ISSN: 2586-7652(Print) Vol. 04, No. 02, September 2021
ISSN: 2635-7607 (Online)
International Journal of Advanced Engineering
Source: http://ictaes.org
Manuscript received: June 14, 2021 ; Revised: August 26, 2021 ; Accepted: August 28, 2021

Image Captioning: An efficient approach using CNN-LSTM

Ghimire.B[1], Regmi.S[2], Niroula.S[3], Panta.R[4], Pandey.R[5], Guragain.MK[6], Kafle.S[7]

[1,2,3,4,5,6,7]*Department of Electronics and Computer Engineering, Purwanchal Campus, Institute of Engineering, Tribhuvan University, Dharan, Nepal*

bhubanghimire3130@gmail.com[1], sakushalregmi@gmail.com[2], santosh073bct@ioepc.edu.np[3], rabinapanta999@gmail.com[4], rajeshpandey2053@gmail.com[5], mkguragain@gmail.com[6], sabinkafley@ioepc.edu.np[7]

Abstract

Captioning images automation is one of the hearts of the human visual system. In this paper, we present a model that automatically detects the objects in the images and generates captions for that image describing the surroundings, based on CNN-LSTM neural networks. To be more specific, this project first detects objects using Transfer Learning based pre-trained models following the use of CNN and LSTM to generate captions. This model can perform two operations. The first one is to detect features in the image using Convolution Neural Networks and the other is to caption the images using RNN based LSTM (Long Short-Term Memory). With all these operations, the main use case of this project is to help the visually impaired to understand the surrounding environment and act according to that. Caption generation is one of the interesting and focused areas of artificial intelligence which has many challenges to pass on. To get relevant results, caption generation projects go through many complex scenarios with the very first task that involves picking the appropriate dataset following other important steps such as training the model, validating the model, creating pre-trained models to test the images, detecting the images and finally generating the captions.

Keywords: *CNN, RNN, LSTM, Captioning*

1. Introduction

Image contains a vast amount of visually perceptible information that is difficult to completely characterize with natural language. Despite the recent advancement in the image annotation field, there remain many challenging tasks. The image annotation model needs to infer the most likely sentence in the form of a sequence

Corresponding Author : Kafley S, Guragain MK.
Author's affiliation : Dept. of Electronics and Computer Engineering, Purwanchal Campus, IoE, TU, Dharan, Nepal
Email: sabinkafley@ioepc.edu.np , mkguragai@gmail.com
Copyright © ICT-AES

of words given an image. Specifying the exact form of this model has many inevitable parts such as how to represent the images, how to represent the sentences, and how to fuse this language model and image feature in one place to generate the best quality caption. There are two general paradigms in image captioning: top-down and bottom-up. The top-down approach starts from the image and converts it into words, while the bottom-up approach comes up with words describing various aspects of an image and then combines them. The top-down approach starts from the image feature extraction to the sentence generation using recurrent neural networks. The top-down approach applies the encode-decode network architecture. Convolution Neural Network used as the encoder whereas Recurrent Neural Network used as a decoder in the top-down approach. The encoder network takes the image into the encoder to get the image features and fed into the decoder to generate the image caption. The bottom-up approach involves several separated tasks, such as identifying objects, arranging words, describing sentences using a language model to extract the image caption. The starting point of the caption generation model is to understand the many images constituent such as the object in it, position of the object, relation among those objects, and many silent features. The relationship between objects and the appearance of objects greatly impacts the determination of the sequential order of words into a sentence. This model should keep only the most relevant silent features and discard others. In this paper, we propose an image captioning system that takes advantage of the parallel structures between images and sentences. The image model will generate the image feature from the provided image whereas the language model learns the caption from the provided caption in parallel. After feature extraction by both models, the captioning model fuses the theme together to generate the best caption for the image.

2. Related Works

Image caption generation has long been a challenging problem in computer vision and natural language processing. A traditional approach is to use predefined templates to generate sentences by filling detected visual elements such as objects [5,12,13,14,15]. [8] proposed a multimodal Recurrent Neural Network (m-RNN) to fuse text information and visual features extracted from images. [9] used a deep convolutional neural network (CNN) to extract the whole image feature, and the feature is only input once to an RNN as the initial start word. [1] proposed to use multiple layers of LSTMs and investigated the best way to feed visual and text information to different LSTMs. Image caption generation is generally divided into two categories: top-down and bottom-up. [3][5] uses the bottom-up approach where they start from visual concepts, objects, attributes, words and phrases and combine them into sentences using language models. There are some recent attempts in this category where [2] and [6] are close to state-of-art performance on various image captioning benchmarks. Top-down approaches are modern ones in which the visual representation comes from the convolutional neural network and recurrent neural networks generate the required caption. [9, 8,4,1,10,7] used this approach. Generally pre-trained models are used for the image feature extraction as in Figure 1 [11].

