# Image Captioning Using Deep Learning

Sanjana Annamaneni[1], Naveen Kumar[2], Kiran Noolvi[3], Sai Krishna[4]
Department of Computer Science
University of Texas at Dallas, Dallas, TX.
[1]sanjana.annamaneni@utdallas.edu, [2]naveenkumarreddy.inaganti@utdallas.edu, [3]kiran.noolvi@utdallas.edu,
[4]saikrishna.chirumamilla@utdallas.edu

***Abstract*—Image Captioning implies automatically describing the contents of an image, this field of research is gaining attention as it stands as a fundamental problem in the field of Machine Learning. In this project we have proposed and developed a model that helps to solve this problem by automatically predicting captions for a given image. Our model implements search algorithms like beam search and argument-max search for the automatically generating the captions for images. We have evaluated the correctness of the generating captioning by using BLEU Score and have achieved an average score of 54.55.**

***Keywords—Image Captioning, Deep Neural Networks, Word-embedding, Convolution Neural Networks, LSTM, Search Algorithms, Flickr8K dataset.***

## I. Introduction

As humans we can quickly see and understand what an image means, but for a machine to interpret an image is not an easy task. But the impact which can be created by a machine by interpreting an image and thus describing its content is very high. Thus, helps to solve problems from the image classification models or object detection models. For example, it can be used by self-driving cars to properly understand the environment around the car, it can be helpful for visually impaired to commute by themselves without waiting for any help. It can also be used by CCTV cameras to understand the situation and send alerts as needed in case of an emergency. It can also be helpful in the field of internet by helping us search an image by describing the content.

To achieve this task of automatically describing the content of an image many methods were proposed in the past years, but the recent advancements made by deep learning has brought about a major change in this field. The past works for image captioning suggested creating models and then combining the results together, but Deep learning came up a sequential solution [1] which helped in the captioning the image in a better and useful way. We have used the deep learning technique for doing the image captioning in this project. We trained our model on several images and captions dataset, later used the model to predict captions for a given image and the returned captions were tested against the ground truth captions already present in the data and we have achieved an average BLEU score of 54.55.

## II. LITERATURE SURVEY

Captioning of images is done by extracting the features from the images or we can tag images objects with specific keywords. The different technical approaches for captioning image simple hand-designed And-or graphs which were used earlier these were later combined with rule based natural language techniques to device a caption for the image, this method is highly sensitive and domain specific like for various domains we use different language rules.

The later techniques which were used for image captioning used the template base approach which converts features of an image into text using templates. One such template-based captioning which took object triplets from the image and later converted them into text using templates. A more complex captioning technique using template-based text conversion was later done. The later techniques of image captioning included the usage of ranking based techniques which aimed to embed both image and text into the same vector and ranked the description for an image.

Edge based segmentation methods divide an image based on abrupt changes in the intensity of pixels near the edges. The result is a binary image with edges of the objects being detected. Neural network has been used in this approach to put images and text together. However, these approaches do not predict a caption for unseen image as they do not have the object content in them to caption the new unseen images. And these approaches do not check the results with the ground truth and evaluate the correctness of the caption generated.

## III. PROPOSED METHODOLOGY

.

### A. Understanding the Dataset

For this project we need an ample amount of data that contains both images and their corresponding captioning which act as the ground truth for the prediction our model will make. They were many such datasets that were available online for example Flickr8K, Flickr30K, MS-COCO. Both the Flickr30K and MS-COCO datasets have huge amounts of images and training such huge datasets was not very feasible on laptops. Thus, we have used Flickr8K as our dataset and proceeded our further work on this dataset, this dataset contains 8000 images and each image has 5 captions that clearly describe the content in the image. We have later divided our dataset into train and test set. The train set consisted of 6000 images and 1000 images

### B. Data Preprocessing

Our dataset set has both images and text data thus we have two steps of preprocessing for preparing a valid dataset for training our model

### B.1 Image Preprocessing

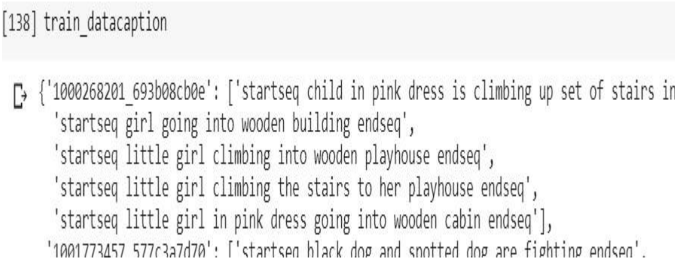
The Flickr8K dataset had 8000 various images, since all the images did not have any water marks or labels, we did not apply any morphological operations on these images, but we have resized the images to a constant size of 299x299 which in turns helps us during the feature extraction from the images using convolution neural networks.

