Automated Image Captioning Using Deep Learning

Dr. Savita Sangam[1], Abhijeet Sawant[2], Abhishek Malap[3], Deepak Yadav[4]

[1]Dr. Savita Sangam, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India
[2] Student Abhijeet Sawant, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India
[3]Student Abhishek Malap, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India
[4] Student Deepak Yadav, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India

---***---

Abstract – *Computer vision has become ubiquitous in our society, with applications in several fields. In this project, we focus on one of the visual recognition facets of computer vision, i.e image captioning. We analyze Image Cationing using deep neural networks based caption generation method. Image as input and the method can output short english sentence describes the content in the image. We used RNN, CNN, LSTM for generating suitable captions according to image. Flicker8k dataset we used for training and testing purpose.*

***Key Words*:** **Deep learning, CNN, RNN, LSTM, Dataset**

1.INTRODUCTION

Artificial Intelligence (AI) is now at the heart of innovation economy and thus the base for this project is also the same. In the recent past a field of AI namely Deep Learning has turned a lot of heads due to its impressive results in terms of accuracy when compared to the already existing Machine learning algorithms. The task of being able to generate a meaningful sentence from an image is a difficult task but can have great impact, for instance helping the visually impaired to have a better understanding of images.

The task of image captioning is significantly harder than that of image classification, which has been the main focus in the computer vision community. A description for an image must capture the relationship between the objects in the image. In addition to the visual understanding of the image, the above semantic knowledge has to be expressed in a natural language like English, which means that a language model is needed. The attempts made in the past have all been to stitch the two models together.

In the model proposed in the paper we try to combine this into a single model which consists of a Convolutional Neural Network (CNN) encoder which helps in creating image encodings. We use the VGG16 architecture proposed by _____ with some modifications. We could have used some of the recent and advanced classification architectures but that would have increased the training time significantly. These encoded images are then passed to a LSTM network which are a type of Recurrent Neural Network. The network architecture used for the LSTM network work in similar fashion as the ones used in machine translators. The input to the network is an image which is first converted in a 224*224 dimension. We use the Flickr8k dataset to train the model. The model outputs a generated caption based on the dictionary it forms from the tokens of caption in the training set.

1.1 Problem Definition

In day to day life we have seen lots of images on internet and almost everywhere like news, articles. Sometimes images having some short amount of description about it but out of them some images are just images and nothing extra as we are human we can figure out what's in it. So we are trying to build a model that will take input image from user and then machine gives a suitable caption. So due to this our model can analyse thousands of images and then it will be used for test purposes to know what images says. It is very helpful in Artificial Intelligence to recognize images and gives responses to request sends comes from users.

2. Literature Survey

Image caption generation is a core part of scene understanding, which is important because of its use in a variety of applications (eg. - image search, telling stories from albums, helping visually impaired people understand the web etc.). Over the years, many different image captioning approaches have been developed.

The architectures used by the winners of ILSVRC have contributed a lot to this field. One such architecture used by us was the VGG16 proposed by He et. al. in 2014 [2]. Apart from that the research in the tasks of machine translation have consistently helped in improving the state of the art performance in language generation.

In 2015, researchers at Microsoft's AI Lab used a pipeline approach to image captioning [3]. They used a CNN to generate high-level features for each potential object in the image. Then they used Multiple Instance Learning (MIL) to figure out which region best matches each word. The approach yielded 21.9% BLEU score on MSCOCO. After the pipeline approach, researchers at Google came up with the first end-to-end trainable model. They were inspired by the RNN model used in machine translation.

Vinyals et al. [1] replaced this encoder RNN with CNN features of the image as the CNN features are widely used in all computer vision tasks. They called this model as Neural

Image Caption(NIC). Following this, two researchers at Stanford modified the NIC. They used an approach that leverages datasets of images and their sentence descriptions to learn about the inter-modal correspondences between language and visual data. Their alignment model was based on a novel combination of Convolutional Neural Networks over image regions, bidirectional Recurrent Neural Networks over sentences, and a structured objective to align the two modalities through a multimodal embedding. They used the Flickr8K, Flickr30K and MSCOCO datasets and achieved state-of-the-art results in the same [4]. Their model was further modified by Jonathan et. al. [5] in 2015 when they proposed a dense captioning task in which each region of an image was detected and a set of descriptions generated. Another model which used a deep convolutional neural network (CNN) and two separate LSTM networks was proposed by Wang et. al. [6] in the year 2016.
One of the most recent work was inspired by the NIC model and was proposed by Xu et. al. in 2016 [7]. They were inspired by the advancements in the field of machine translation and object detection and introduced an attention based model that automatically learned to describe the content of images.
In the past few years, progress has been made not only in image captioning models but also in various evaluation metrics. The accuracy metric used by us was the BLEU score [8]. BLEU - which was a standard evaluation metric adopted by many of the groups - is slowly being replaced by CIDEr proposed by Vedantam et. al. in 2015 [9].

3. Develop Deep Learning Model

This section is divided into the following parts:

1. Loading Data.
2. Defining the Model.
3. Fitting the Model.
4. Complete Example.

3. 1 Loading Data

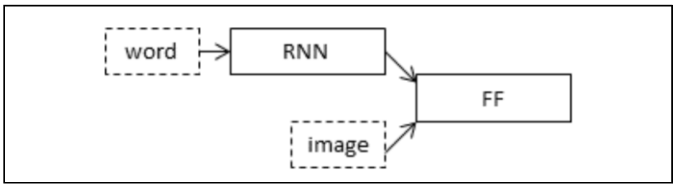
First, we must load the prepared photo and text data so that we can use it to fit the model.

