Automatic Image Captioning Using Neural Networks

Subash Pandey[1], Rabin Kumar Dhamala[1], Bikram Karki[1], Saroj Dahal[1], and Rama Bastola[1]
[1]Department of Electronics and Computer Engineering, Thapathali Campus, Institute of Engineering,
Tribhuvan University, Kathmandu, Nepal

[*]**Corresponding Email**: tha072bex342@tcioe.edu.np

ABSTRACT

Automatically generating a natural language description of an image is a major challenging task in the field of artificial intelligence. Generating description of an image bring together the fields: Natural Language Processing and Computer Vision. There are two types of approaches i.e. top-down and bottom-up. For this paper, we approached top-down that starts from the image and converts it into the word. Image is passed to Convolutional Neural Network (CNN) encoder and the output from it is fed further to Recurrent Neural Network (RNN) decoder that generates meaningful captions. We generated the image description by passing the real time images from the camera of a smartphone as well as tested with the test images from the dataset. To evaluate the model performance, we used BLEU (Bilingual Evaluation Understudy) score and match predicted words to the original caption.

Keywords: CNN, Image Captioning, Image Description, LSTM, RNN

1. INTRODUCTION

Automatically generating image captions have practical applications and have great impact in the society. For instance, helping visually impaired people for better understanding the content of images, cost and extensive human effort minimization of labeling millions of images that are uploaded on the internet daily. Generating meaningful description of an image requires high level image classification and object detection. It connects the computer vision with natural language which are two major field of artificial intelligence.

A quick glance at an image is enough for humans to describe an image. Same cannot be said about the machines. Visual description is challenging because it requires recognizing not only objects, but other visual elements such as actions and attributes, and constructing fluent sentence describing how objects, action and attributes are related in an image. Similarly converting text into audio format has been studied and applied by many professionals. Mixing of these ideas has not been properly realized. This project tries to connect these dots and tries to find new area for deep learning to be used in.

The manual annotation by humans to such humungous data is extremely difficult for information extraction. The creation of image and video data has grown exponentially during the past few decades. In the present context, Deep learning has a very good success rate in image classification and object detection. [1] Thus, this can be further extended for the generation of relevant captions for those images which can have a wide variety of applications. [2]

We can also use it in social media platforms for automatic generation of captions from the users' uploaded images. This makes fun and interactive which can appeal new user to the site. It can be used as a useful tool for real time crime monitoring and surveillance. The relevant suspicious captions can be made to fire-up alarms and inform the crime department to take corrective actions immediately. A handy tool for robot navigation and other computer vision applications like self-driving car.

2. RELATED WORK

There are various works that have been done related to this project. The two general paradigms in existing image captioning approaches, the first one, top-down paradigm starts from gist of an image and then converts it into words while bottom-up paradigm first comes up with words describing various aspects of an image and then combines them. This image captioning algorithm is based upon a novel semantic attention model which combines the visual information in both bottom-up and top-down approaches in the framework of recurrent neural network. The intermediate filter responses from a classification Convolutional Neural Network (CNN) is used to build a global visual description and a set of attribute detectors are run to get a list of visual attributes that are most likely to appear in the image which then corresponds to an entry in the vocabulary set or dictionary. [3]

This approach finds a set of k nearest images. The images which are "nearest" are examined using GIST, pre-trained deep features and deep features fine-tuned for the task of caption generation. Once a set of k NN images are found, the captions describing these images are combined into a set of candidate captions from which the final caption is generated. The best candidate caption is generated by finding the one that scores the highest with respect to other candidate captions which is referred as "consensus" caption. The scores between pairs of captions are computed using either the Consensus-based Image Description Evaluation (CIDEr) or Bilingual Evaluation Understudy (BLEU) metric. [4]

