Image Caption Generator using ResNet50 and LSTM model

Phalak Sharma **Rupam Goyal** **Tushar**

CO17345 **CO17351** **CO17361**

Abstract

“An image says alot more than words” is true for humans but when we talk about computers, telling what an image is representing seems inconceivable. Humans can deduce not just one, but a number of descriptions for just a single image and until the coming up of deep neural networks, this was not practical for a computer. But with the recent development of Deep Neural Networks, availability of huge datasets and computer power, it is possible to generate captions for an image. This project involves the use of Computer Vision and Natural Language Processing. In this paper, we have tried to develop a model which can take an image as an input and output a sentence that can describe the things in that picture. The model uses the Flickr8 dataset for the training purpose. The components of the method we used are: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and sentence generation. The image is captioned by recognizing the objects that appear in the input image, using automatic feature engineering. Image captioning has alot of important usage these days as the motivation behind this project comes from several real-life scenarios like- Self driving cars, Aid to the blind etc. The performance can be measured with the help of BLEU score. With certain advancements and reducing the training dataset size, Image Captioning using CNN can solve alot of purposes.

Keywords- Computer Vision, Natural Language Processing, CNN, RNN, BLEU score, automatic feature engineering

1.0 INTRODUCTION

Automatically describing the content of an image is a fundamental problem in AI that combines computer vision and natural language processing. For a machine to be able to automatically describe objects in an image with its relationships or the work being done using a learned language model is a daunting task, but plays an essential role in many areas. For example, it can help visually impaired people with better under-stand visual input, thereby acting as a facilitator or a guide. Not only should the model be able to solve computer vision challenges of identifying objects in the image, but it must also be intelligent enough to capture and express the object's relationships in natural language. For this reason, the image caption generation is considered the difficult problem. Its purpose is to mimic the human ability to understand and process visuals heavily in the creation of descriptive language, making it an attractive problem in the major problem of AI. The generated image caption should not only contain the image object names, but also their properties, relationships and functions. In addition, the generated caption must be expressed through a natural language such as English. This means that the already prevalent neural language model requires an additional visual information to generate an image caption. There are numbers of research for solving this problem. Some of them [6] [22] [4] offer a combination of existing image Object detection and sentence creation system. But there is a more efficient solution [20] that provides a combined model. It takes an image and generates caption, which adequately describes the image. Heads of major tech companies are investing heavily in deep learning and AI research, because of which the particular problem of image captioning is being studied in many organizations by many different teams. In the last few years, one of the most common frameworks applied in this line of research is the neural network model composed of two sub-networks, where a convolutional neural network (CNN) is used to encode an image into an image representation; Whereas a recurrent neural network (RNN) is applied to decode a natural language description. The Long Short-Term Memory (LSTM) model based on RNN architecture has emerged as one of the most popular choice, as it has the ability to capture long-term dependencies and preserve sequence. Recently, several variants of this LSTM framework were introduced and achieved good results, such as with attention mechanisms and features.

In this paper, we have tried to develop a model that can take an image as input and output a sentence that can describe things in that picture. The model uses the Flickr8 dataset for training purpose. The components of the method we use are: Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Sentence Generation. In this paper, CNN is used to create a dense feature vector. This dense vector, also called an embedding, is used as a feature input to other algorithms or networks. For an image caption model, it becomes a dense representation of the embedding image and is used as the initial state of the LSTM. In this image caption model, we created the embedding of the image. This embedding will be fed into an LSTM as the initial state. This becomes the first previous state of the language model, which affects the next predicted words.

At each time-step, the LSTM considers the previous cell position and outputs a prediction for the most likely next value in the sequence. This process is repeated until the end is sampled, indicating the end of the caption. Image captioning is an important use these days, as the inspiration behind this project comes from many real-life scenarios such as self-driving cars, ad to blind, etc.

2.0 MOTIVATION

Computers are just machines and have no senses like humans do. Most importantly, they do not have sense of vision and hence cannot perform a task for a visual input. This was true until Computer Vision came to rescue. It is quite a challenging task to automatically describe the content of an image using properly formed English sentences. Indeed, a description must capture not only the objects in an image, but also express how these objects relate to each other as well as their attributes and the activities involved. Moreover, the above semantic knowledge has to be expressed in a natural language like English, which means that a language model is needed in addition to visual understanding.

The inspiration of our work came from the advent of Deep Learning that this problem can be solved very easily if we have the required dataset and the models which can be used for feature extractions from the images and relating these features with English language vocabulary to generate a description.

Many real-world problems have led us to the development of such a model. Few of them are listed below:

- Self-driving cars — Automatic driving is a big challenge and if we can properly caption the scenes around a car, it can give a boost to the self-driving system.
- Aid to the blind — We can create a product for the blind people which can guide them while travelling on the roads without the support of anyone else. We can do this by first converting the scene into text and then to voice. Both are now famous applications of Deep Learning.
- CCTV cameras are everywhere these days, but along with viewing the world, if we are also able to generate relevant captions, then we can raise alarms for any malicious activity going on somewhere. This could probably help reduce some crime and/or accidents.
- Automatic Captioning can help, make Google Image Search as good as Google Search, as then every image could be first converted into a caption and then search can be performed based on that caption.

