KLE TECH

KLE Technological University

Creating Value
Leveraging Knowledge

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

MACHINE LEARNING

Image Captioning Using CNN-LSTM

Project Report

Team No: 08
Akshay Joshi 237
Kartik Kalal 251
Dhiraj Bhandare 252
Vaishnavi Patil 258

Under the Guidance of
Prof. Uday Kulkarni

ABSTRACT

We regularly encounter a large number of images from various fields such as news articles, diagrams, literature, and the internet. Even though several images do not have a description, humans have the capability to understand them without their captions but this isn't the case with machines. As long as machines do not behave, think, and talk like humans, natural language descriptions will be challenging to solve. In this project, CNN-LSTM Model is used to generate captions for images by processing its features. Though image captioning is a complicated task, many researchers have achieved significant improvements. This paper mainly describes an image captioning method using a deep learning approach, a combination of LSTM and Xception model, which shows representative work of the model. This will summarize the images without any human intervention.

ACKNOWLEDGEMENTS

Without mentioning the names of the persons who assisted us in completing our project and its report, the feeling of fulfilment and exhilaration that comes with it would be lacking.

We are indebted to our guide Prof. Uday N Kulkarni who was nothing but a constant source of enthusiasm and whose profound guidance, valuable suggestions, and beneficent direction was mainly responsible for us to complete this project.

We take this opportunity to express our deep sense of gratitude and sincere thanks to our Head , School of Computer Science and Engineering , Dr. Meena S. M. for her tremendous source of inspiration and help in challenging our effort in the right direction.

Last but not the least we would like to thank all the course faculty, teaching and non-teaching staff for helping us during the project.

Akshay Joshi
Kartik Kalal
Dhiraj Bhandare
Vaishnavi Patil

Contents

List of Figures

1 INTRODUCTION

Every picture tells a story. Images are a rich source of data which provide information about different objects their properties, and how they are related to one another. Humans are capable of recognizing the crucial details in a photo and provide a concise summary with just one or a few captions.

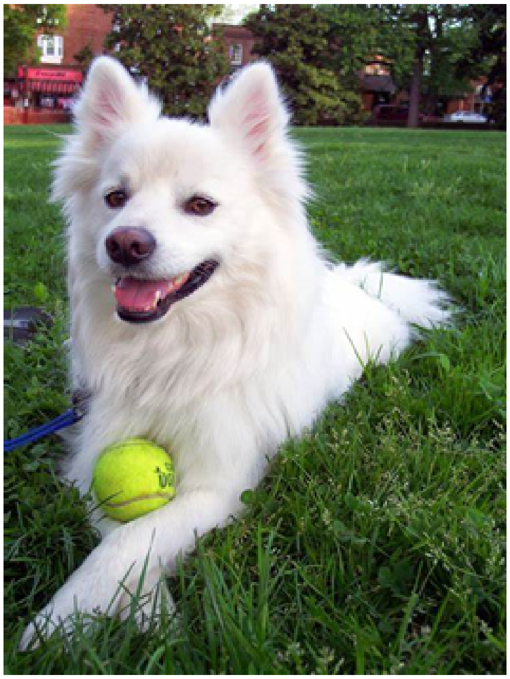


Figure 1: A dog sitting in the grass.

When we ask any human to describe the above image, they will describe it as: “A dog sitting in the grass” or “A white dog playing with a yellow ball”. We do this by seeing the image and creating a meaningful sequence of words simultaneously.

Image captioning is a difficult issue that requires the integration of computer vision and natural language processing techniques. Despite the difficulties, developments in this field could lead to novel and helpful approaches to interacting with and comprehending visual data. This task can be achieved with the help of deep neural networks.

Deep learning is a machine learning method that teaches computers to perform tasks that humans accomplish without thinking about it. It is one of the fundamental technologies that enables driver-less cars to recognize a stop sign, or to tell a lamppost from a pedestrian and vice-versa. Deep learning is the process through which a computer model directly learns to carry out categorization tasks from images, text, or sound. Deep learning algorithms can achieve cutting-edge precision, occasionally surpassing human performance. An enormous amount of labelled data and multi-layered neural network architectures are used to train models.

The process of creating a textual description for given photos is known as image captioning. In the field of deep learning, it has been a crucial and vital task. As it converts images, which are conceived as a sequence of pixels to a sequence of words, image captioning can be seen as an end-to-end Sequence to Sequence problem. For this purpose, we need to process both the language or statements and the images. In this paper, we are generating a model that will be able to generate a caption for an image. Our work is largely inspired by recent developments in machine translation, where the goal is to translate a sentence S written in a source language into its translation T in a target language by maximising P(T—S).To achieve this task, we require both the Language component and the Image component. For the Language component, we will be using Long Short-Term Memory (LSTM) and for the Image component, we will be using Convolution Neural Network (CNN) to obtain the feature vectors. The model will be trained to maximize the likelihood of the target description sentence given the training image.

CNN are a type of Artificial Neural Network (ANN) used frequently in deep learning to interpret visual data.CNNs are regularized versions of multi-layer perceptrons. Fully linked networks, or multi-layer perceptrons, are those in which every neuron in one layer is connected to every neuron in the following layer. In comparison to other image classification methods, CNNs employ comparatively less pre-processing i.e the network automatically learns how to improve the filters (or kernels). The main benefit is the lack of dependence on past knowledge or human assistance in feature extraction. LSTM is an advanced Recurrent Neural Network (RNN), that retains information for a longer period of time. It is used for processing, predicting, and classifying on the basis of time-series data.

1.1 Motivation

Generating captions for images is an important task in both computer vision and natural language processing. A blind or someone with low vision who relies on sound and texts to explain a scene can benefit from it when we utilise it to describe the surroundings for them.

1.2 Objectives

- To understand what is happening in an image and to generate a description that accurately reflects the content of the image.
- To make images more accessible to people with visual impairments.

1.3 Literature Survey

[9] Image Captioning Based on Deep Neural Networks

Three deep neural network-based image captioning techniques - CNN-CNN based, CNN-RNN based and Reinforcement-based framework - have been discusses by the authors in this paper. Both the encoder and the decoder in a CNN-CNN based architecture are composed of CNN. These networks lack a recurrent function and are feed-forward, unlike RNNs. In CNN-RNN based framework, CNN enables embedding a fixed-length vector which will help to produce a rich representation of input image which can be utilised in many different applications like object detection, segmentation and recognition. CNN will therefore be employed commonly for encoder-decoder based image captioning techniques. RNN performs better when compared to more in-depth linguistic knowledge, like syntax and semantic information in word sequence, and RNN has greater training capability. Historical information may be obtained using RNN by using continuous circulation of the hidden layer. The employed Multimodal Recurrent Neural Network, also known as m-RNN, cleverly brings together CNN and RNN to resolve the image captioning issue. The generative model (RNN) will communicate with the outside world by integrating reinforcement learning with image captioning. Sequence generation can be used to predict the following word in the series. This modifies it's internal state. The agent is rewarded when it reaches the end of the sequence. The RNN decoder functions as a stochastic policy, picking an action that corresponds to producing the subsequent word.