The architecture of a typical multi-model space-based method contains a language encoder part, a vision part, a multimodal space part and a language decoder part. The vision part uses a deep convolutional neural network for feature extraction. The language encoder extracts the word features and learns a dense feature embedding for each word. The semantic temporal context is then forwarded to the recurrent layers and the multimodal space part maps the image features into a common space with the word features. The resulting map is then passed to the language decoder which generates captions by decoding the map. [5]

A pre-trained captioning model can hardly be applied to a new domain in which some novel object (the objects and the description words unseen during model training categories exists. In novel object captioning, the machine generates descriptions without extra training the sentences about the novel object. A Decoupled Novel Object Captioner (DNOC) framework fully decouples the language sequence model from the object descriptions. A Sequence Model with the Placeholder (SM-P) generates a sentence containing placeholders representing unseen novel objects. A key-value object memory built upon the freely available detection model contains the visual information and corresponding word for each object. A query generated from SM-P is used to retrieve the words from the object memory and the placeholder further filled with correct word results in a caption with novel object descriptions. [6]

3. METHODOLOGY

3.1. DATASET

We use Microsoft Common Objects in Context (MSCOCO) images and text data to build our datasets. MSCOCO datasets have rich image datasets and each image has at least five captions. The datasets contain

82783 training datasets and 40775 tests datasets. We download all the images and saved it on google drive for further processing in project.

3.2. IMAGE FEATURE

1) Image Processing: The image files are saved on google drive and we load the images on our working environment i.e google colab. In the image preprocessing, we convert the image into finite shape and size. The image is decoded into 3 channels because all the images must have 3 channels

2) Feature Extraction: In the automatic description generation of image, the main feature is object content in our image. For extracting features from image, we use a pre-trained model architecture of VGG16 and we remove the last fully connected layer and after that the output of the layer of VGG16 model is the main feature of our project.

3) Encoder: When image is passed to CNN layer, we get the encoded form of image. CNN can produce a representation of input image by embedding it into fixed-length vector. Such a representation can be used in a variety of computer vision tasks. [7]

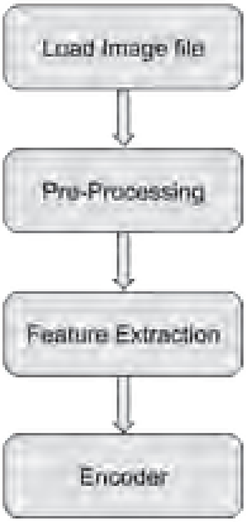


Figure 1: Image feature extraction and encoding

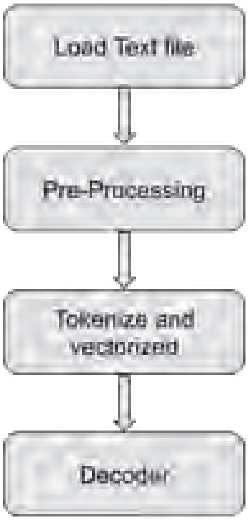


Figure 2**:** Text processing and decoding

3.3. TEXT PROCESSING

1) Text Processing: In text preprocessing, we clean the unwanted data. We remove the special character/token contained in the caption and we also eliminate a combination of text and number like ‘subarna124’. All the words in the caption are lowercased.