3.0 RELATED WORK

The problem of generating human language descriptions from visual data has long been studied in computer vision [7, 25]. This led to complex systems of visual primitive recognizers combined with a structured formal language, which are further converted to natural language via rule-based systems. The problem of still image description with natural text has gained interest more recently. Leveraging recent advances in object recognition, their attributes and locations, allows us to drive natural language generation systems, though these are limited in their expressivity. Farhadi et al. [6] use detections to infer a triplet of scene elements which is converted to text using templates. Similarly, Li et al. [19] start off with detections and piece together a final description using phrases containing detected objects and relationships. A more complex graph of detections beyond triplets is used by Kulkani et al. [16], but with template-based text generation. More powerful language models based on language parsing have been used as well [23, 1, 17, 18, 5]. The above approaches have been able to describe images “in the wild”, but they are rigid when it comes to text generation. A large body of work has addressed the problem of ranking descriptions for a given image [11, 8, 24]. Most closely, neural networks are used to co-embed images and sentences together [24] or even image crops and sub sentences [13] but do not attempt to generate novel descriptions. In general, the above approaches cannot describe previously unseen compositions of objects, even though the individual objects might have been observed in the training data. Moreover, they avoid addressing the problem of evaluating how good a generated description is. Karpathy et al. (Karpathy and Fei-Fei,)[4] proposed a visual-semantic alignment (VSA) method. The method generates descriptions of different regions of an image in the form of words or sentences (see Fig. 2). Technically, the method replaces the CNN with Region-based convolutional Networks (RCNN) so that the extracted visual features are aligned to particular regions of the image. The limitation is that the method consists of two separate models. This method is further developed to dense captioning (Johnson et al., 2016) and image-based question and answering system

In our work we have combined deep convolutional network for image classification [12] with recurrent networks for sequence modeling [10], to create a single network that generates descriptions of images. The model is inspired by recent successes of sequence generation in machine translation [3, 2, 26], with the difference that we provide an image processed by a convolutional net. The closest works are by Kiros et al. [15] who use a neural net, but a feedforward one, to predict the next word given the image and previous words. A recent work by Mao et al. [21] uses a recurrent NN for the same prediction task.

This is very similar to the present proposal but there are a number of important differences: we use a more powerful RNN model, and provide the visual input to the RNN model directly, which makes it possible for the RNN to keep track of the objects that have been explained by the text. Lastly, as Kiros et al. [14], we have proposed to construct a joint multimodal embedding space by using a powerful computer vision model and an LSTM that encodes text.

4.0 PROPOSED SYSTEM

The section discusses in detail the dataset, workflow, methods and experimental setup used in the proposed work.

4.1 Dataset Description

To work with any Deep Neural Network project, a dataset is important to work on so as to train the model and make it learn what we want it to do. For Image Captioning, a large dataset with alot of variety of images is required to make the computer be able to learn to captions images with a different object.

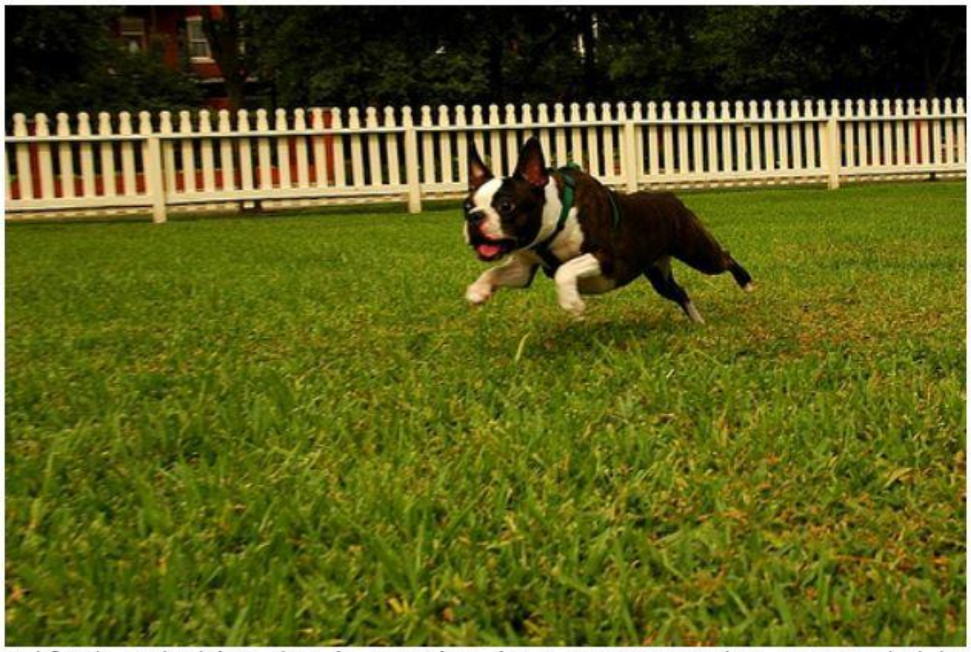
In our work, we have used the Flickr8k image dataset to train the model for understanding how to discover the relation between images and words for generating captions. Some other big datasets like Flickr_30K and MSCOCO dataset are also available, but it can be quite complex to train the network so we will be using a small Flickr8k dataset. The advantage of a huge dataset is that we can build better models.