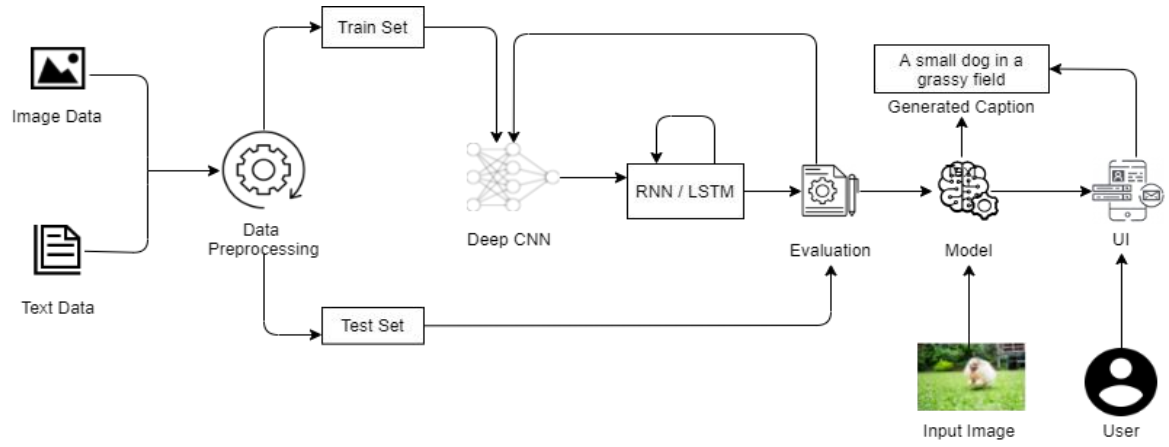


Figure 1 : Process Model

3. Methodology

For a machine to be able to automatically describe objects in an image along with their relationships or the actions being performed using a learnt language model is a challenging task. It requires the use of many algorithms and techniques accompanied with deep learning models. We are creating a web application where the user selects the image and the image is fed into the model that is trained and the generated caption will be displayed on the webpage.

3.1 Modules Description

3.1.1. Dataset Collection

To build a model that generates correct captions we require a dataset of images. We have used Flickr8k dataset due to limited computing resources. The Flicker8k dataset contains two folder Flicker8k Dataset and Flicker8k text. The dataset folder contains the images for train and test the model and Flicker8k text contains the five annotations for each image for Dataset folder. Flickr8k_Dataset contains a total of 8092 images in JPEG format with different shapes and sizes. Of which 6000 are used for training, 1000 for tests and 1000 for development. Flickr8k_text contains text files describing train_set, test_set. Flickr8k.token.txt contains 5 captions for each image i.e., total 40460 captions.

3.1.2 Dataset Preprocessing

The collected data set cannot be used directly for training the model. The datasets go through this phase to extract relevant features for all images to transform visual information into the vector space. 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.pkl” pickle file. The collected data set has the image and caption for each of the images which are pre-processed separately.

(1) Image preprocessing: The images are sub-sampled into the 299x299x3 RGB image.
(2) Text preprocessing: The annotation sentences had various unnecessary words such as stop-words. They are removed. To preprocess the text the following steps are done

(i)Tokenization: The provided sentences are converted into the words. The words are converted into lower case also.

(ii) Indexing: Each unique word is given a unique index and the index numbers are stored in a separate file.