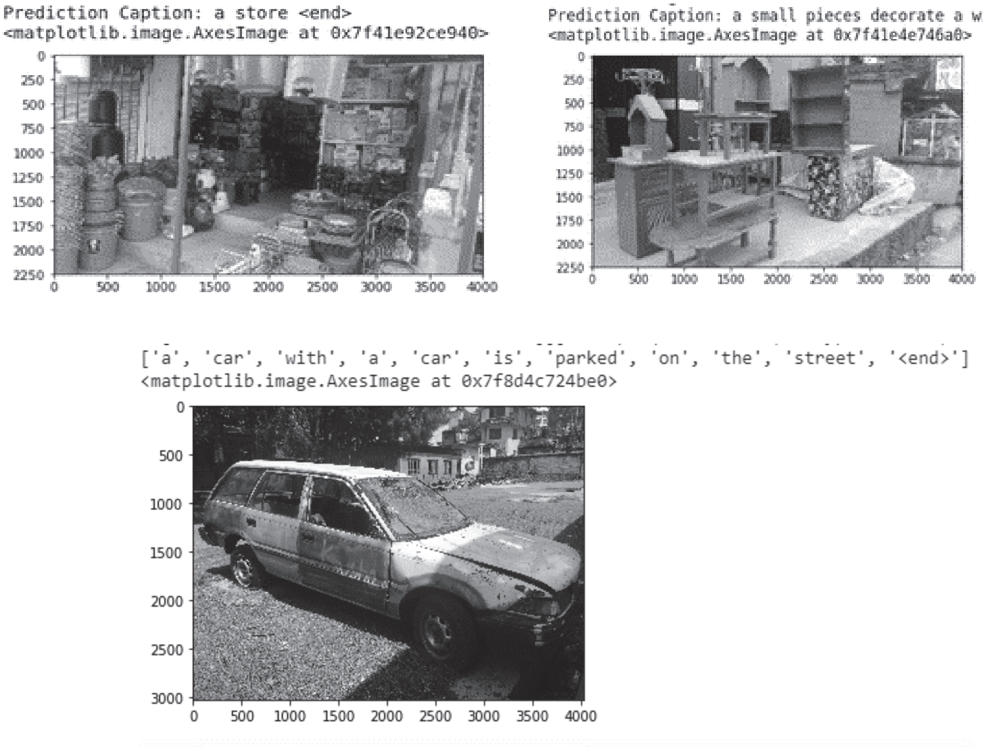
2) Tokenize and Vectorize: We split text of caption by the space and get all unique words from each caption. We have to limit the vocabulary size to save the memory size. We add two tokens at each of caption ‘start’ and ‘end’. All the tokens of caption are vectorized.

3) Decoder: LSTM (Long Short-Term Memory) network is used as a decoder to give a decoded RNN (Recurrent Neural Network) <start> token indicates the start the sentence and <end> token indicates the end of the sentence. The model learns to predict the captions which are the target variables. The captions are predicted word by word. So, each word is encoded into a fixed size vector. [8] [9]

4. EXPERIMENT

4.1. OUTPUT

All the tasks have been performed in high neural network API called keras running on top of tensorflow. For the training of our dataset we used google colab. It is a free cloud service and provides free GPU which is enough to train our datasets. The output encoded image from the CNN encoder was fed to the RNN decoder and it decodes the encoded image information one word at a time to generate an output. We trained our datasets at desirable epochs like 50 or100 epochs and for the result, we passed the real time image from the camera of a smartphone and passed to the CNN encoder fig. 1.



```
Prediction Caption: a store <end>
<matplotlib.image.AxesImage at 0x7f41e92ce940>
```

```
Prediction Caption: a small pieces decorate a window <end>
<matplotlib.image.AxesImage at 0x7f41e4e746a0>
```

```
['a', 'car', 'with', 'a', 'car', 'is', 'parked', 'on', 'the', 'street', '<end>']
<matplotlib.image.AxesImage at 0x7f8d4c724be0>
```

Figure 3: Some output images with their predicted captions

4.2. ANALYSIS

For the analysis of our working model, we used BLEU score and matched predicted words to their original caption. While training part we observed that on increasing the number of training epochs the loss gradually decreases as demonstrated in fig. 4. If we were able to train our datasets for larger number of epochs, then our model would be able to generate better description. For comparative study of our model we only trained for 50 epochs.

BLEU is a score that compares text to one or more reference text(s). This score is used to find the accuracy between the generated text to the original text. Fig. 5, shows the graphical representation of BLEU score of the generated text of one image per epoch which is calculated as an average of scores evaluated by comparing predicted caption with 5 other original captions. The graph shows that when the number of training epochs increases the average value of BLEU score also gradually increases.