It contains 8000 images in JPEG format with different shapes and sizes and each image has 5 different captions. The images are chosen from 6 different Flickr groups, and do not contain any well-known people or locations. These were manually selected to depict a variety of scenes and situations.

The images are bifurcated as follows in the code:

- Training Set — 6000 images
- Dev Set — 1000 images
- Test Set — 1000 images

There are also some text files related to the images. One of the files is “Flickr8k.token.txt” which has each image along with its 5 captions. Every line contains the <image name>#i <caption>, where 0≤i≤4 i.e. the name of the image, caption number (0 to 4) and the actual caption.



```
A black and white dog is running in a grassy garden surrounded by a white fence .
A black and white dog is running through the grass .
A Boston terrier is running in the grass .
A Boston Terrier is running on lush green grass in front of a white fence .
A dog runs on the green grass near a wooden fence .
```

Fig1. An example from dataset

Why this dataset?

- Training a model with large number of images may not be feasible on a system which is not a very high end.
- Data is properly labelled. For each image, five captions are provided. Hence more objects can be learnt by the model.

We are working with 2 kinds of data –images and text. The images will help in feature extraction and the captions will enable to match the words with particular images and thus train the model to generate a new description for any input image.

4.2 Models used for caption generator

4.2.1 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a deep, feed-forward neural network which is mostly used in image processing applications. CNNs require little preprocessing as it learns the features by itself from the given data. A CNN consists of an input layer, an output layer and multiple hidden layers. The hidden layers of a CNN include convolutional layers, pooling layers, fully connected layers and normalization layers.

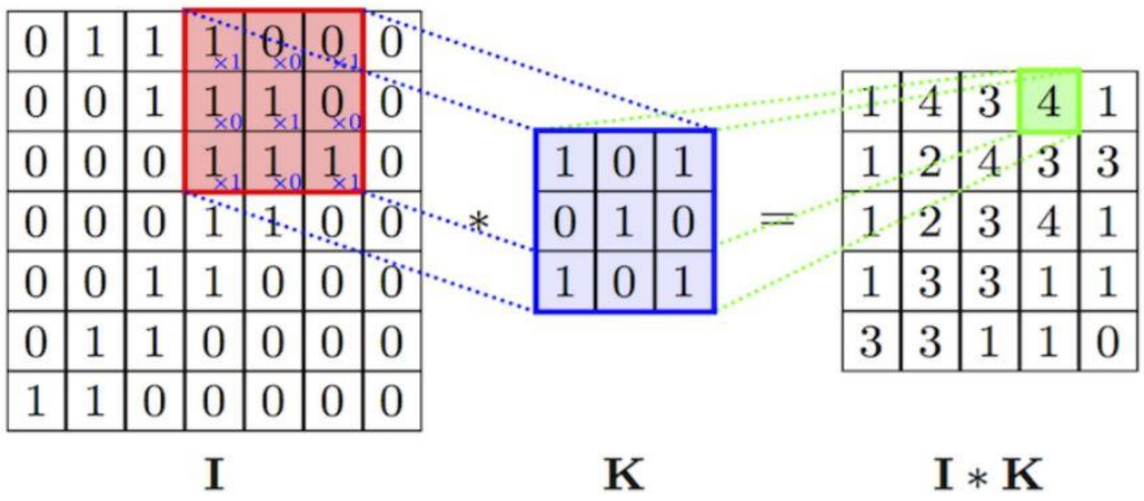


Fig2. Convolutional Neural Networks

1. Convolutional layer

This is the first layer which extracts features from images. It performs a mathematical operation taking the image and filter as input and returns a Feature map as output.

2. Pooling layer

Pooling layer is used when the images are large as it reduces the number of parameters in an image. This can be of different types:

- Max Pooling
- Average Pooling
- Sum Pooling

3. Fully connected layer

The input to Fully connected network is matrix flattened into a vector. Features are passed into this which creates a model to determine the weights of the edges. In the end, we have an activation function as a softmax or sigmoid function to classify the images.

4. Normalization layer

This layer helps in normalizing the outputs from a previous layer by changing the output values between 0 to 1. This layer helps in making the model to train faster.

4.2.1.1 ResNet50

ResNet50 is a type of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has 3.8 x 10^9 Floating points operations. The ResNet-50 model consists of 5 stages, each further consisting of a convolution and an Identity block. Each convolution block has 3 convolution layers and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters.

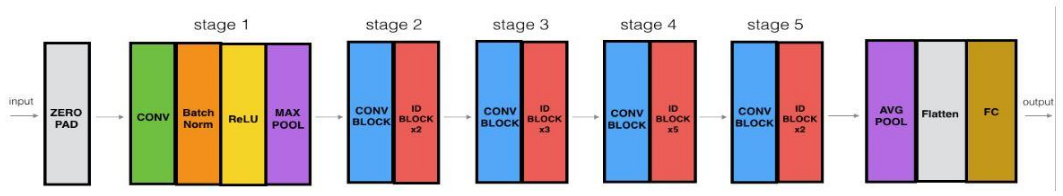


Fig3. ResNet50 Model

In stage 1 of ResNet-50 model, 2D convolution layer has 64 filters of shape (7,7), followed by BatchNorm and MaxPooling that uses a (3,3) window and a (2,2) stride.