[10] Show and Tell: A Neural Image Caption Generator

In this paper, the authors offer a generative model that integrates recent developments in computer vision and machine translation to produce natural sentences that describe an image. The model is based on a deep recurrent architecture. A deep Convolution Neural Network (CNN) has been used in place of the encoder Recurrent Neural Network (RNN). This model yielded a BLEU Score of 28 on the Flickr30k dataset.

[11] Image Captioning: Transforming Objects into Words

In this paper, the authors have proposed a Transformer encoder-decoder architecture called Object Relation Transformer which is a modification of the conventional Transformer .It demonstrates the use of object spatial relationship modeling for image captioning.In this model the Faster-RCNN with ResNet-101 is used as the base CNN that acts as the object detector .This object detector extracts the appearances and geometric features from all the detected objects in the image.Then the Object Relation Transformer generates a suitable caption using feature vectors obtained from the object detector as input.

[29] **LSTM-VGG-16: A Novel and Modular Model for Image Captioning Using Deep Learning Approaches**

In this paper the authors have proposed the approach of CNN-LSTM based model to solve the challenge of image captioning. The data set used was Flikr8k Here in this the authors have used Visual Geometry Group(VGG)-16 to extract the features of the images which are than fed along with their captions to LSTM based model for training. This approach has brought revolution in the field of image captioning problem. This involves both Deep Learning and Natural language processing technologies. Using these networks various different methods are developed to perform image captioning in various different domains. For caption generation the authors have use RNN model that is LSTM which can retain important information over time using its respective gates. The evaluation is done for all the test dataset using standard cost function. BiLingual Evaluation Understudy was used for evaluating the generated captions against different actual captions.

1.4 Problem definition

Images include a huge amount of data, including information about the various items and how they relate to other objects. Humans posses the power to summarize a picture using one or more caption. Our aim is to develop a deep learning algorithm to generate a caption for a given image based on the objects present in it.

2 PROPOSED SYSTEM

2.1 Dataset Description

Flickr30k dataset consists of 30k images. The names of all 30,000 images and the 5 captions that go with them are listed in the file Flickr30k.token.txt. The captions or each image are listed as values in a dictionary, with the name of the image (excluding the .jpg extension) serving as the key.

Table 1: Dataset Description

Dataset Name	**Size**		
	Train	***Valid***	***Test***
Flickr8k	6000	1000	1000
Flickr30k	28000	1000	1000

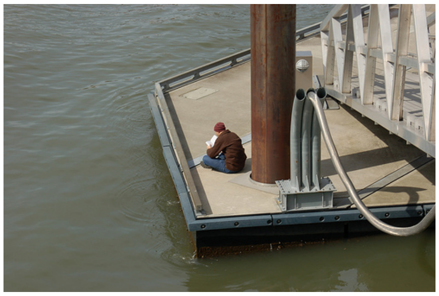


Figure 2: A man is sitting and reading a book.

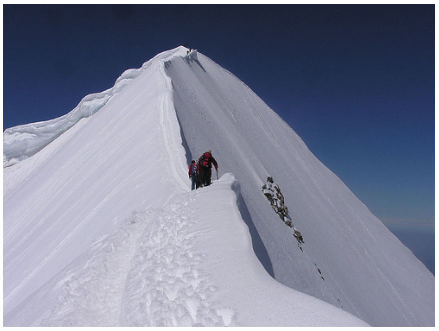


Figure 3: A man is climbing on a snowy mountain.

2.2 Data Pre-processing

From the inferences through our Exploratory Data Analysis (EDA), the conclusions on the dataset were as follows:
The text file,'captions.txt' present in the dataset doesn't contain the tokens that determine the start and end of the caption.
Pre-processing methods taken into action:
Generate a new text file called descriptions.txt by loading the tokens present with captions.txt and adding two additional tokens 'start' at the beginning and 'end' at the end of the caption.

2.3 Initial Approach

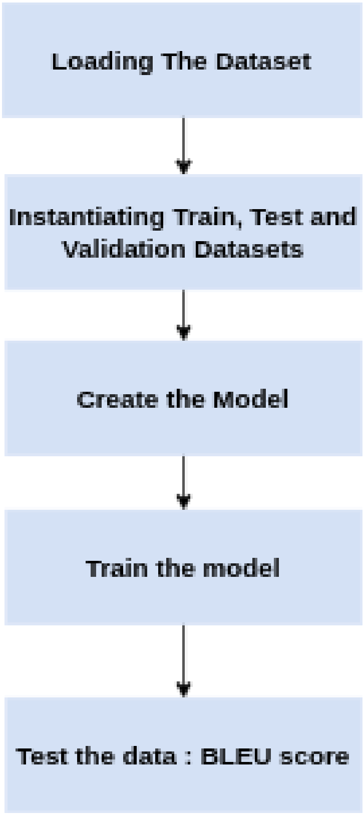

Loading The Dataset
Instantiating Train, Test and Validation Datasets
Create the Model
Train the model
Test the data : BLEU score