Match words is also a score that counts the matched words from the generated text to the original captions and divides by the total number of words in the generated text of an image. As we can see in fig. 7, we found the percentage of matched words of generated description of an image to each caption, calculating the average value of total match words of the caption per epoch. As number of training epochs increases the average match words to the original caption also increases.

As we compare BLEU score and match words in each epoch, we saw the graph as shown in the fig. 8. Our model is trying to generate words almost contained in the original caption.

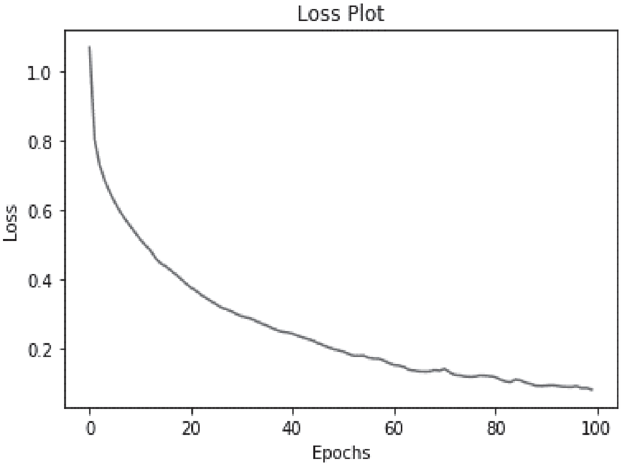


Figure 4: Loss plot vs. Epochs

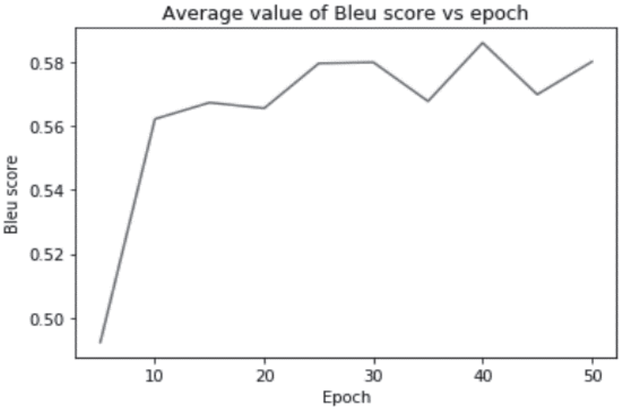
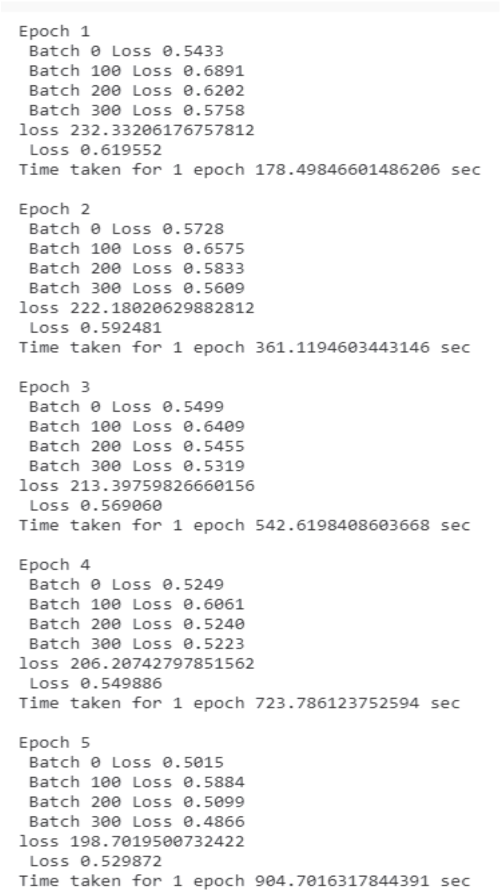