In stage 2 of the model, convolutional block uses three sets of filters of size [64,64,256]. Also, it has two identity blocks that use three sets of filters of same size as the convolution block.

In stage 3 of the model, convolutional block uses three sets of filters of size [128,128,256] and three identity blocks that use three sets of filters.

In stage 4 of the model, convolutional block uses three sets of filters of size [256,256,1024] and 5 identity blocks that use three sets of filters.

In stage 5 of the model, convolutional block uses three sets of filters of size [512,512,2048] and two identity blocks that use three sets of filters.

Further the model uses 2D Average Pooling layer that uses a window of shape (2,2).

the fully connected layer reduces its input to the number of classes using the softmax activation.

4.2.2 Recurrent Neural Networks

A Recurrent Neural Network (RNN) has nodes connected in the form of a directed graph like a sequence. This facilitates RNN to handle series operations like time sequence, handwriting recognition and speech sequence. RNNs can remember important information about the input received and hence enables them to predict the next element in a sequence. In RNN, the information loops between the nodes. RNNs predict results by considering the current input and also uses the previous inputs before making a decision.

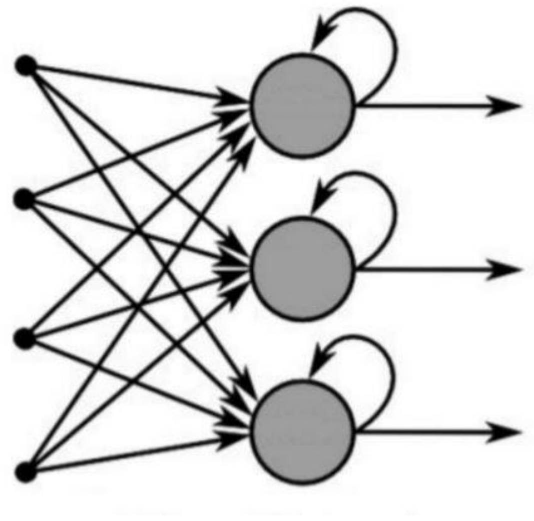


Fig4. RNN Model

4.2.2.1 LSTM

Long short-term memory (LSTM) network is a recurrent neural network. LSTM is used in the field of deep learning and has proved to be extremely effective for sequence prediction problems. The reason they worked so well is because LSTM is able to store past information that is important and forget the information that is not. LSTMs are specifically designed to avoid the problem of long-term dependency. Remembering information for a long time is their default behaviour. All standard neural networks have complex multiplication methods for neural network modules. For standard RNNs, this multiplication module will have a much simpler structure, such as the width of a single tanh layer. LSTMs also have this module as a structure, but the replication module has a different structure. Instead of having one neural layer, rather than four, they communicate in a special way.

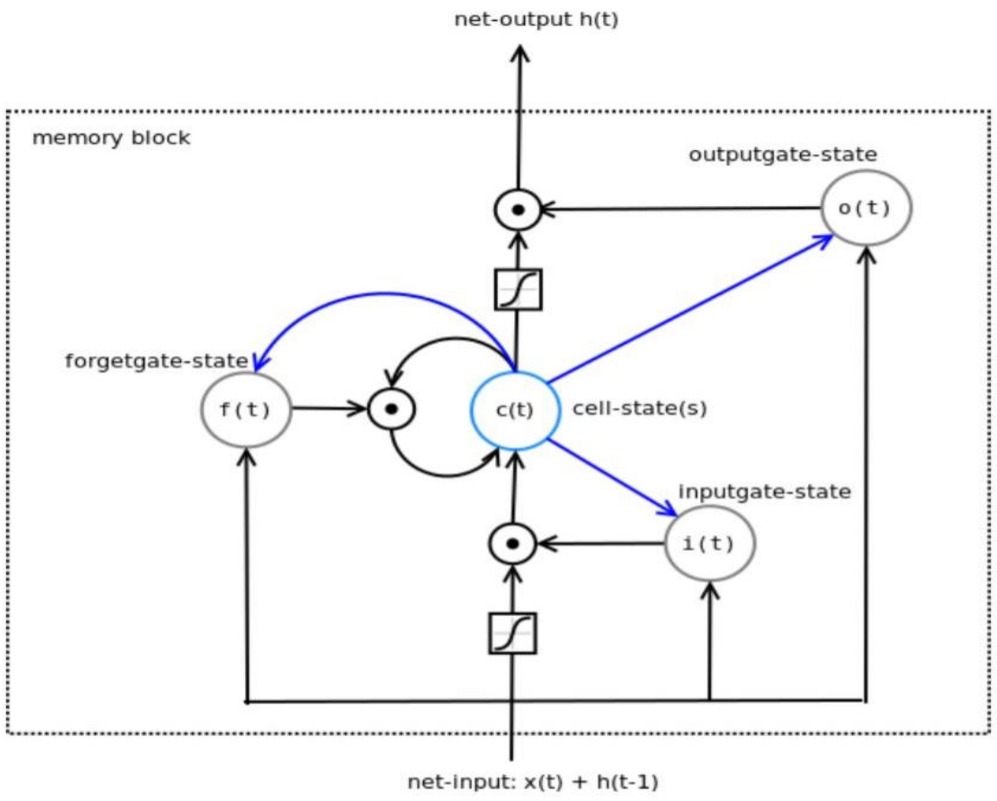


Fig5. LSTM Architecture

Recurrent neural networks are designed to use the predicted values at time t in order to predict the values at time t+1, therefore LSTM also remembers past events and patterns of a time which makes it useful in various aspects.