### B.2 Text Preprocessing

In the Flickr8K each image had 5 captions describing the content of the image these captions formed the text dataset. Since the text dataset was crude, we have applied the text preprocessing to make the text ready for training and later prediction of captions. We have reduced the all text characters to lower case, removed all the special characters and punctuations as part of data cleaning. We have also handled alphanumeric and characters with single occurrences. After cleaning the dataset, we have tokenized the captions and obtained a set of words, later we have reduced the size of the vocabulary by considering only those words that occurred at least 10 times and thus obtained a vocabulary of unique words.

### *Preparing Dataset for training*

The data obtained after preprocessing was stored into a dictionary along with the image name as the key and each caption as a value, each caption was appended with a start-sequence and end-sequence so as to know where a caption starts and ends, which in turns helps us in training and testing and obtained a dataset for training



```
[138] train_datacaption

{'1000268201_693b08cb0e': ['startseq child in pink dress is climbing up set of stairs in
   'startseq girl going into wooden building endseq',
   'startseq little girl climbing into wooden playhouse endseq',
   'startseq little girl climbing the stairs to her playhouse endseq',
   'startseq little girl in pink dress going into wooden cabin endseq'],
 '1001773457_577c3a7d70': ['startseq black dog and spotted dog are fighting endseq',
```

Fig.1.Image of the dictionary storing the dataset prepared for training.

### C. Model

Since we are dealing with both images and text data which are the captions for the images our model uses the functional API developed by keras which adds up both the models one for images which is convolution neural network model and one for the text data which is Recurrent neural network model and makes one model.

.

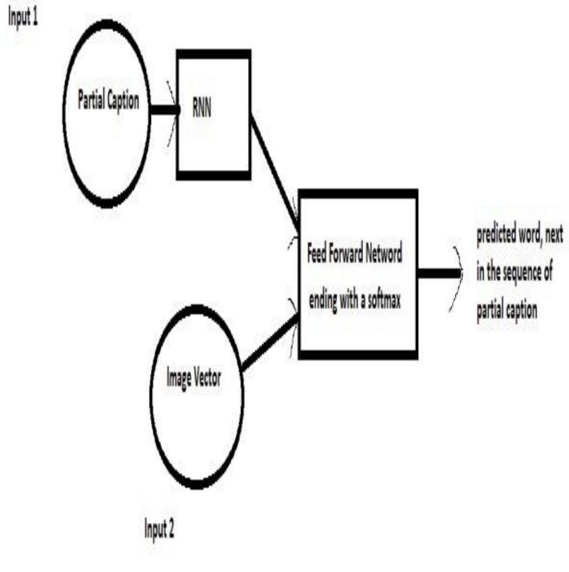

Input 1
Partial Caption
RNN
Feed Forward Netword ending with a softmax
predicted word, next in the sequence of partial caption
Image Vector
Input 2

Fig.2.Image explaining the architecture of the functional API that merges two models into one model.