Figure 4: Flow process for methodology

2.4 Proposed methodology

2.4.1 Proposed model 1: VGG16 with LSTM Model

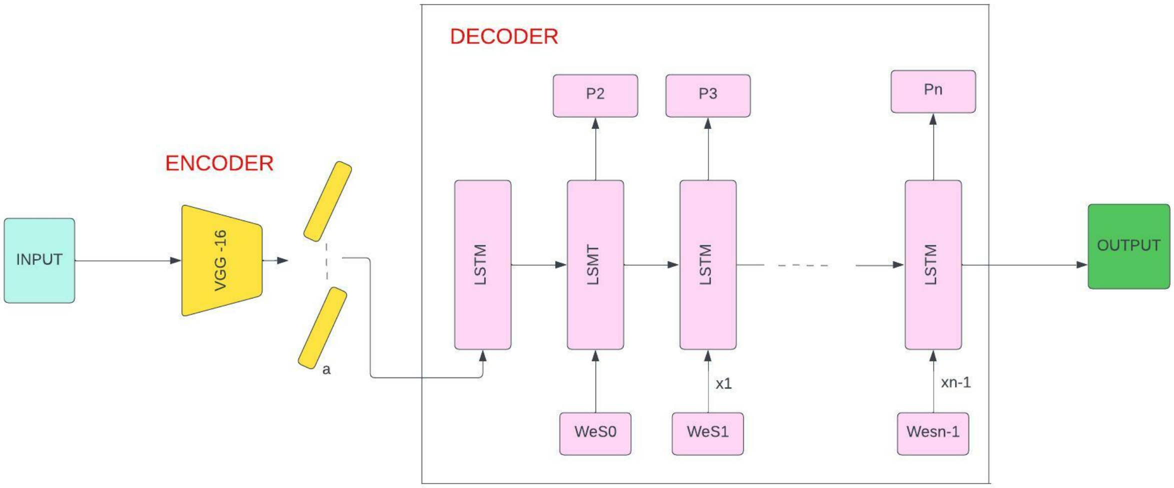


Figure 5: VGG16 with LSTM Architecture

Visual Geometry Group 16(VGG16) with Long Short-Term Memory(LSTM) Model was implemented at first where the VGG16 layers were used to extract the features from the images . The features extracted were then given to the LSTM layers which generated the captions for the images.

2.4.2 Proposed model 2: Xception with LSTM

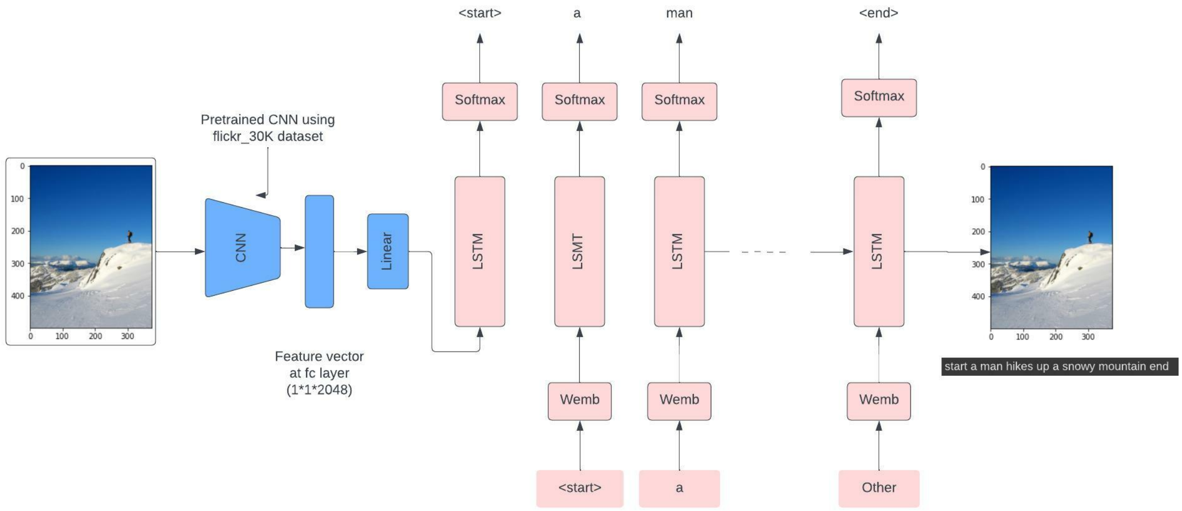


Figure 6: Architecture of Xception with LSTM

In this proposed work, we are implementing a model for image captioning using Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM). This is called the CNN LSTM model, designed for caption generation problems with spatial inputs like images. Given an input image, the model should output a sequence of words or sentence as a caption to the image. The models requires both CNN and LSTM for prediction of the sequences. This is due to the fact that CNN can extract the features and nuances of the input image, whilst LSTM can translate the features and objects given by the image into a natural sentence.

3 IMPLEMENTATION

3.1 Xception model architecture

3.1.1 Entry Flow

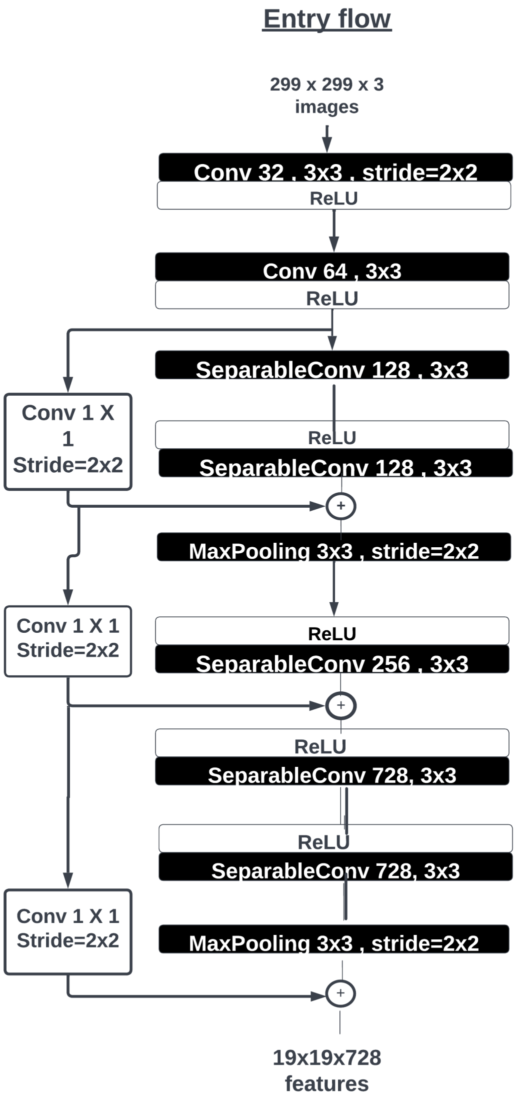


Figure 7: Entry Flow

Entry Flow: The entry flow of the Xception model begins with a standard convolutional layer that processes the input image as shown in Fig. 7. This is followed by a series of depthwise separable convolutional layers, which apply a single filter to each input channel, followed by a pointwise convolutional layer that combines the output of the depthwise convolutional layers. This process is repeated multiple times, with each subsequent block of depthwise separable convolutions increasing the number of filters applied to the input. The output of the entry flow is then passed through a series of additional convolutional layers, pooling layers, and fully connected layers before being fed into the final output layer of the model. This output layer uses a softmax activation function to produce a probability distribution over the possible classes for the input image, allowing the model to make a prediction about the class of the input image. Overall, the entry flow of the Xception model is designed to efficiently extract high-level features from the input image and pass them on to the later layers of the model for further processing and classification.