Figure 5: Average BLEU score vs. Epoch



```
Epoch 1
 Batch 0 Loss 0.5433
 Batch 100 Loss 0.6891
 Batch 200 Loss 0.6202
 Batch 300 Loss 0.5758
loss 232.33206176757812
 Loss 0.619552
Time taken for 1 epoch 178.49846601486206 sec

Epoch 2
 Batch 0 Loss 0.5728
 Batch 100 Loss 0.6575
 Batch 200 Loss 0.5833
 Batch 300 Loss 0.5609
loss 222.18020629882812
 Loss 0.592481
Time taken for 1 epoch 361.1194603443146 sec

Epoch 3
 Batch 0 Loss 0.5499
 Batch 100 Loss 0.6409
 Batch 200 Loss 0.5455
 Batch 300 Loss 0.5319
loss 213.39759826660156
 Loss 0.569060
Time taken for 1 epoch 542.6198408603668 sec

Epoch 4
 Batch 0 Loss 0.5249
 Batch 100 Loss 0.6061
 Batch 200 Loss 0.5240
 Batch 300 Loss 0.5223
loss 206.20742797851562
 Loss 0.549886
Time taken for 1 epoch 723.786123752594 sec

Epoch 5
 Batch 0 Loss 0.5015
 Batch 100 Loss 0.5884
 Batch 200 Loss 0.5099
 Batch 300 Loss 0.4866
loss 198.7019500732422
 Loss 0.529872
Time taken for 1 epoch 904.7016317844391 sec
```