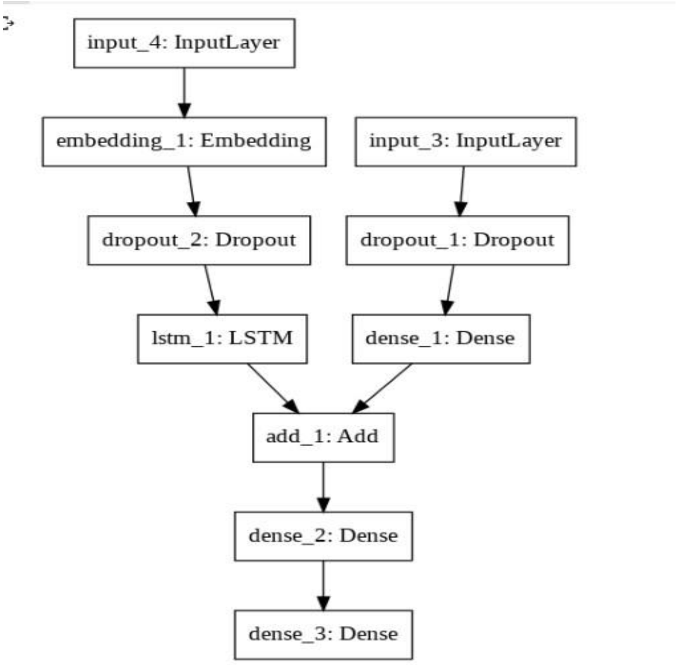


Fig.3. Image showing the architecture of the model for image captioning.

## C.1 Convolution Neural Network Model for images

CNN is deep-learning algorithm which takes an input as an image, learns the different features of the image and helps differentiate different images.

Convolution Neural Network Model

For this project we are not classifying the images thus we need a model which helps us extract the features from an image which can be later fed to another neural network to as to get the required output. Thus, we have used the already trained InceptionV3 model for this purpose which we altered accordingly by removing the last two layers. IncepectionV3 model was developed by google and was trained on ImageNet dataset and it classified the images into 1000 various classes.

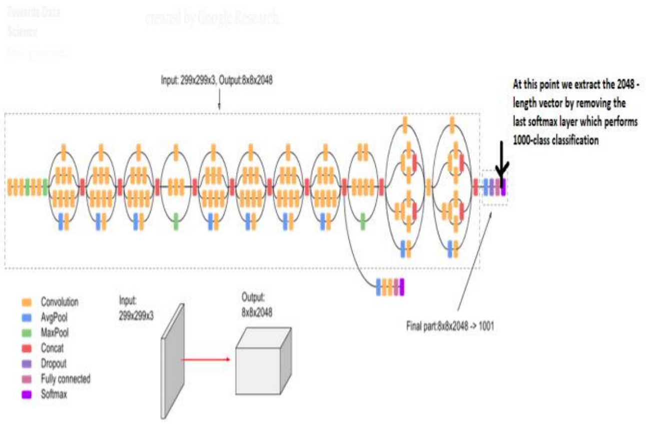

Towards Data Science
created by Google Research.
Input: 299x299x3, Output:8x8x2048
At this point we extract the 2048 -length vector by removing the last softmax layer which performs 1000-class classification
Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax
Input: 299x299x3
Output: 8x8x2048
Final part:8x8x2048 -> 1001

Fig.4. Image of InvectionV3 model for image feature extration

We have fed our images from our dataset into this model and returned the output before the classification layers and obtained an output image vector which was of size 2048. Same was done for both test and train datasets and a fixed length an image vector was obtained. The output from this model given to a dropout layer to avoid overfitting, this layer helps us to select and drop neurons from the neural network. This was later fed into a dense layer which outputs the image vector of the size 256.

## C.2 LSTM Model for captions

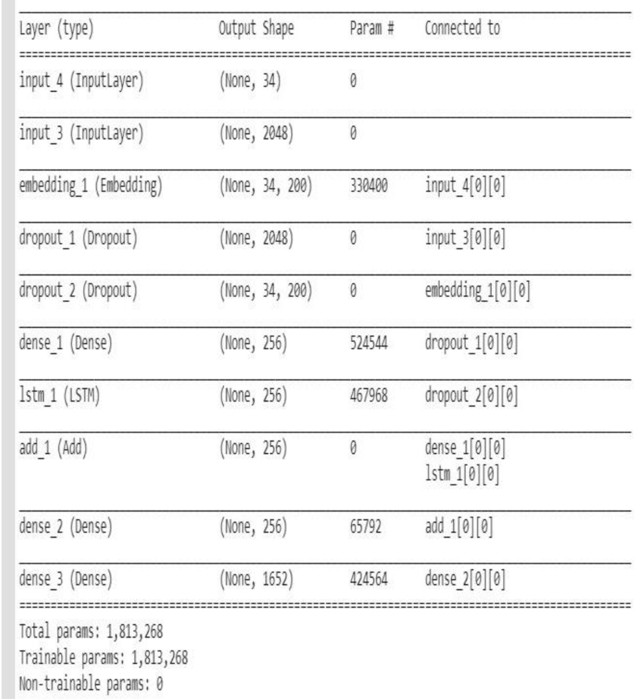
In the field of deep learning, LSTM (Long Short-Term Memory) is an artificial RNN (Recurrent Neural Network) which can process an input given in the form a sequence.
In out project we use the output from this LSTM and combine it with the image vector of size 256 and feed it into another dense layer.