3.1.2 Middle Flow

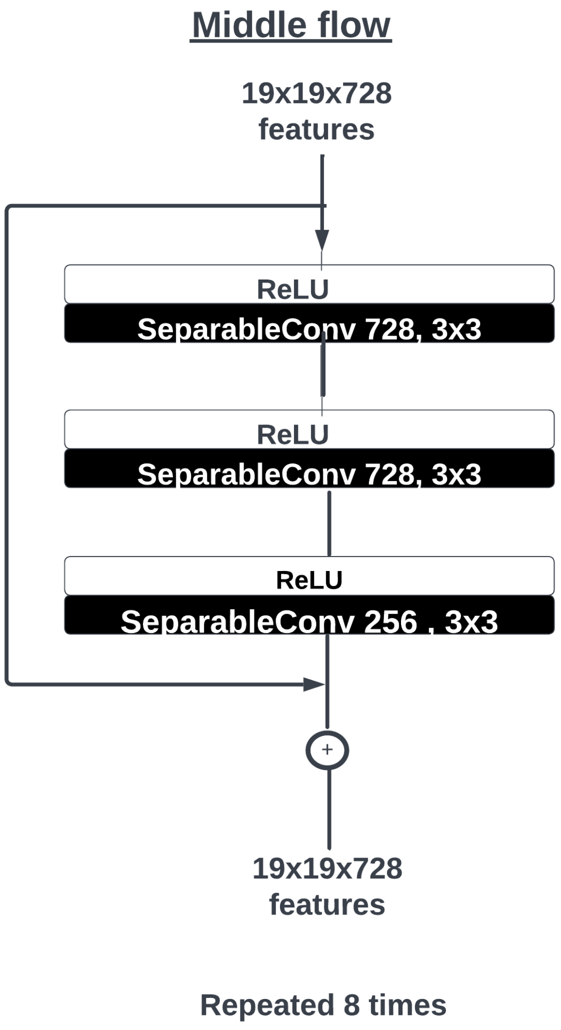


Figure 8: Middle Flow

Middle Flow: Fig. 8 depicts the middle flow of the Xception model that is made up of a series of blocks, each of which consists of a set of depthwise separable convolutional layers followed by a pointwise convolutional layer. As in the entry flow, the depthwise convolutional layers apply a single filter to each input channel, followed by the pointwise convolutional layer which combines the outputs of the depthwise convolutional layers. The number of filters used in each block increases as the input progresses through the middle flow, allowing the model to learn increasingly complex and abstract features from the input image. Additionally, the middle flow includes skip connections, which allow the output of earlier layers in the network to be directly added to the output of later layers, allowing the model to better preserve spatial information and make more accurate predictions. The output of the middle flow is then passed through additional convolutional and pooling layers before being fed into the exit flow of the model. This helps to further refine and abstract the features extracted by the middle flow, preparing them for the final classification step in the model. Overall, the middle flow of the Xception model is designed to efficiently learn increasingly complex and abstract features from the input image, using a combination of depthwise separable convolutions and skip connections to improve performance and accuracy.

3.1.3 Exit Flow

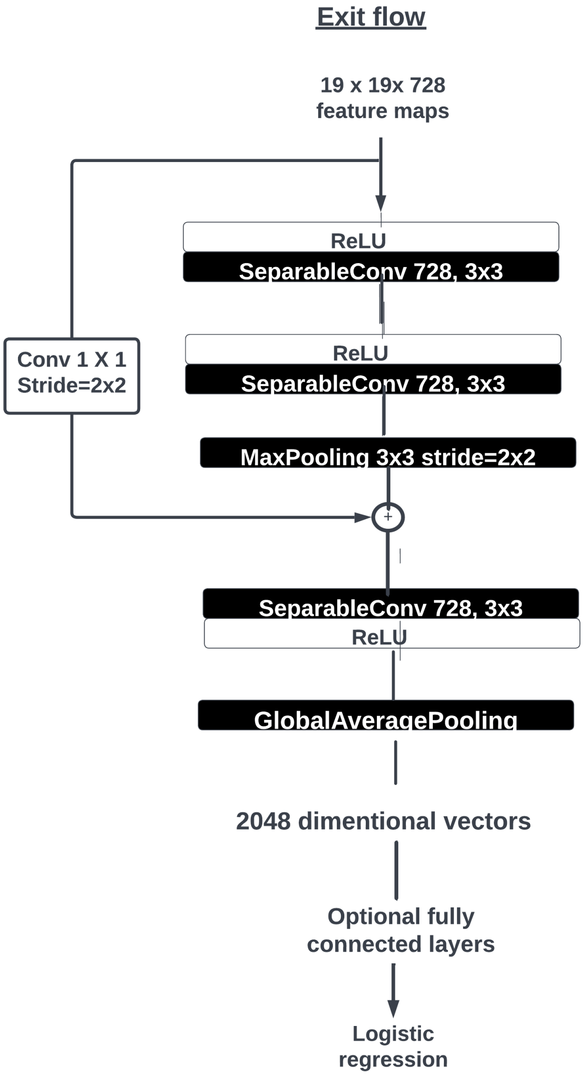
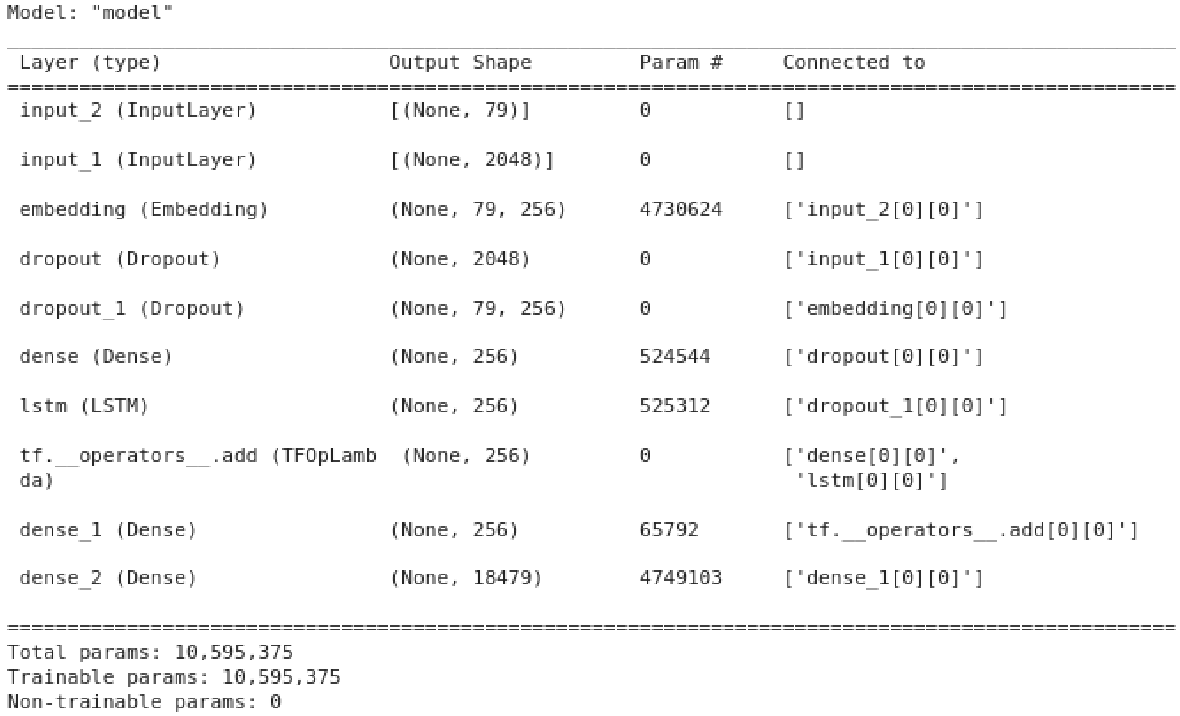