3.2 Features Extraction

This is the most important step for this project. It consists of two steps image features extraction and word embedding:

3.2.1. Image features extraction

The underlying features of the image are extracted using the deep neural network. Convolution Neural Network (CNN) was used to extract the underlying image features [Figure 2]. The CNN consists of multiple layers of neuron to give the feature vector required

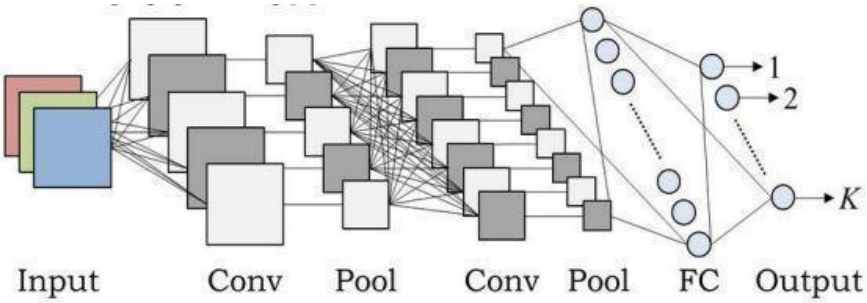


Figure 2: Convolutional Neural network architecture,, Consecutive Dimensionality Reduction by Canonical Correlation Analysis for Visualization of Convolutional Neural Networks - Scientific Figure on ResearchGate, https://www.researchgate.net/figure/An-example-of-CNN-architecture_fig1_320748406 [accessed 25 Aug, 2021]

We used InceptionV3 as CNN architecture. There are altogether 43 layers in InceptionV3. As mentioned above, this model was originally trained on over a million images from 1000 classes on some very powerful machines. The first layers of any neural network are basically responsible for identifying low-level features, such as edges, colors, and blobs, but the last layer is usually very specific to the task they are trained for. The input image in InceptionV3 is of dimension 299x299x3. The last layer in InceptionV3 gives final output but in our case, we need only features of the image. So, we do not use the last layer i.e., dense layer as shown in Figure 3.

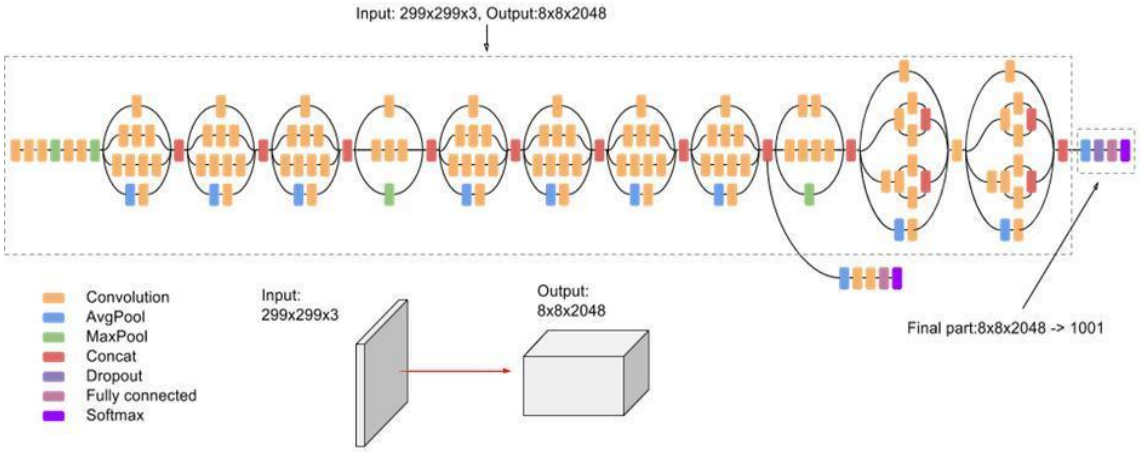


Figure 3: Inception V3, C.Szegedy, V.Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the Inception Architecture for Computer Vision, 2015, https://paperswithcode.com/method/inception-v3, [accessed 25 Aug, 2021]

(1) Convolution layer: It is the first layer of the CNN which consists of a set of filters and the filters are convolved over the image and the features are extracted from the image. Filter kernels used for convolution are initialized and adjusted by back propagation methods using gradient descent.

(2) The pooling layer: This layer reduces the feature size by preserving the important features and removing the others.

3.3 Caption Processing

From all the captions of the given images the unique words are extracted and these unique words are stored in a pickle file named (unique.p). There is a function called enumerator that is used to map unique words to a unique index which is later used to train LSTM. The output obtained from the LSTM is also a unique index.

Captions are read from Flickr8k.token.txt file and stored in dictionary k: v where k = image id and value = [list of caption]. Since there are 5 captions for each image and we have preprocessed and encoded them in below format.