Here we pass the indices of the captions as an input to the input layer, which passes the inputs into the embedding layer which in turn maps each index of the caption to a 200-dimensional vector. The output of this layer is passed on to the dropout layer which avoids overfitting by removing neurons.

Then the output of this is passed to LSTM layer which has internal contextual state cells that act as memory cells [2]. LSTM network's output changes basing on the state of these cells. This is the core of how an LSTM network works where the prediction depends on older inputs rather than depending only on the previous input.
So, in our model the LSTM predicts what the next partial caption should be based on the input partial captions. The output of this LSTM layer is of the size 256 which is same as the output from the CNN model at this stage of the model. Later, these two output vectors are combined and fed into a new neural layer.

## C.3 Training Approach

The output obtained from both the dense layers of CNN model and LSTM model are now combined and fed into a dense layer, which distributes the probability across all the unique words in the vocabulary set. The last dense layer which does the probability distribution is the softmax layer from which we can get the output caption.
The image below shows the summary of our model.



```
____________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
============================================================================================
input_4 (InputLayer)            (None, 34)           0
____________________________________________________________________________________________
input_3 (InputLayer)            (None, 2048)         0
____________________________________________________________________________________________
embedding_1 (Embedding)         (None, 34, 200)      330400      input_4[0][0]
____________________________________________________________________________________________
dropout_1 (Dropout)             (None, 2048)         0           input_3[0][0]
____________________________________________________________________________________________
dropout_2 (Dropout)             (None, 34, 200)      0           embedding_1[0][0]
____________________________________________________________________________________________
dense_1 (Dense)                 (None, 256)          524544      dropout_1[0][0]
____________________________________________________________________________________________
lstm_1 (LSTM)                   (None, 256)          467968      dropout_2[0][0]
____________________________________________________________________________________________
add_1 (Add)                     (None, 256)          0           dense_1[0][0]
                                                                 lstm_1[0][0]
____________________________________________________________________________________________
dense_2 (Dense)                 (None, 256)          65792       add_1[0][0]
____________________________________________________________________________________________
dense_3 (Dense)                 (None, 1652)         424564      dense_2[0][0]
============================================================================================
Total params: 1,813,268
Trainable params: 1,813,268
Non-trainable params: 0
```

*Fig.5. Screenshot of the model summary.*

Tuning Parameters and Hyperparameters

Initially we trained the model with 10 epochs, batch sizes as 3 and learning rate as 0.001. But the model trained with these parameters generated same and irrelevant captions for a group of 10-20 images and again same captions for the next group. Thus, we started with changing the number of epochs to 50, 100 and so on... Though the results improved from the previous we did not get the relevant captions. Later we started changing our batch size from 3 to 900 which improved our results and train loss decreased from 2.71 to 1.90 and we were able to predict relevant captions that describe the content of an image. Increasing the batch size even more could have given us better predictions but after current batch size any another batch size is crashing the system, thus we have stopped at 900 batch size. The results at this value are also near accurate. We have used Adam optimizer for adaptively optimizing the learning rate.

```
epochs_count = 200
Batch_size=512
Learning_rate=0.001
Iteration=len(trained_data_captions)/Batch_size
```

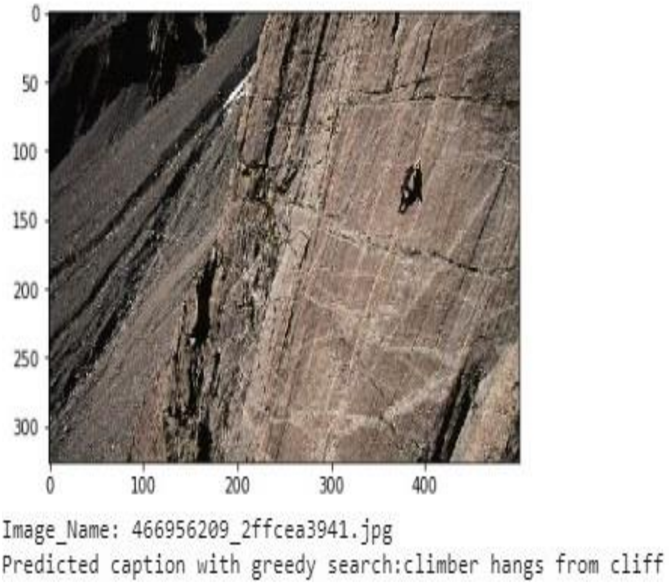
Fig.6. Sample code for parameters.