LSTM doesn't have neurons, whereas it has memory blocks which are connected through its different layers. a memory block has different components that help it regulate how value flows through the system. There are three types of gates:

The input gate controls the extent to which new values flow into memory, i.e., adds information to the cell state

The forget gate controls the extent to which a value remains in memory, i.e., removes the information that is no longer required by the model

The output gate controls the extent to which the input and memory is used to compute the output, i.e., selects the information to be shown as output

4.2.3 Sequential Model

Sequential Model is used for a stack of layers where each layer has exactly one input and one output. Sequential model groups linear stack of layers and provides training and inference features. a sequential model can be created by passing a list of layers to the sequential constructor all it can be created by incrementing the layers to the model one by one by the 'add()' method. Also, there is a method named 'pop()', which is used to remove layers. So, we can say that a sequential model behaves very much like a list of layers.

in most deep learning network that we will belt we are most likely to use the sequential model because it allows us to easily stack up the sequential layers off the network in order from input to output.

4.3 Model Development

4.3.1 Model Architecture

A merge-model architecture is used in this project to create an image caption generator. In this model, the encoded features of an image are used along with the encoded text data to generate the next word in the caption. In this approach, RNN is used only to encode text data and is not dependent on the features of the image. After the captions have been encoded, those features are then merged with the image vector in another multimodal layer which comes after the RNN encoding layer. This architecture model provides the advantage of feeding preprocessed text data to the model instead of raw data.

The model has three major blocks:

• Image feature extractor

• Text processor

• Output predictor

4.3.2 Image Feature Extractor

The feature extractor needs an image 224x224x3 size. The model uses ResNet50 pretrained on ImageNet dataset where the features of the image are extracted just before the last layer of classification. Another dense layer is added and converted to get a vector of length 2048.

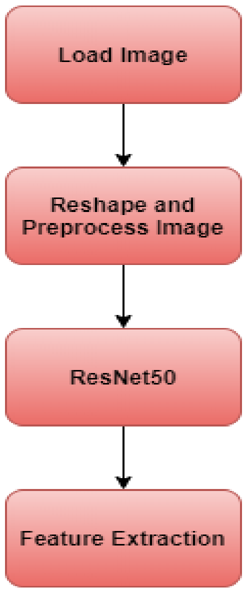


Fig6. Flowchart of Image Feature Extraction

4.3.3 Text Processor

This layer contains the word embedding layer to encode text data. Long Short-Term Memory (LSTM) in RNN is added with 256 memory units. This unit also outputs a vector of length 128.

4.3.4 Output Predictor

Output vector from both the image feature extractor and the text processor are of same length (128) and a decoder merges both the vectors using an addition operation. This is then fed into two dense layers. The first layer is of length 128 and the second layer makes a prediction of the most probable next word in the caption. This layer uses softmax activation function to predict the most probable next word in the vocabulary.

4.3.5 Fitting the Model

After building the model, the model is fit using the training dataset. The model is made to run for 210 epochs and the best model is chosen among the 210 epochs by computing loss function on Flickr8k development dataset. The model with the lowest loss function is chosen for generating captions.

4.4 Working

It involves various phases including Importing the dataset and preprocess, Image Resizing and feature extraction, Vocabulary definition and indexing, Data generator, Training the model, Testing the model and Caption Generation

4.4.1 Importing the dataset and preprocess

All the data that we will work upon has been imported and we have performed the basic cleaning and preprocessing. The textual data has been sorted and each image number and its corresponding caption has been segmented intp individual rows. A dictionary is made to store the images along with their captions, where the images are keys and the captions serve as values. All the 5 captions are stored for each image. Also, the dataset has been divided into 3 –train, validation and test sets and 3 files are made respectively containing the images and captions for each set. Next, we have added, <start> and <end> sequence tokens before and after each caption and stored in encoded version in the files.

4.4.2 Image Resizing and feature extraction

For feature extraction, we have used a pretrained model, i.e. we have used transfer learning technique. We have used ResNet50 model for the same with weights as Imagenet having already pre-trained models on standard Imagenet dataset provided in keras. It is a standard dataset used for classification and contains more than 14 million images in the dataset, with little more than 21 thousand groups or classes. However, our aim is not to classify, but just get fixed-length informative vector for each image. This process is called **automatic feature engineering.** Then this vector will be used for the generation of the caption.

But the images in the dataset are of different shapes and sizes whereas the feature extraction model takes only a fixed size vector. To input images into the feature extraction model, we have to resize them into target size which in our case is ResNet50 which takes 224x224x3 as input size.