<start> + caption +<end>

The reason behind start and end is,

<start>: Will act as our first word when the feature extracted image vector is fed to the decoder. It will kick-start the caption generation process.

<end>: This will tell the decoder when to stop. We will stop predicting words as soon as the end appears or we have predicted all words from the train dictionary, whichever comes first. We must note that captions are something that we want to predict. So, during the training period, captions will be the target variables (Y) that the model is learning to predict. But the prediction of the entire caption, given the image does not happen at once. We will predict the caption word by word. Thus, we need to encode each word into a fixed size vector.

3.4 Training Deep Neural Network

The extracted features of images and vectors that represent the words are used to train the model. The feature vector of the image and word vector are combined together and passed as input to the deep neural network. There are different neural network layers which are shown in the figure given below.

3.4.1 Sequential Layer

A Sequential Layer is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor. Both the image vector and word vector are of different sizes thus we convert them into the same dimensional vector using the Sequential layer.

3.4.2 Embedding Layer

A word embedding is a class of approaches for representing words and documents using a dense vector representation. It requires that the input data be integer encoded, so that each word is represented by a unique integer. As the embedding layer required the same length for each sentence thus, we pad zeros to make all captions with equal length in the caption preprocessing step. The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset. It is used to learn a word embedding that can be saved and used in our model later. The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments: input_dim: This is the size of the vocabulary in the text data.

output_dim: This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. input_length: This is the length of input sequences. For the word embedding we can also use the pre-trained model like Word2Vec.

3.4.3 Dense Layer

Dense layer is the regular deeply connected neural network layer. It is the most common and frequently used layer. Dense layer does the below operation on the input and returns the output.

output = activation (dot (input, weight) + bias)

Activation functions live inside neural network layers and modify the data they receive before passing it to the next layer. Activation functions give neural networks their power—allowing them to model complex non-linear relationships. By modifying inputs with non-linear functions neural networks can model highly complex relationships between features. Popular activation functions include relu and sigmoid. The weights in a neural network represent the strength of the connections between the different neurons and contain information about the input signal.

3.4.4 Merge Layer

The image feature vector and the word vector generated from the above steps are merged together by using the Keras merge layer [Figure 4].

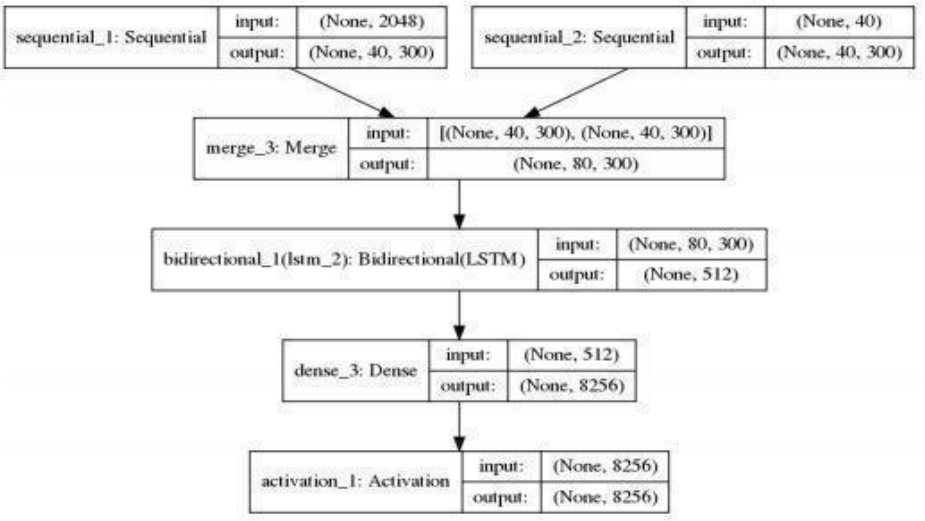


Figure 4: Final Model

3.5 LSTM

For generating the caption, we use the LSTM (Long short-term memory). The merged feature vector is passed into the LSTM.

The image vector is plugged-in first-time stamp only after that the index of the caption is fed into the model. The next word of the caption was predicted in each time stamp by the LSTM cell. The image feature vector was memorized by the LSTM throughout the sequence of the caption learning process.

LSTM is suitable for the sequence of data. The annotation is a sequence of text. LSTM is capable of remembering the long sequence of the data.