### *D. Search Algorithms*

After building and training the model now our task is to generate captions for a test image, for this purpose we use the search algorithms to generate captions for a new test image. There are many search algorithms of which we have implemented Greedy and Beam Search algorithms.

### *D.1 Greedy search Algorithm*

In greedy search algorithm we initially pass the image vector of the test image and the start sequence and then we generate words for the caption one by one that is it starts with startseq and predicts the next word. But our model generates a vector with all the unique words (1652) and their probabilities, in greedy search we choose the words with maximum probability. With the trained model and the image vector with features we can get the next word which is mostly likely to occur and generate the next caption word accurately and stops when we reach the endseq.

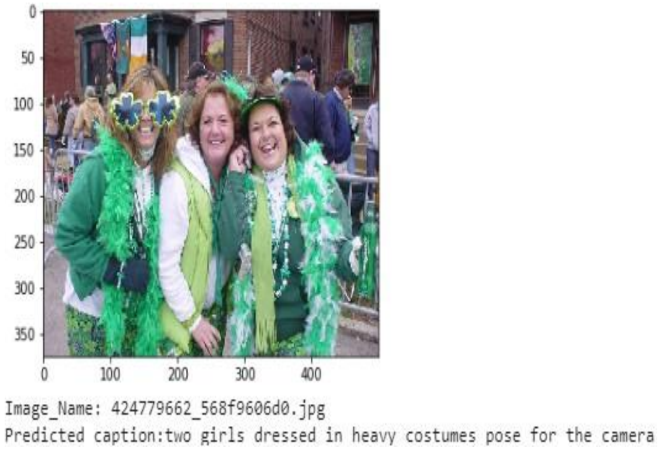


```
Image_Name: 466956209_2ffcea3941.jpg
Predicted caption with greedy search:climber hangs from cliff
```

Fig.7. Image of the caption generated for an image using greedy search.

### *D.2 Beam Search Algorithm*

Similar to greedy search we intially pass the image vector and startsequence, but unlike greedy search it does not choose the next word which is most likely to occur but instead chooses next k words, where k is beam count, that go after the startseq. These words are now appended with the startseq and form a list of k such partcal captions. This is repeatedly done until we reach to the endseq of the caption.



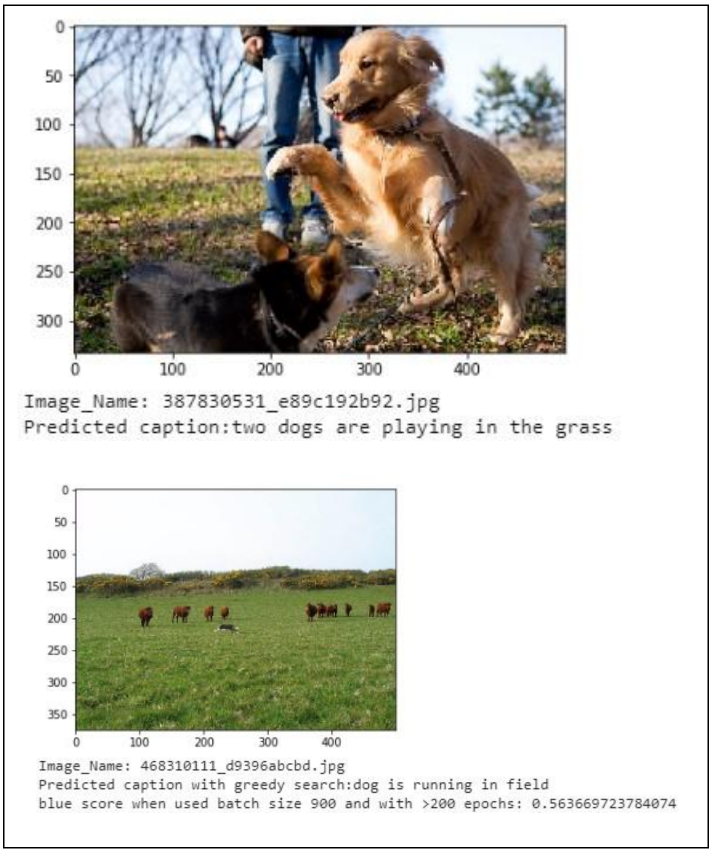
```
Image_Name: 424779662_568f9606d0.jpg
Predicted caption:two girls dressed in heavy costumes pose for the camera
```

Fig.8.Image of caption generated for an image using beam search.