Figure 9: Exit Flow

Exit Flow: As illustrated in Fig. 9, the exit flow of the Xception model begins with a series of convolutional layers, pooling layers, and fully connected layers, which process and abstract the features extracted by the middle flow of the model. This is followed by a final global average pooling layer, which combines the output of the previous layers by taking the average of each feature map. The output of the global average pooling layer is then fed into the final output layer of the model, which uses a softmax activation function to produce a probability distribution over the possible classes for the input image. This allows the model to make a final prediction about the class of the input image, based on the features extracted and abstracted by the earlier layers of the network. Overall, the exit flow of the Xception model is designed to process and abstract the features extracted by the middle flow of the model, and use them to make a final prediction about the class of the input image. This is done using a combination of convolutional, pooling, and fully connected layers, along with a global average pooling layer and a softmax output layer.

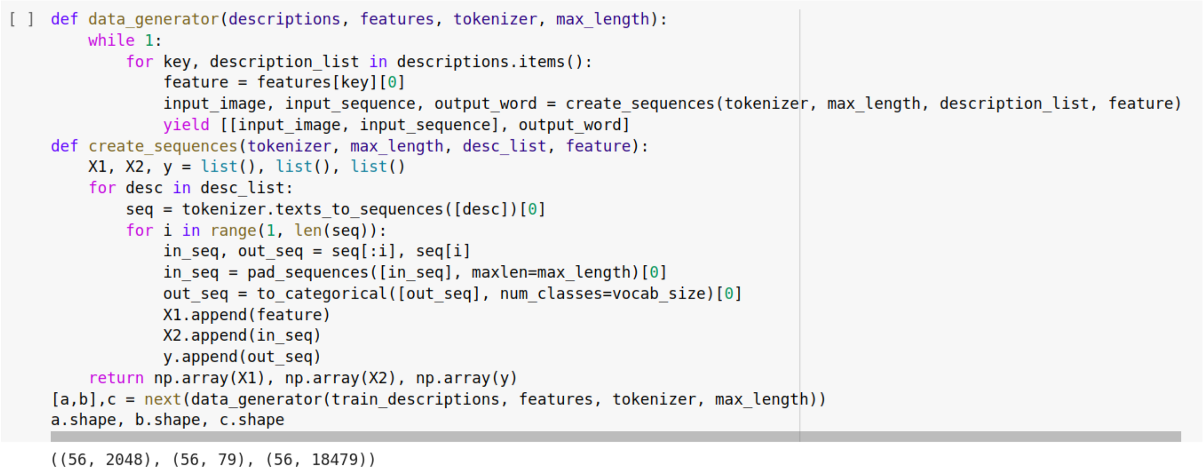
3.2 Proposed model summary



```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_2 (InputLayer)           [(None, 79)]         0           []

 input_1 (InputLayer)           [(None, 2048)]       0           []

 embedding (Embedding)          (None, 79, 256)      4730624     ['input_2[0][0]']

 dropout (Dropout)              (None, 2048)         0           ['input_1[0][0]']

 dropout_1 (Dropout)            (None, 79, 256)      0           ['embedding[0][0]']

 dense (Dense)                  (None, 256)          524544      ['dropout[0][0]']

 lstm (LSTM)                    (None, 256)          525312      ['dropout_1[0][0]']

 tf.__operators__.add (TFOpLamb  (None, 256)         0           ['dense[0][0]',
 da)                                                              'lstm[0][0]']

 dense_1 (Dense)                (None, 256)          65792       ['tf.__operators__.add[0][0]']

 dense_2 (Dense)                (None, 18479)        4749103     ['dense_1[0][0]']

==================================================================================================
Total params: 10,595,375
Trainable params: 10,595,375
Non-trainable params: 0
```