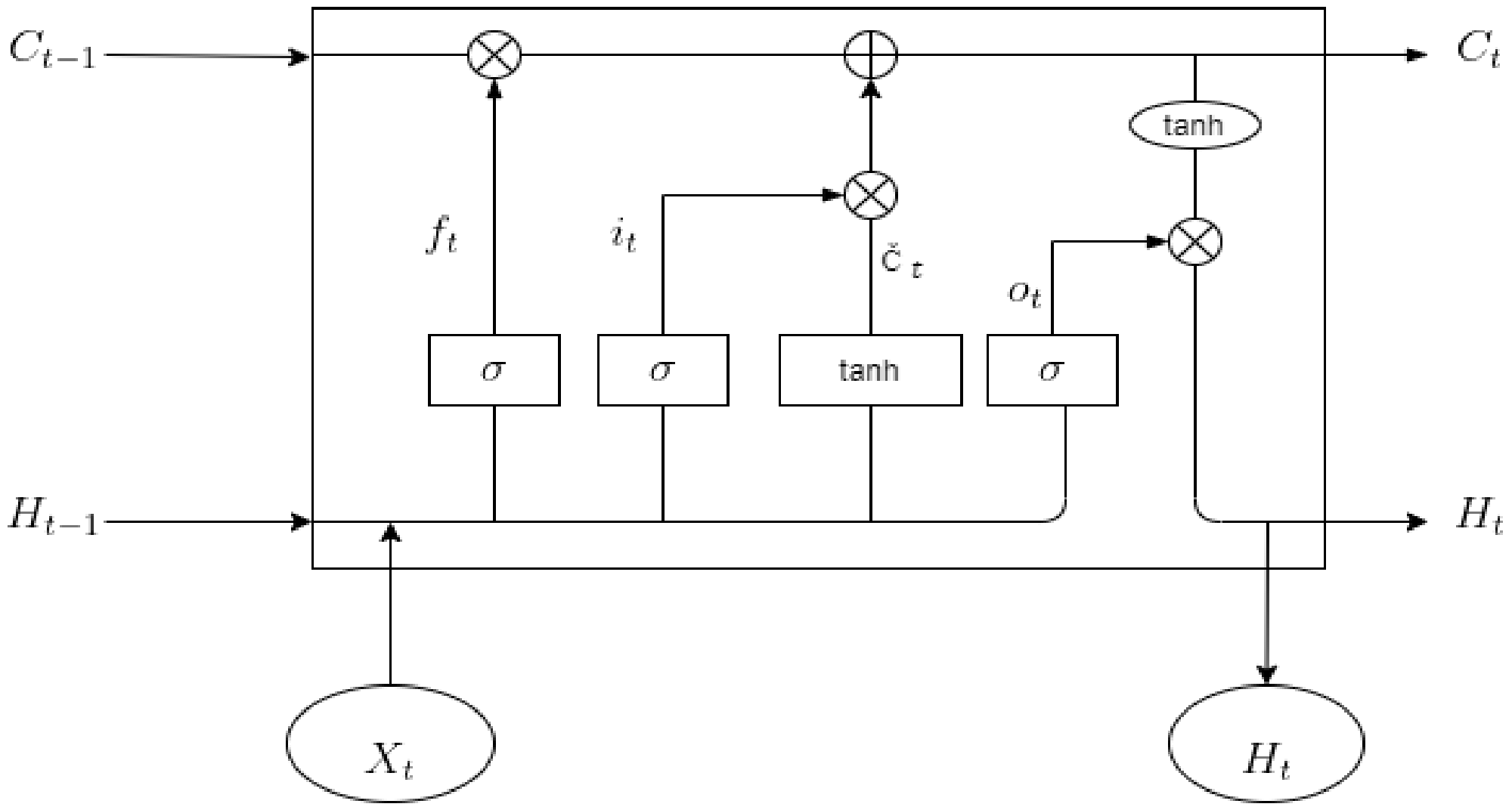


Figure 5: LSTM architecture

LSTM [in Figure 6] consists of three main gates which are used to remember, forget, store and output the context of the sequences. They are given below.