## IV. EXPERIMENTAL RESULTS

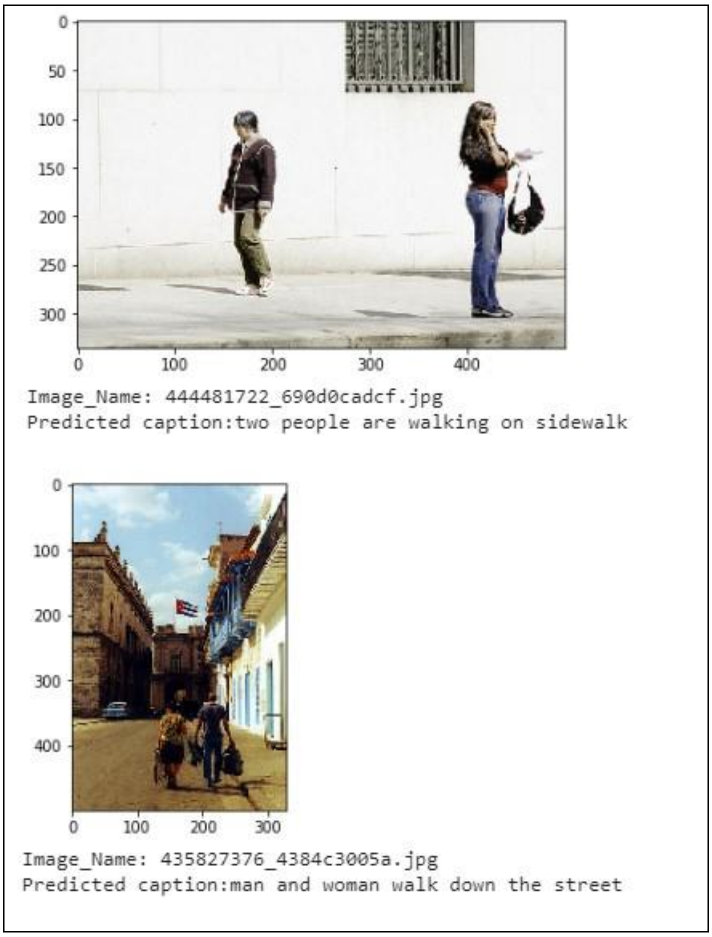
### *A. Results*

The results obtained from our model are below



```
Image_Name: 387830531_e89c192b92.jpg
Predicted caption:two dogs are playing in the grass
```

```
Image_Name: 468310111_d9396abcbd.jpg
Predicted caption with greedy search:dog is running in field
blue score when used batch size 900 and with >200 epochs: 0.563669723784074
```



```
Image_Name: 444481722_690d0cadcf.jpg
Predicted caption:two people are walking on sidewalk
```

```
Image_Name: 435827376_4384c3005a.jpg
Predicted caption:man and woman walk down the street
```

Fig.9. Images of results obtained from our model using both beam search and greedy search.

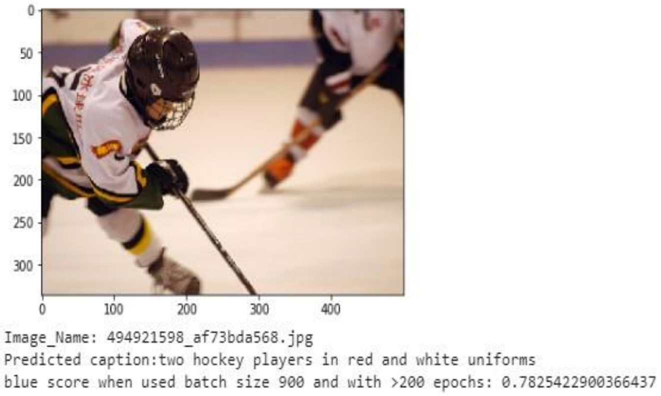
### *B. Evaluation*

Evaluating the captions generated can be done through human evaluation in which we see the image and infer the caption and check that with the generated caption, but doing so removes the automatic nature of the model and doing this for thousands of images id difficult and will vary from person to person, thus human evaluation is a bad practice for such deep learning models.