Figure 6: Losses and execution time during each epoch

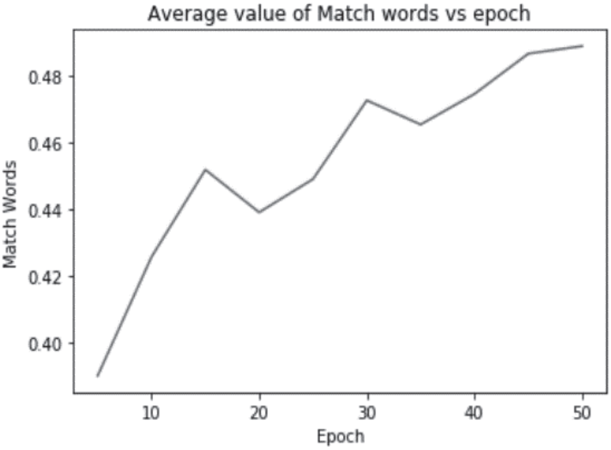


Figure 7: Average value of match words Vs Epoch

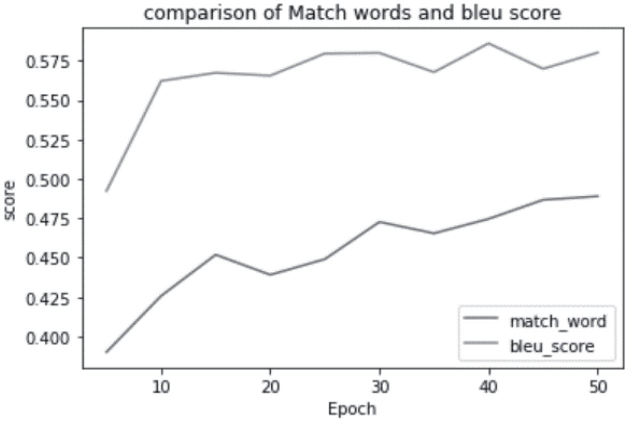


Figure 8: Match words Vs BLEU score

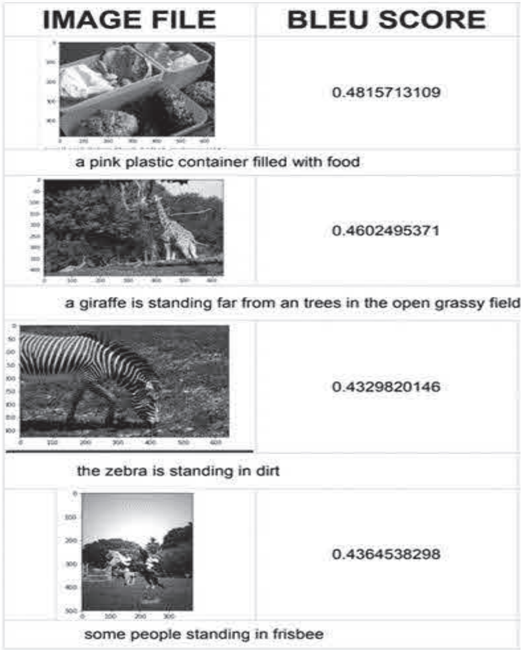
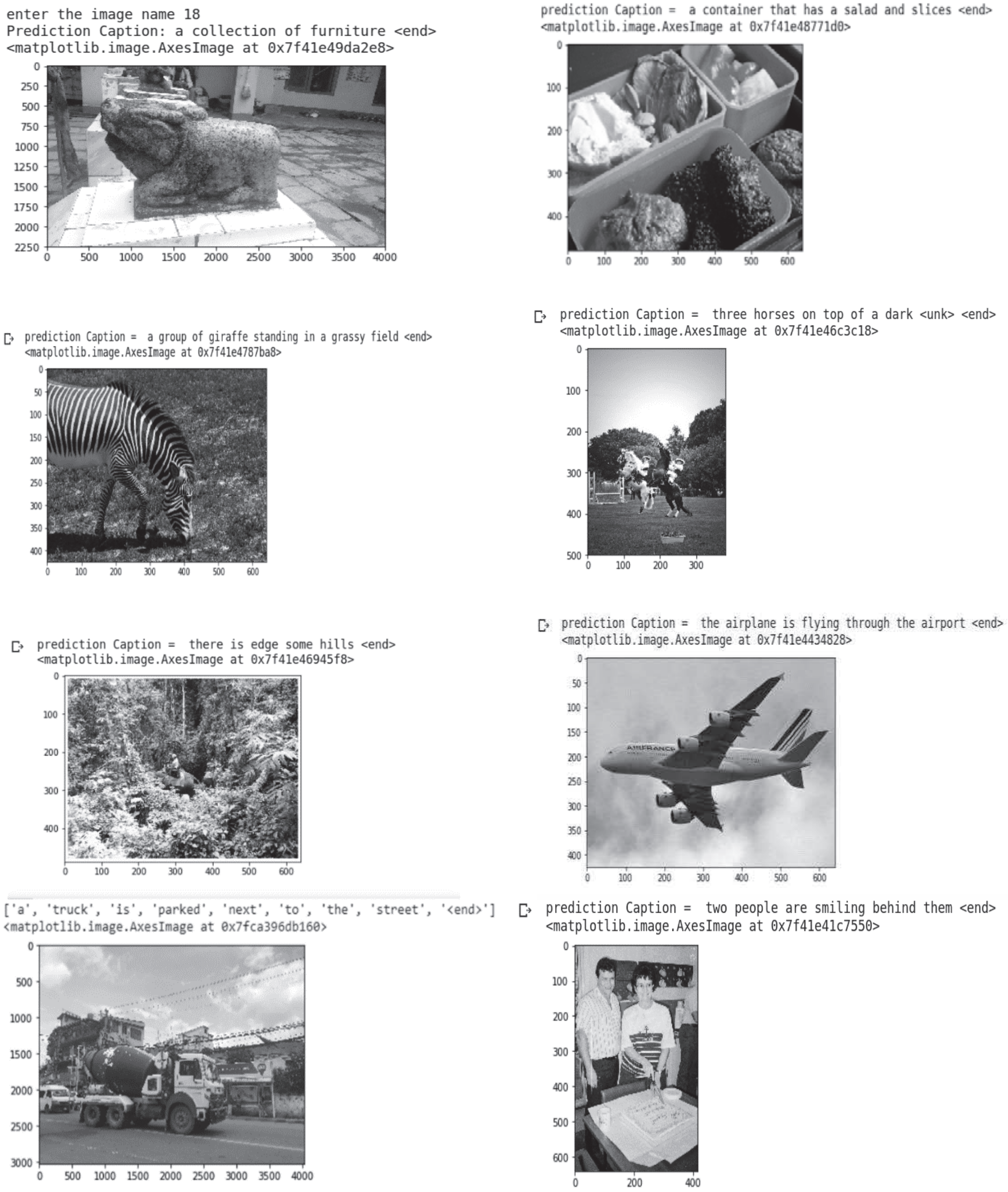