Figure 10: Model summary

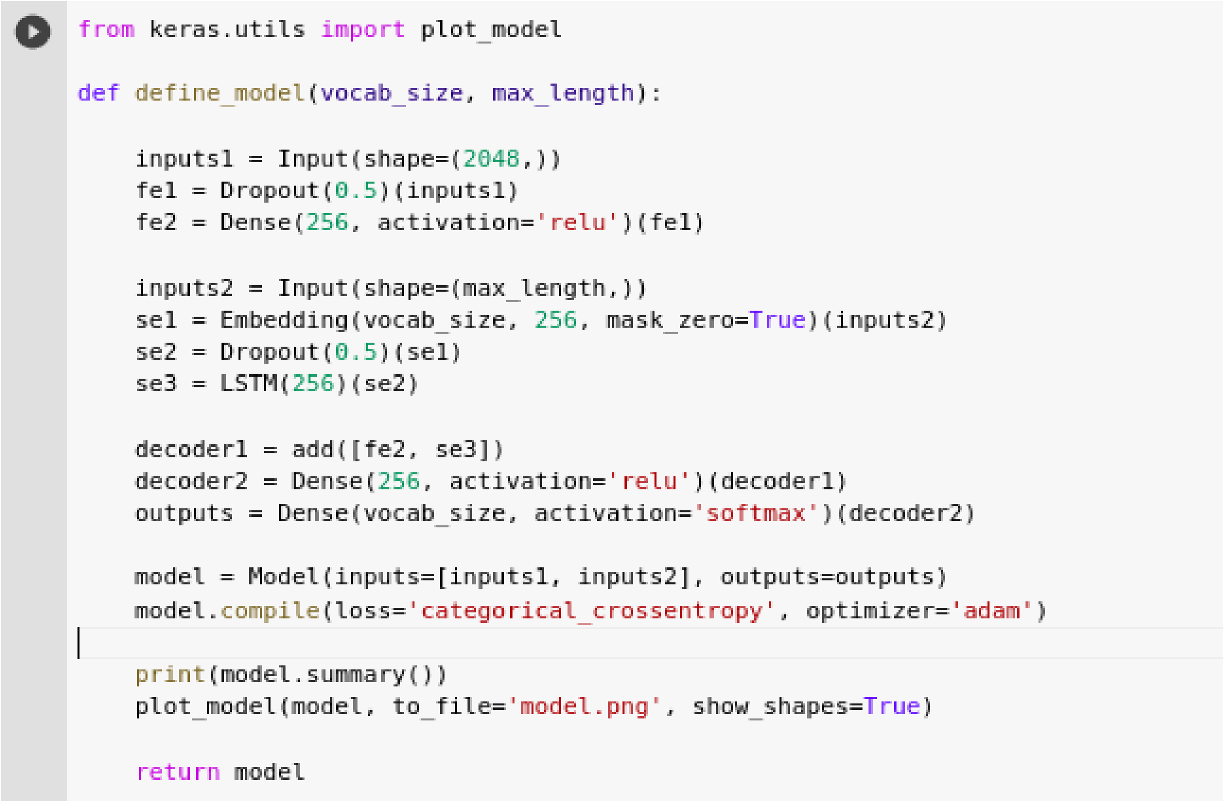
3.3 Code snippets



```
[ ] def data_generator(descriptions, features, tokenizer, max_length):
        while 1:
            for key, description_list in descriptions.items():
                feature = features[key][0]
                input_image, input_sequence, output_word = create_sequences(tokenizer, max_length, description_list, feature)
                yield [[input_image, input_sequence], output_word]
    def create_sequences(tokenizer, max_length, desc_list, feature):
        X1, X2, y = list(), list(), list()
        for desc in desc_list:
            seq = tokenizer.texts_to_sequences([desc])[0]
            for i in range(1, len(seq)):
                in_seq, out_seq = seq[:i], seq[i]
                in_seq = pad_sequences([in_seq], maxlen=max_length)[0]
                out_seq = to_categorical([out_seq], num_classes=vocab_size)[0]
                X1.append(feature)
                X2.append(in_seq)
                y.append(out_seq)
        return np.array(X1), np.array(X2), np.array(y)
    [a,b],c = next(data_generator(train_descriptions, features, tokenizer, max_length))
    a.shape, b.shape, c.shape
```

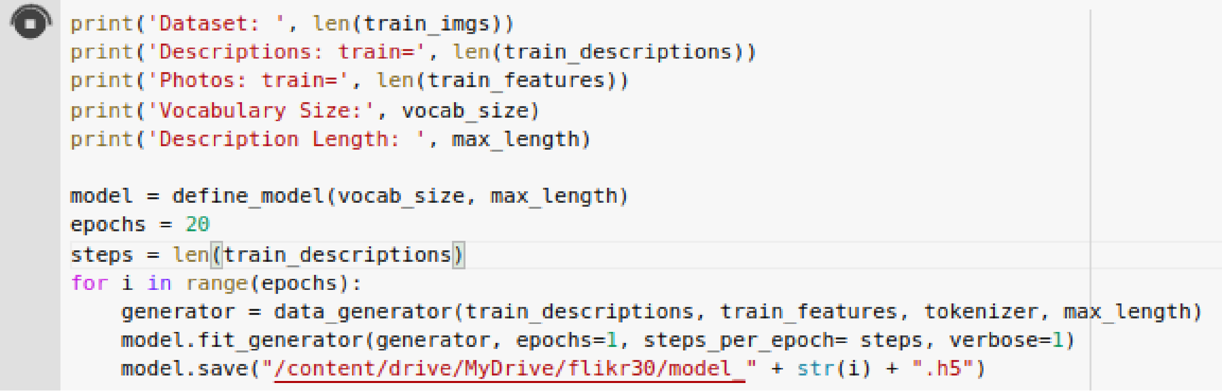
```
((56, 2048), (56, 79), (56, 18479))
```

Figure 11: Function



```
from keras.utils import plot_model

def define_model(vocab_size, max_length):

    inputs1 = Input(shape=(2048,))
    fe1 = Dropout(0.5)(inputs1)
    fe2 = Dense(256, activation='relu')(fe1)

    inputs2 = Input(shape=(max_length,))
    se1 = Embedding(vocab_size, 256, mask_zero=True)(inputs2)
    se2 = Dropout(0.5)(se1)
    se3 = LSTM(256)(se2)

    decoder1 = add([fe2, se3])
    decoder2 = Dense(256, activation='relu')(decoder1)
    outputs = Dense(vocab_size, activation='softmax')(decoder2)

    model = Model(inputs=[inputs1, inputs2], outputs=outputs)
    model.compile(loss='categorical_crossentropy', optimizer='adam')

    print(model.summary())
    plot_model(model, to_file='model.png', show_shapes=True)

    return model
```

Figure 12: LSTM training model



```
print('Dataset: ', len(train_imgs))
print('Descriptions: train=', len(train_descriptions))
print('Photos: train=', len(train_features))
print('Vocabulary Size:', vocab_size)
print('Description Length: ', max_length)

model = define_model(vocab_size, max_length)
epochs = 20
steps = len(train_descriptions)
for i in range(epochs):
    generator = data_generator(train_descriptions, train_features, tokenizer, max_length)
    model.fit_generator(generator, epochs=1, steps_per_epoch= steps, verbose=1)
    model.save("/content/drive/MyDrive/flikr30/model_" + str(i) + ".h5")
```

Figure 13: Training

4 RESULTS AND DISCUSSION

4.1 CNN-LSTM model evaluation

Bilingual Evaluation Understudy (BLEU) is an algorithm that has been used for evaluating the quality of a machine-translated text. BLEU is easy to understand and compute. BLEU is a language independent algorithm that lies between 0 and 1. Higher the BLEU score better the quality of the caption. To compute the BLEU score for predicted caption concerns for actual captions first, convert the predicted caption and references to unigram/bigrams. Then modified n-gram precision is calculated from the following formula.