3.5.1 Forget gate

This gate decides which information is thrown away. This gate takes previous cell output and current input. These inputs are passed through the sigmoid layer given by equation (1).

$$f_t = \sigma (W_f * [h_{t-1}, x_t + b_f) \ldots\ldots\ldots\ldots\ldots\ldots \ldots\ldots\ldots\ldots\ldots \ldots(1)$$

3.5.2 Input gate

It is another sigmoid layer which shapes the data between the 0 and 1 and decides which value to update. The candidate values which will be used to update the cell state are calculated by tanh layer. And the output from these two layers is combined to create an update to cell state as given in equation (2) and (3).

$$i_t = \sigma (W_i * [h_{t-1}, x_t]+ b_i)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(2)$$

$$\check{C}_t = tanh (W_c * h_{t-1}, x_t] + b_c)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

After that the old cell state was updated as equation (4).

$$C_t = f_t * C_{t-1} + i_t * \check{C}_t \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4)$$

3.5.3 *Output gate*

Finally, this gate decides which values are getting out of the cell. The sigmoid layer computes the output of the cell using current input and previous cell output. The tanh layer pushes the current cell state in between the -1 and 1 and multiplying it with the sigmoid layer outputs this gate to control the output of the current cell , which is given as in equation (5) and (6).

$$O_t = \sigma\,(W_o * [h_{t-1}, x_t + b_o)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

$$h_t = O_t * tanh\,(O_t)\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(6)$$

Layer (type)	Output Shape	Param #	Connected to
dense_1 (Dense)	(None, 300)	614700	
repeatvector_1 (RepeatVector)	(None, 40, 300)	0	
embedding_1 (Embedding)	(None, 40, 300)	2476800	
lstm_1 (LSTM)	(None, 40, 256)	570368	
timedistributed_1 (TimeDistribut	(None, 40, 300)	77100	
bidirectional_1 (Bidirectional)	(None, 512)	1140736	merge_1[0][0]
dense_3 (Dense)	(None, 8256)	4235328	bidirectional_1[0][0]
activation_1 (Activation)	(None, 8256)	0	dense_3[0][0]

Total params: 9,115,032
Trainable params: 9,115,032
Non-trainable params: 0

Figure 6: Model Summary

3 Result and Discussion

The main target of this project was to generate the caption from the image provided by the user. Among all the processes the most important task was to obtain the stable model with highest level of accuracy by training the convolutional neural network without its own training data set. Two deep learning models: CNN and RNN were used. The whole model summary is shown in Figure 6.