From our model, we have removed soft-max layer to use it as a feature extractor. For a given input image, ResNet50 gives us 2048-dimensional feature extracted vector.

All the extracted features are stored in a Python dictionary and saved on the disk using Pickle file, namely whose keys are image names and values are corresponding 2048 length feature vector.

4.4.3 Vocabulary definition and indexing

To define the vocabulary from which the captions for the input images will be predicted, we had to find the words from all the captions i.e. we need to tokenize each caption and list all unique words from all them. From the training dataset we made a vocabulary of 8253 words. As computers do not understand English words, we have represented them with numbers and mapped each word of the vocabulary with a unique index value. Also to decide the language model parameters, we need to specify each caption in a fixed size, hence we calculated the maximum length of the captions. Max_length of caption is 40.

4.4.4 Data generator

The prediction of the entire caption, given the image does not happen at once. It is done word by word. Thus, we encoded each word into a fixed sized vector and represented each word as a number. The main purpose of this module is to generate various formats for images and captions, which could be further used in various models while implementing the training of data. Firstly, is capsule is considered and it is splitted into words and it referenced to its corresponding index. Then, beginning from the '<start>' prefix, each time we append next word in the partial sequence, the number of elements in the same are incremented by 1. Also, we maintain a list for each caption that stores the next word at each sub-iteration. Further, one hot encoding is applied on the list that contains the next word. Further, both partial sequence and one hot encoded next word are converted into arrays.

INPUT SEQUENCE	NEXT WORD
<start>	A
<start>, A	black
<start>, A, black	dog
<start>, A, black, dog	is

<start>, A, black, dog, is	running
<start>, A, black, dog, is, running	after
<start>, A, black, dog, is, running, after	a
<start>, A, black, dog, is, running, after, a	white
<start>, A, black, dog, is, running, after, a, white	dog
<start>, A, black, dog, is, running, after, a, white, dog	in
<start>, A, black, dog, is, running, after, a, white, dog, in	the
<start>, A, black, dog, is, running, after, a, white, dog, in, the	snow
<start>, A, black, dog, is, running, after, a, white, dog, in, the, snow	.
<start>, A, black, dog, is, running, after, a, white, dog, in, the, snow, .	<end>

Table 1. Data Generator

Then, keeping in mind space and time complexities, only '2000' images and their captions are considered. The formats of this images and captions are replicated and manipulated, which leads to generation of new formats of images and captions. Further, different files are created and, images and captions in different forms are stored respectively in these files.

4.4.5 Training the model

In the training of the model, we first apply the Sequential model which contains a Dense layer that uses 'relu' as the element-wise activation function. Then we add a Repeat vector layer with argument '40', which would repeat the input 40 times.

Then we apply another sequential model, in which we use the Embedding layer as the first layer of the model. In this layer, the input dimension is the size of the vocabulary and the output dimension, i.e., the dimensions of the dense embedding is declared to be '128'. Further we will add a LSTM layer with '256' as the dimensions of the output space and the Boolean 'return_sequences' is set TRUE, which would return the full sequence and not just the last output. Then there is a TimeDistributed layer whose argument is a regular deeply connected neural network of size '128'. This layer will help in applying a layer to every temporal size of an input.

After defining the above two sequential models, we concatenate them, followed by addition of two LSTM layers with output dimensionality equal to '128' and '512' respectively. Then we will apply a Dense layer, that would deeply connect the network with size equal to the vocabulary size. Then the fully connected layer reduces its input to number of classes using softmax activation

For the we train the model by passing the images as a list whose dimensions were reshaped after applying the ResNet50 model; and captions as a list of list which would store the the sequence in which a caption is generated word by word for each image. Here, the batch size, i.e., the number of samples trained in a single iteration is defined as '512' and the number of epochs, i.e., the number of times the entire dataset is pass both forward and backward is set to '210'.

At last, the result of the training of the model is important in the form of weights and is also saved to a file for future reference.

4.4.6 Testing the model and Caption Generation

To test our trained model, we input an image to the model. Next the image is fed into the feature extractor to recognize what all objects and scenes are depicted in the image, after resizing it. The process of caption generation is done using the RNN trained model. Then for that image, sequentially, word-by-word the caption is generated by selecting the word with maximum weight for the image at that particular iteration. The indexed word is converted to word and then appended into final caption. When <end> tag is detected or the size of the caption reaches 40, the final caption is generated and printed along with the input image.

5.0 RESULTS

Through the testing phase of our implemented model, we found out that the model can generate sensible descriptions of images in valid English sentences. The generated captions are helpful enough to tell about the objects or elements in the image. Such Image Captioning can be helpful to visually impaired people, for image search and autonomous driving system etc. The loss and accuracy of the system has helped us achieve such good results.

We pictorially represent the value of loss and accuracy at every epoch in the respective two graphs. Loss value would decrease, and accuracy value would increase as we move from lower to higher number of epochs. Also, more the number of epochs, more would be the smoothness of these curves.