IMAGE FILE	BLEU SCORE
	0.4815713109
a pink plastic container filled with food	
	0.4602495371
a giraffe is standing far from an trees in the open grassy field	
	0.4329820146
the zebra is standing in dirt	
	0.4364538298
some people standing in frisbee	

Figure 9: BLEU score evaluation on different images

5. CONCLUSION AND FUTURE WORK

From this research, we created a system where we can pass the real time image from the camera of a smartphone into the google drive where all the works are being done. We are able to generate the relevant captions describing the scenario in a real-time image. We got BLEU score above 45% and match score above 50%. To make our system more accurate, we have to work on our datasets and increase the number as much as possible and we also have to train our datasets for larger number of epochs. For all these works, we need high GPU training platforms but currently we were not able to access one. It has many applications in Artificial Intelligence, Computer Vision, social media platforms, real-time surveillance etc. We are also working on building a user-friendly interface that normal users can use easily. We find the idea of this project extremely applicable and interesting if can be brought in real use which can be proved as an epitome for artificial intelligence.



```
enter the image name 18
Prediction Caption: a collection of furniture <end>
<matplotlib.image.AxesImage at 0x7f41e49da2e8>
```

```
prediction Caption =  a container that has a salad and slices <end>
<matplotlib.image.AxesImage at 0x7f41e48771d0>
```

```
prediction Caption =  a group of giraffe standing in a grassy field <end>
<matplotlib.image.AxesImage at 0x7f41e4787ba8>
```

```
prediction Caption =  three horses on top of a dark <unk> <end>
<matplotlib.image.AxesImage at 0x7f41e46c3c18>
```

```
prediction Caption =  there is edge some hills <end>
<matplotlib.image.AxesImage at 0x7f41e46945f8>
```

```
prediction Caption =  the airplane is flying through the airport <end>
<matplotlib.image.AxesImage at 0x7f41e4434828>
```

```
['a', 'truck', 'is', 'parked', 'next', 'to', 'the', 'street', '<end>']
<matplotlib.image.AxesImage at 0x7fca396db160>
```

```
prediction Caption =  two people are smiling behind them <end>
<matplotlib.image.AxesImage at 0x7f41e41c7550>
```

Figure 10: Some other results with the predicted captions on test images

REFERENCES

[1] N. H. V. G. R. Manoj krishna, "Image Classification using Deep Learning," International Journal of Engineering & Technology, 2018.

[2] R. Singh and A. Sharma , "Image Captioning using Deep Neural Networks," 2018.

[3] Y. Quanzeng, J. Hailin, W. Zhaowen, F. Chen and L. Jiebo, "Image Captioning with semantic attention," IEEE, 2015.

[4] J. Devlin, S. Gupta, R. Girshick, M. Mitchell and L. C. Zitnick, "Explaining Nearest Neighbour Approach for Image Captioning".

[5] Z. M. Hossain, F. Sohel, F. M. Shiratuddin and H. Laga, "A Comprehensive Survey of Deep Learning for Image Captioning," 2018.

[6] Y. Wu, L. Zhu, L. Jiang and Y. Yang, "Decoupled Novel Object Captioner," 2018.

[7] M. Najman, "Image Captioning with Convolution Neural Networks," Prague, 2017.

[8] F. Shaikh, "Automatic Image Captioning using Deep Learning (CNN and LSTM) in PyTorch," 2018.

[9] J. Mao, X. Wei, Y. Yang, J. Wang, Z. Huang and A. Yuille, "Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN)," arXiv, 2014.