description is automatically aligned to different objects in the input image when it is generated.

REFERENCES

[1] Yash Badani, et al, Automated Neural Image Caption Generator for Visually Impaired People, IOSR Journal of Engineering (IOSRJEN), Volume 10.
[2] Moses Soh, Learning CNN-LSTM Architectures for Image Caption Generation, 2016.
[3] Zhongliang Yang, et al, Image Captioning with Object Detection and Localization, International Conference on Image and Graphics, 2017.
[4] Ankit Gupta, et al, CS771 Project Image Captioning.
[5] Haley MacLeod, et al, Understanding Blind People's Experiences with Computer-Generated Captions of Social Media Images, In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, 2017.
[6] Christopher Elamri, et al, Automated Neural Image Caption Generator for Visually Impaired People.
[7] Chenyou Fan, et al, DeepDiary: Automatically Captioning Lifelogging Image Streams, European Conference on Computer Vision, 2016.
[8] Ingrid Hrga, et al, Deep Image Captioning: Models, Data and Evaluation.
[9] Jianhui Chen, et al, Image Caption Generator Based on Deep Neural Networks.
[10] Oriol Vinyals, et al, Show and Tell: A Neural Image Caption Generator, In Proceedings of the IEEE Conference on computer vision and pattern recognition, 2015.
[11] Raffaella Bernardi, et al, Automatic Description Generation from Images: A Survey of Models, Datasets, and Evaluation Measures, 2016.
[12] Akanksha P. Deshmukh, et al, A Survey on Vision Based Approaches for Image Description, 2003.