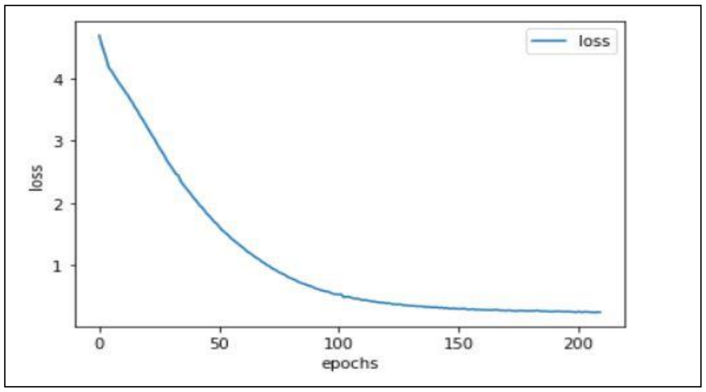


Fig7. Graph of loss vs epochs

Loss: 0.2441

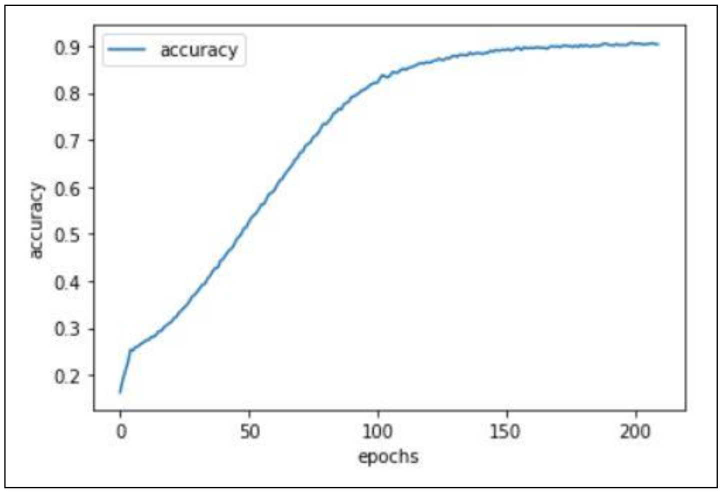
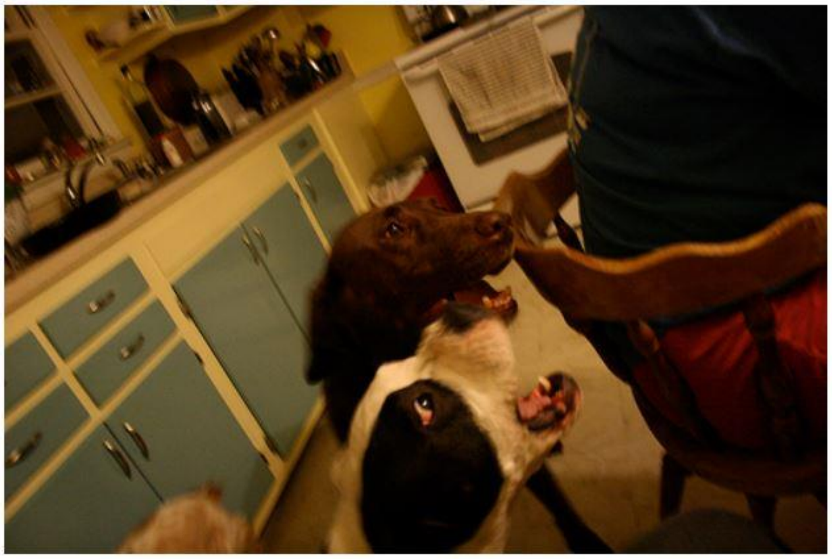


Fig8. Graph of accuracy vs epochs

Accuracy: 90.29%

5.1 Outputs generated from the model

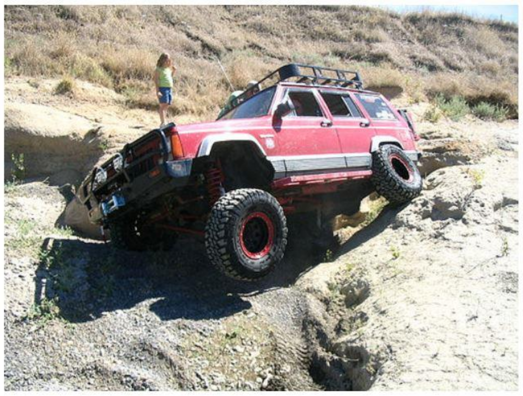
Some sample images from within and outside the dataset have been tested and we have got the following results.



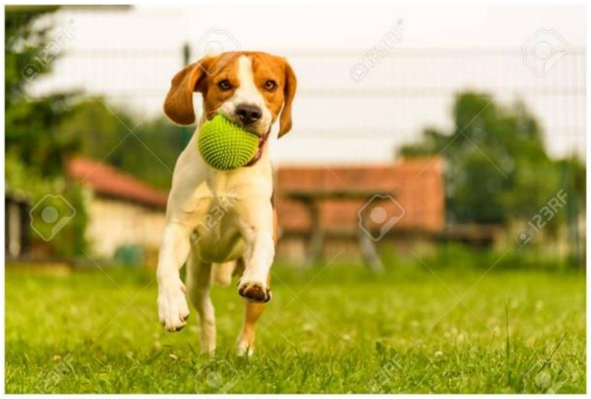
```
A dog in a headwraps pushes two down a man to the seat .
```

```
A kid of girl is playing by a ball with a and ball on kick other .
```

```
A man wearing a helmet riding glove to a red helmet wearing a help .
```



```
A group of people are steamboat on a rocky road .
```



```
A black and white dog is playing with a ball to to its grass .
```

Fig9. Outputs generated from the model

6.0 CONCLUSION AND FUTURE SCOPE

We have tried to present a deep learning model that automatically generates novel captions for images. Our described model is based on a CNN that encodes an image into a compact representation, followed by a RNN that generates corresponding sentences based on the learned image features. We have trained the model to maximize the likelihood of the sentence given the image.