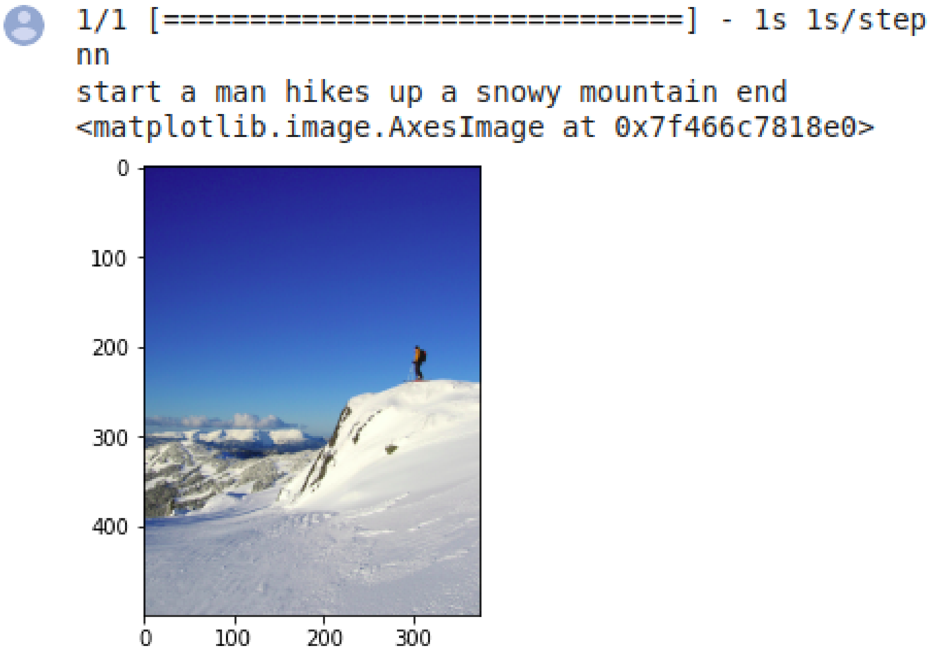
BLEU tells how good our predicted caption is compared to the provided reference captions.

Table 2: BLEU Scores

Model	**Training Dataset**	**Test Size**	**BLEU-1 Score**
VGG-LSTM	Flickr8k	1000	0.128326
Xception-LSTM (Ours)	Flickr8k	1000	**0.231245**
Xception-LSTM (Ours)	Flickr30k	1000	**0.318762**

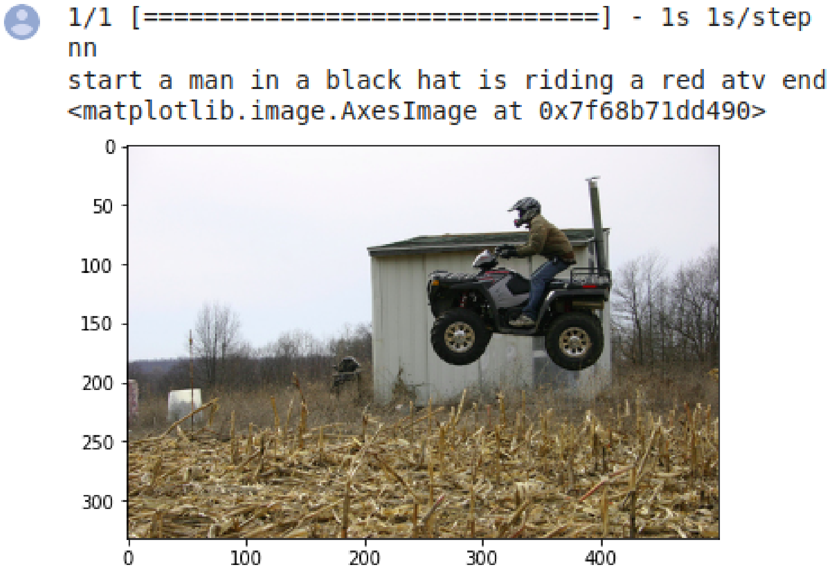
4.2 Model Predictions

The proposed model was tested against various input images and the output captions for each image is as shown in Fig.14, Fig.15, Fig.16 and Fig.17.



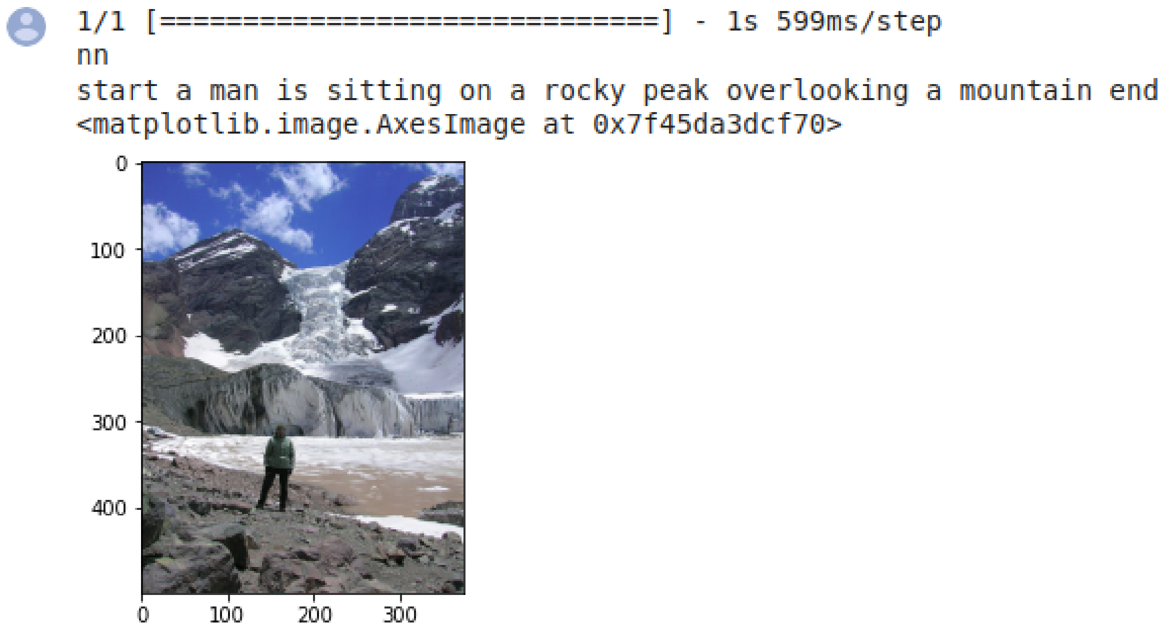
```
1/1 [==============================] - 1s 1s/step
nn
start a man hikes up a snowy mountain end
<matplotlib.image.AxesImage at 0x7f466c7818e0>
```