We are going to train the data on all of the photos and captions in the training dataset. While training, we are going to monitor the performance of the model on the development dataset and use that performance to decide when to save models to file.

The train and development dataset have been predefined in the Flickr_8k.trainImages.txt and Flickr_8k.devImages.txt files respectively, that both contain lists of photo file names. From these file names, we can extract the photo identifiers and use these identifiers to filter photos and descriptions for each set.

3.2 Defining the Model

We defined a deep learning based on the “Merge-Model” described by Marc Tanti.



The author provides nice schematic of the model, reproduced below:

Model describe in three parts:

1. Photo Feature Extractor:
   This is a 16-layer VGG model pre-trained on the ImageNet dataset. We have pre-processed the photos with the VGG model (without the output layer) and will use the extracted features predicted by this model as input.

2. Sequence Processor:
   This is a word embedding layer for handling the text input, followed by a Long Short-Term Memory (LSTM) recurrent neural network layer.

3. Decoder (for lack of a better name):
   Both the feature extractor and sequence processor output a fixed-length vector. These are merged together and processed by a Dense layer to make a final prediction.

The Photo Feature Extractor model expects input photo features to be a vector of 4,096 elements. These are processed by a Dense layer to produce a 256 element representation of the photo.

The Sequence Processor model expects input sequences with a pre-defined length (34 words) which are fed into an Embedding layer that uses a mask to ignore padded values. This is followed by an LSTM layer with 256 memory units.

Both the input models produce a 256 element vector. Further, both input models use regularization in the form of 50% dropout. This is to reduce overfitting the training dataset, as this model configuration learns very fast.

The Decoder model merges the vectors from both input models using an addition operation. This is then fed to a Dense 256 neuron layer and then to a final output Dense layer that makes a softmax prediction over the entire output vocabulary for the next word in the sequence.

3.3 Fitting the Model

Before Fitting the Model understand the summary of the model. In the input_2 layer that is the first input layer we pass the sequence of indices of partial caption. After that it passed to the embedding_1 or the embedding in this every index gets mapped to the 200 dimensional vector. Alongside we pass the image feature vector of length 2048 in the input_1. Output of the embedding_1 is then passed to the dropout_2 and the output of the input layer that is input_1 is then passed to the dropout_1 to avoid over-fitting of the model.

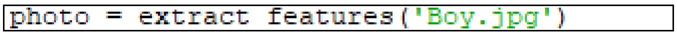
The output of the dropout_2 is then passed to input of lstm_1 and output of the dropout_1 is then passed to the dense_1. Output of lstm_1 in case of image caption model and output of dense _1 in case of image model is of the shape (batch_size,256).Since both input tensors to this layers is of same shape they can be merged into a single tensor by tensor addition which is state as add_1 then we add another dense layer dense_2. In the end we add another dense layer which is also the output layer and has an activation function softmax which generates probability distribution across all the 1652 words in the vocabulary.

After defining the model we compile the model with categorical_crossentropy and adam optimizer. Model is train by calling model.fit_generator function with 20 epochs.

3.4 Complete Example

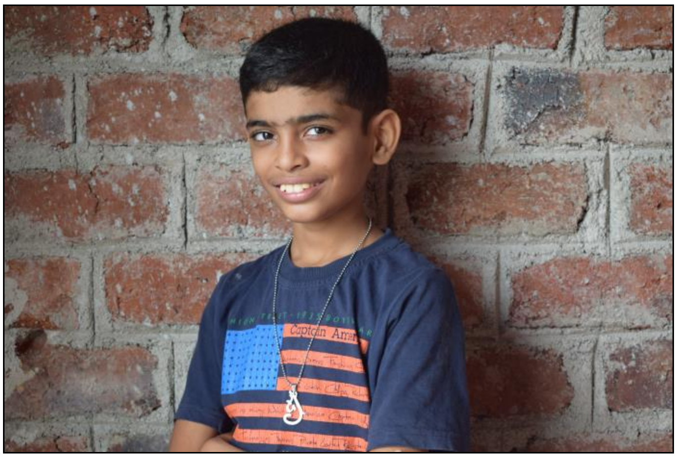
We have developed a command line user interface to generate captions for the image.

We provide the input image using arguments.

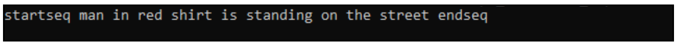


```
photo = extract features('Boy.jpg')
```

After the image is transferred to the model for caption generation.



And at last caption is displayed on the screen.



```
startseq man in red shirt is standing on the street endseq
```

4. CONCLUSIONS

After studying different research papers published nationally and internationally, we have seen many types of algorithms, techniques. We implemented LSTM using the Keras and tensorflow which reduced the lines of codes which are actually required, if we implemented everything from scratch but we also faced some difficulties also such as customization of the model as we want which result in moderate results and we are going to tackle that problem by exploring the Keras and TensorFlow library, and bring more customization to our project. Although we learned a lot about the implementation of deep learning algorithms using the keras and tensorflow libraries.

5. REFERENCES

[1] Vinyals, Oriol, et al. "Show and tell: A neural image caption generator." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[3] Fang, Hao, et al. "From captions to visual concepts and back." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[4] Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[5] Johnson, Justin, Andrej Karpathy, and Li Fei-Fei. "Densecap: Fully convolutional localization networks for dense captioning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[6] Wang, Cheng, et al. "Image captioning with deep bidirectional LSTMs." Proceedings of the 2016 ACM on Multimedia Conference. ACM, 2016.

[7] Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International Conference on Machine Learning. 2015.

[8] Papineni, Kishore, et al. "BLEU: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics, 2002.

[9] Vedantam, Ramakrishna, C. Lawrence Zitnick, and Devi Parikh. "Cider: Consensus-based image description evaluation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

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