Our model worked quite well when we tested it on several images from within and outside the training dataset. The captions it generated for the images were quite accurate. But the source of input image also played an important role in feature extraction and hence caption generation. Certain images from the phones or cameras are not well recognized and hence we found out that there is, still some scope of improvement and that the model can be improvised further. There are certain points which can be incorporated into this model to make it even better.

The points of discussion are as follows:

- We have used a relatively smaller dataset Flickr8k with 8000 images as the larger datasets need a lot of more powerful computers and GPUs. A still larger dataset can be used so that the model can be trained even better. Also, images from different sources can be worked upon so that the model is able to work on input images of all kind.
- The vocabulary size can also be increased by working on more images and captions and hence new captions generated can be diversified and contain a better description of objects and scenes in the image. Also, the length of the caption is 40 words in or model which can be increased by working on longer descriptions in the training phase.
- In our model, the caption is generated word-by-word i.e. we have used greedy algorithm for caption generation. Another approach that can be used is Beam Search in which, at each step, it expands all the successors and hence selects the best result.
- Lastly, for the performance evaluation, the BLEU score can be implemented to the model to check its accuracy in comparison to some other reference captions. The problem here is that all the images need to have some reference captions for them so that a comparison can be drawn, and an accuracy score be generated for the new, model predicted caption.
- Future work can include this text-to-speech technology, so that the generated descriptions are automatically read out loud to visually impaired people.

Implementing some or all these improvements can lead to the development of yet another and advanced model for Image Captioning.

REFERENCES:

[1] A. Aker and R. Gaizauskas. Generating image descriptions using dependency relational patterns. In ACL, 2010. 2

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. arXiv:1409.0473, 2014. 1, 2

[3] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In EMNLP, 2014. 1, 2, 3

[4] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In IEEE CVPR.

[5] D. Elliott and F. Keller. Image description using visual dependency representations. In EMNLP, 2013. 2

[6] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In ECCV, 2010. 1, 2, 5

[7] R. Gerber and H.-H. Nagel. Knowledge representation for the generation of quantified natural language descriptions of vehicle traffic in image sequences. In ICIP. IEEE, 1996. 2

[8] Y. Gong, L. Wang, M. Hodosh, J. Hockenmaier, and S. Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In ECCV, 2014. 2, 5

[9] A. Graves. Generating sequences with recurrent neural networks. arXiv:1308.0850, 2013.

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8), 1997. 2, 3

[11] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. JAIR, 47, 2013. 2, 4, 6, 7

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In arXiv:1502.03167, 2015. 2, 3, 4

[13] A. Karpathy, A. Joulin, and L. Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. NIPS, 2014. 2, 7

[14] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. In arXiv:1411.2539, 2014. 2, 5, 6, 7

[15] R. Kiros and R. Z. R. Salakhutdinov. Multimodal neural language models. In NIPS Deep Learning Workshop, 2013. 2

[16] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating simple image descriptions. In CVPR, 2011. 1, 2, 6

[17] P. Kuznetsova, V. Ordonez, A. C. Berg, T. L. Berg, and Y. Choi. Collective generation of natural image descriptions. In ACL, 2012. 2

[18] P. Kuznetsova, V. Ordonez, T. Berg, and Y. Choi. Treetalk: Composition and compression of trees for image descriptions. ACL, 2(10), 2014. 2, 5, 6

[19] S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi. Composing simple image descriptions using web-scale n-grams. In Conference on Computational Natural Language Learning, 2011. 2

[20] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge," IEEE

[21] J. Mao, W. Xu, Y. Yang, J. Wang, and A. Yuille. Explain images with multimodal recurrent neural networks. In arXiv:1410.1090, 2014. 2, 6, 7

[22] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, "Baby talk: Understanding and generating simple image descriptions," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 12, pp. 2891--2903, 2011.

[23] M. Mitchell, X. Han, J. Dodge, A. Mensch, A. Goyal, A. C. Berg, K. Yamaguchi, T. L. Berg, K. Stratos, and H. D. III. Midge: Generating image descriptions from computer vision detections. In EACL, 2012. 2

[24] R. Socher, A. Karpathy, Q. V. Le, C. Manning, and A. Y. Ng. Grounded compositional semantics for finding and describing images with sentences. In ACL, 2014. 2

[25] B. Z. Yao, X. Yang, L. Lin, M. W. Lee, and S.-C. Zhu. I2t: Image parsing to text description. Proceedings of the IEEE, 98(8), 2010.

[26] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In NIPS, 2014.