Figure 14: Produced caption for the provided test image



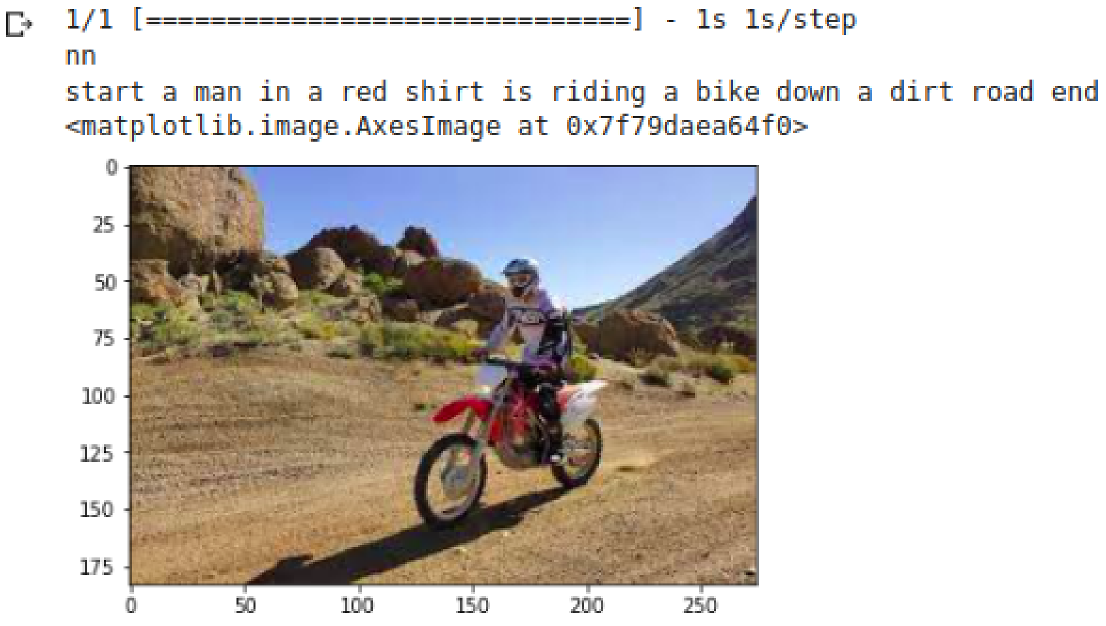
```
1/1 [==============================] - 1s 1s/step
nn
start a man in a black hat is riding a red atv end
<matplotlib.image.AxesImage at 0x7f68b71dd490>
```

Figure 15: Produced caption for the provided test image



```
1/1 [==============================] - 1s 599ms/step
nn
start a man is sitting on a rocky peak overlooking a mountain end
<matplotlib.image.AxesImage at 0x7f45da3dcf70>
```

Figure 16: Produced caption for the provided test image



```
1/1 [==============================] - 1s 1s/step
nn
start a man in a red shirt is riding a bike down a dirt road end
<matplotlib.image.AxesImage at 0x7f79daea64f0>
```

Figure 17: Produced caption for the provided test image

CONCLUSION

In this paper, a deep learning model ”CNN-LSTM” is proposed for image captioning. This model has been trained on Flickr30k dataset. This model is an entire neural network system that has the ability to analyse images automatically and produce an appropriate English caption. The model is built on the Xception convolution neural network architecture, which compresses an image into a small representation (feature extraction), and the LSTM, a recurrent neural network, which uses the extracted features from the CNN to produce a relevant sentence. The model is trained to increase the efficiency with which it generates an appropriate caption for the given image. When we use the Graphics Processing Unit to run this model, it operates effectively. The image captioning deep learning model can be further converted into an application to guide people blind people or people with low vision.

References

[1] Kulkarni, Uday, S. M. Meena, Sunil V. Gurlahosur, and Gopal Bhogar. ”Quantization Friendly MobileNet (QF-MobileNet) architecture for vision based applications on embedded platforms.” Neural Networks (2020).

[2] U. Kulkarni, S. M. Meena, S. V. Gurlahosur and U. Mudengudi, ”Classification of Cultural Heritage Sites Using Transfer Learning,” 2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM), 2019, pp. 391-397, doi: 10.1109/BigMM.2019.00020.

[3] U. Kulkarni, S. M. Meena, P. Joshua, K. Rodrigues and S. V. Gurlahosur, ”Integrated Crowdsourcing Framework Using Deep Learning for Digitalization of Indian Heritage Infrastructure,” 2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM), New Delhi, India, 2020, pp. 200-208, doi: 10.1109/BigMM50055.2020.00036.

[4] Sreekanth, P., Kulkarni, U., Shetty, S., and Meena, S. M. (2019). Head pose estimation using transfer learning. Paper presented at the Proceedings of the 2018 International Conference on Recent Trends in Advanced Computing, ICRTAC-CPS 2018, 73-79. doi:10.1109/ICRTAC.2018.8679209

[5] Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.

[6] Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 2015, 28, 1–14.

[7] Ralf C. Staudemeyer and Eric Rothstein Morris: Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks

[8] Muhammad Abdelhadie Al-Malla, Assef Jafar and Nada Ghneim: Image captioning model using attention and object features to mimic human image understanding

[9] Shuang Liu1, Liang Bai1,a, Yanli Hu1 and Haoran Wang1: Image Captioning Based on Deep Neural Networks

[10] Oriol Vinyals, Alexander Toshev, Samy Bengio and Dumitru Erhan: Show and Tell: A Neural Image Caption Generator

[11] Simao Herdade, Armin Kappeler, Kofi Boakye and Joao Soares: Image Captioning: Transforming Objects into Words

[12] Francois Chollet: Xception: Deep Learning with Depthwise Separable Convolutions

[13] Karen Simonyan Andrew Zisserman: VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

[14] Łukasz Kaiser, Aidan N. Gomez and François Chollet: DEPTHWISE SEPARABLE CONVOLUTIONS FOR NEURAL MACHINE TRANSLATION

[15] Jiahui Tao et. al.: Research on vgg16 convolutional neural network feature classification algorithm based on Transfer Learning

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014.

[17] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks.

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

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

[20] R. Vedantam, C. L. Zitnick, and D. Parikh. CIDEr: Consensus-based image description evaluation. In arXiv:1411.5726, 2015.

[21] 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.

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

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

[24] 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.

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

[26] 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.

[27] 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.

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

[29] LSTM-VGG-16: A Novel and Modular Model for Image Captioning Using Deep Learning Approaches