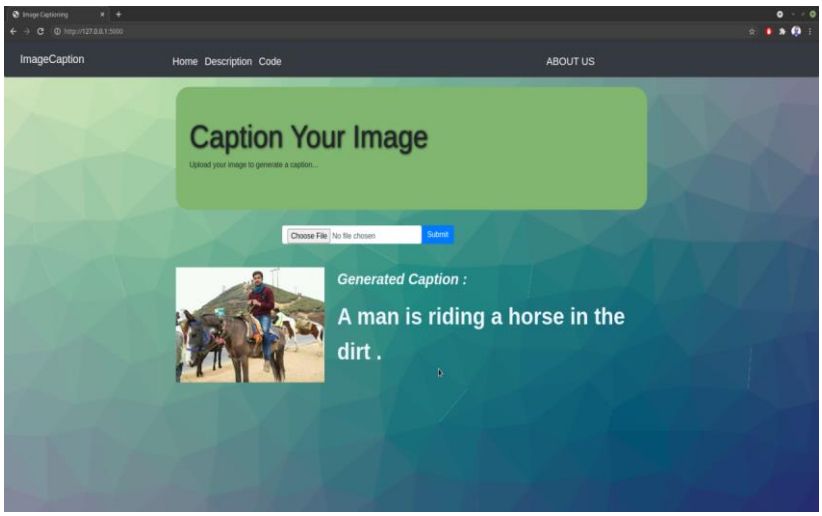


Figure 7: Output-1

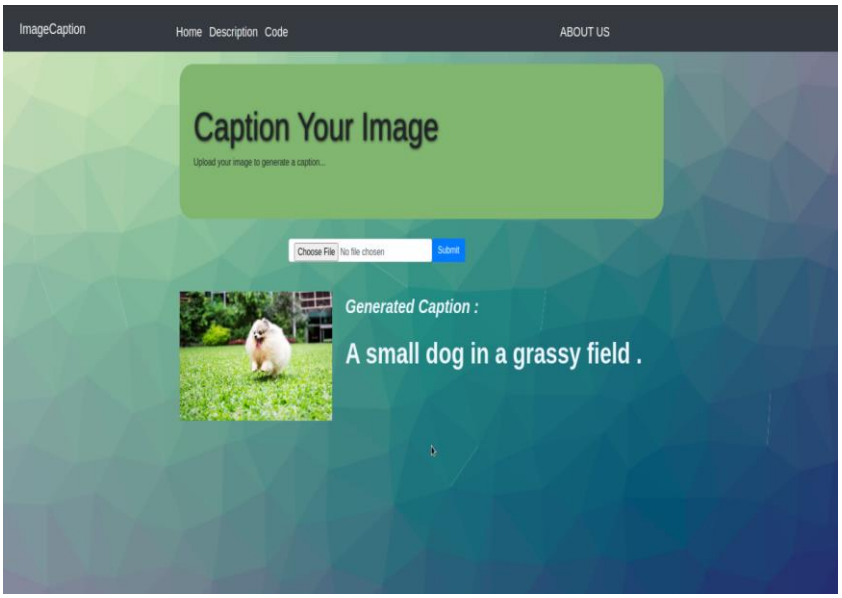


Figure 8: Output-2

Model was trained with 6k images and employed different hyper-parameters such as tweaking batch sizes and number of epochs, learning rates and finally training accuracy is 75 % and model provides well founded caption for test images while analyzing output 1 and sample output-2 as in Figure 7 and 8 above. However due to lack of high computation resources, we managed to train only 6k datasets out of 8k datasets. As a result, the model cannot predict words or generate sentences out of its trained vocabulary sets. The model also fails to generate accurate descriptions if the image is distorted. Some samples of the output generated by our model are shown below:

5. Conclusion

The proposed paper can be useful in generating computerized descriptions of the images and eventually aid visually impaired people in learning from video resources and acting precisely according to interpretations of the surroundings obtained from the model. Highly robust implementations of image caption generators have been developed recently. However, our scope was to generate a model which can leverage low computation power engines using a blend of CNN and LSTM model. More future enhancements can be done by increasing the dataset sample (i.e. using MS COCO datasets), augmenting the layers of neural networks, and appending attention mechanisms in RNN.

References

[1] Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T., & Saenko, K. (2015). Long-term recurrent convolutional networks for visual recognition and description. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2015.7298878

[2]Fang, H., Gupta, S., Iandola, F., Srivastava, R. K., Deng, L., Dollar, P., Gao, J., He, X., Mitchell, M., Platt, J. C., Zitnick, C. L., & Zweig, G. (2015). From captions to visual concepts and back. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2015.7298754

[3] Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., & Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. Computer Vision – ECCV 2010, 15–29. https://doi.org/10.1007/978-3-642-15561-1_2

[4] Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2015.7298932

[5] Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., & Berg, T. L. (2011). Baby talk: Understanding and generating simple image descriptions. CVPR 2011. https://doi.org/10.1109/cvpr.2011.5995466

[6] R. Lebret, P. O. Pinheiro, and R. Collobert. Simple image description generator via a linear phrase-based approach. Proceedings of the International Conference on Learning Representations (ICLR), 2015.

[7] Mao, J., Wei, X., Yang, Y., Wang, J., Huang, Z., & Yuille, A. L. (2015). Learning like a child: Fast novel visual concept learning from sentence descriptions of images. 2015 IEEE International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv.2015.291

[8] J. Mao, W. Xu, Y. Yang, J. Wang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv preprint arXiv:1412.6632, 2014

[9] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2015.7298935

[10] B. Zhou, V. Jagadeesh, and R. Piramuthu. Conceptlearner: Discovering visual concepts from weakly labeled image collections. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.

[11] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84–90. https://doi.org/10.1145/3065386

[12] S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi, “Composing simple image descriptions using web-scale n-grams,” in CoNLL, 2011.

[13] Y. Yang, C. L. Teo, H. Daumé III, and Y. Aloimonos, “Corpus- guided sentence generation of natural images,” in EMNLP, 2011.

[14] M. Mitchell, X. Han, J. Dodge, A. Mensch, A. Goyal, A. Berg, K. Yamaguchi, T. Berg, K. Stratos, and H. Daumé III, “Midge: Generating image descriptions from computer vision detections,” in EACL, 2012.

[15] D. Elliott and F. Keller, “Image Description using Visual Dependency Representations.” in EMNLP, 2013.