The nest way of evaluating the generated caption can be by checking its similarities with the already present ground truth caption already given to us can lead us to wrong caption prediction because it the original caption has five different words and our predicted caption has five same words that are in the original caption, then the evaluation gives us a 100 percent accuracy even though it is wrong. To overcome these difficulties, we have chosen another way of evaluating the generated caption know as BLEU score

BLEU score evaluates the generated caption with a list of reference captions, it compares the words in the predicted caption and reference captions and gives out a score. When we get a score of 1.0 it is called as the accurate prediction.
We are using the NLTK helps us determine the BLEU score by comparing the predicted caption with a list of references.

In our model we are using sentence_bleu function, which is provided by the NLTK library, to which we are passing the tokens list of generated caption as the candidate and the list of tokens of captions for each image which we have earlier saved as references.



```
Image_Name: 494921598_af73bda568.jpg
Predicted caption:two hockey players in red and white uniforms
blue score when used batch size 900 and with >200 epochs: 0.7825422900366437
```

Fig.10. Image of caption prediction along with bleu score obtained.

| Experiment | Parameters chosen | Results<br>(Bleu score) |
|---|---|---|
| 1 | Epochs:>200<br>Batch_size:900<br>Learning_rate:0.01 | Greedy search:0.52<br>Beam search(3):0.54<br>Beam search(5):0.53<br>Training loss:1.98<br>Train/Test: 6:1 |
| 2 | Epochs:>200<br>Batch_size:512<br>Learning_rate:0.01 | Greedy search:0.53<br>Beam search(3):0.54<br>Beam search(5):0.54<br>Training loss:2.3<br>Train/Test: 6:1 |
| 3 | Epochs:>100<br>Batch_size:512<br>Learning_rate:0.01 | Greedy search:0.49<br>Beam search(3):0.51<br>Training loss:2.5<br>Train/Test: 6:1 |
| 4 | Epochs:>50<br>Batch_size:256<br>Learning_rate:0.01 | Greedy search:0.025<br>Beam search(3):0.09<br>Training loss:2.9<br>Train/Test: 6:1 |

Table.1. Table maintaining the logs of various experiments during training the model.

## V. Conclusions And future works

In this project we have attempted to generate captions which gives a brief description of a given image. After preprocessing of both image and text data, we are using Google’s InceptionV3 model to convert the given image in a vector format. LSTM is used to predict the captions. We are using BEAM search to choose the appropriate sequence of captions for a given image, to evaluate the correctness of the generated caption for a given image we are using BLEU score.
Further, evaluations for the generated caption can be improved and to get more image features and more text vocabulary we can use larger datasets for training. Doing more hyperparameter training can give us better results.

## References

[1] Orial. Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, ‘Show and Tell: A Neural Image Caption Generator’,2015 [online]. Available: https://arxiv.org/pdf/1411.4555.pdf

[2] Assaad, Moawad, ‘magic of LSTM neural networks’, 2018 [online]. Available: https://medium.com/datathings/the-magic-of-lstm-neural-networks-6775e8b540cd

[3] Faizan, Shaikh, ‘Automatic image captioning using deep learning in pytorch’, 2018[online]. Available: https://www.analyticsvidhya.com/blog/2018/04/solving-an-image-captioning-task-using-deep-learning/

[4] Yash Katariya, ‘Image captioning using InceptionV3 and beam search’, 2017[online]. Available: https://yashk2810.github.io/Image-Captioning-using-InceptionV3-and-Beam-Search/

[5] Andrew Ng. ‘Sequemce to sequence models’, 2018 [online]. Available: https://www.youtube.com/watch?v=DejHQYAGb7Q

[6] Andrej karpathy,’Deep visual-semantic Alignments for generating Image Description’, 2015[online]. Avaiable: https://cs.stanford.edu/people/karpathy/cvpr2015.pdf