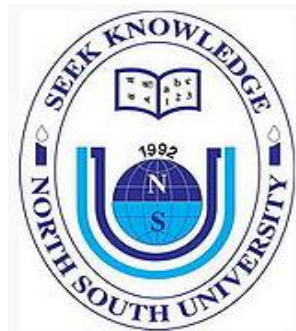


# **Senior Design Project Report**

**CSE 499 (Section: 12)**

## **HYPERPARAMETER ANALYSIS FOR IMAGE CAPTIONING BASED ON DIFFERENT LANGUAGES**



### **Submitted By**

Townim Faisal Chowdhury (ID: 1721327042)

Md. Tahrim Faroque Tushar (ID: 1621148642)

S. M. Al Faruqui (ID: 1721395042)

Uchchwas Das (ID: 1620016042)

### **Supervisor**

Dr. Md Shahriar Karim (MSK1)

Assistant Professor

**DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING  
NORTH SOUTH UNIVERSITY**

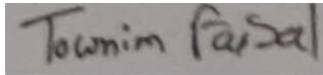
Summer 2020

## **Agreement Form**


We take great pleasure in submitting our senior design project report on "Hyperparameter Analysis For Image Captioning Based On Different Languages." This report is prepared as a requirement of the Senior Design Project CSE499, a two-semester long design course. This course involves multidisciplinary teams of students who build and test IoT devices, websites, mobile apps, or engineering processes. Design projects are selected from proposals submitted by the students, or recommended by the course instructor, or textbook design problems.

We would like to request you to accept this report as partial fulfillment of the Bachelor of Science degree under the Electrical and Computer Engineering Department of North South University.

### **Declared By:**



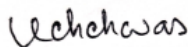
.....  
Name: Townim Faisal Chowdhury  
ID: 1721327042



.....  
Name: Md. Tahrir Faroque Tushar  
ID: 1621148642



.....  
Name: S.M. Al Faruqui  
ID: 1721395042



.....  
Name: Uchchwas Das  
ID: 1620016042

**Approved By:**

.....

Supervisor

Dr. Md Shahriar Karim (MSK1)

Assistant Professor, Department of Electrical and Computer Engineering  
North South University, Dhaka, Bangladesh

.....

Dr. Mohammad Rezaul Bari

Associate Professor & Chair, Department of Electrical and Computer Engineering  
North South University, Dhaka, Bangladesh

# Hyperparameter Analysis for Image Captioning Based on Different Languages

## Abstract:

Hyperparameter tuning is one of the complicated tasks in any deep learning model. Thus training a model with an inefficient hyperparameter combination will lead to poor accuracy. Here, we use two state-of-the-art image captioning models to analyze different hyperparameter combinations based on different languages. In image captioning, generating a meaningful or accurate caption of an image highly depends on the hyperparameters of a deep learning model. A few hyperparameters that affect the training cost are the batch size, LSTM size, etc. In this project, we conducted a detailed hyperparameter analysis for image captioning based on the language perspective. This research is done for two languages; one is English, and the other is Chinese. We tried to find suitable combinations of hyperparameters with this analysis, which demonstrated that a deeper CNN model and larger LSTM sizes could give better accuracy. We have used two different states of the art architectures: 'Show and Tell' and 'Show, Attend and Tell.' Our baseline dataset for both languages is Flickr8K, containing 8000 images with five captions for each image. As observed, the screened hyperparameters improve Bangla image captioning performance if used to train the Deep Learning model for the existing image captioning data set in Bangla. This analysis can help others by giving them an overview of the hyperparameter combinations during training on a new model. This research reveals the importance of hyperparameter tuning and how an efficient hyperparameter combination can lead to better accuracy.

# Table of Contents

CHAPTER 1.....	9
INTRODUCTION.....	9
1.1 Project Statement.....	9
1.2 Motivation.....	9
1.3 Objective.....	9
1.4 Summary.....	9
CHAPTER 2.....	10
RELATED WORKS.....	10
2.1 Introduction.....	10
2.2 Works related to our project.....	10
2.3 Problems with those projects.....	10
2.4 Proposed Solution.....	11
2.5 Summary.....	11
CHAPTER 3.....	12
PROJECT DESIGN.....	12
3.1 Framework.....	12
3.2 Model Used.....	12
3.2.1 Show and Tell.....	12
3.2.2 Show, Attend and Tell.....	13
3.3 Hyperparameter Used.....	15
CHAPTER 4.....	16
APPROACH.....	16
4.1 Dataset.....	16
4.1.1 Flickr8k Dataset.....	16

4.1.2 Flickr8k_CN Dataset.....	16
4.1.3 Chittrun Dataset.....	16
4.2 Encoder.....	17
4.2.1 VGG.....	17
4.2.2 ResNet.....	18
4.3 Decoder.....	20
4.4 Evaluation Metrics.....	21
4.4.1 BLEU.....	21
4.4.2 METEOR.....	22
4.4.3 CIDEr.....	23
4.4.4 ROUGE_L.....	24
CHAPTER 5.....	26
EXPERIMENTAL ANALYSIS.....	26
5.1 Introduction.....	26
5.2 Show and Tell.....	26
5.2.1 ResNet + LSTM.....	26
5.2.2 VGG + LSTM.....	30
5.3 Comparative Analysis on Show and Tell.....	35
5.4 Future Direction.....	40
5.5 Conclusion.....	40
REFERENCES.....	41



## List of Tables

Part No.	Name	Page No.
3.1	Description of Framework	12
3.3	Hyperparameters used in our project	15
4.1	Datasets used in this project	16
5.2.1(a)	Comparative analysis for Flickr8K(English) with Resnet-152 in Show and Tell model	27
5.2.1 (b)	Comparative analysis for Flickr8K(Chinese) with Resnet-152 in Show and Tell model	29
5.2.2 (a)	Comparative analysis for Flickr8K(English) with VGG16 in Show and Tell model	31
5.2.2 (b)	Comparative analysis for Flickr8K(Chinese) with VGG16 in Show and Tell model	33
5.4 (a)	Difference of top highest and smallest accuracy score for VGG16 and Resnet-152 in Show and Tell model	36
5.4 (b)	Comparative analysis for Chittrun dataset in Show and Tell model	39
5.4 (c)	Caption analysis for the Chittrun dataset between Original Chittrun Implement and Our Model	39



## List of Figures

Part No.	Name	Page No.
3.2.1	Image captioning using Show and Tell Model	13
3.2.2	Image captioning using Show, Attend and Tell Model	14
4.2.1	VGG16 Architecture Used in ImageNet	17
4.2.1	Layer formation in VGG16	18
4.2.2	Skip connections in ResNet	19
4.2.2	Comparison between ResNet, plain network, and vgg19 network	19
4.3	Structure of Standard RNN	20
4.3	LSTM Structure	20
5.2.1	Accuracy for Flickr8k English in Resnet-152 of Show and Tell model	28
5.2.1	Accuracy for Flickr8k Chinese in Resnet-152 of Show and Tell model	30
5.2.2	Accuracy for Flickr8k English in VGG16 of Show and Tell model	32
5.2.2	Accuracy for Flickr8k Chinese in VGG16 of Show and Tell model	34
5.4	Accuracy in Resnet152 for English and Chinese Flickr8k	35
5.4	Accuracy in VGG16 for English and Chinese Flickr8k	36
5.4	Effects on evaluation metrics in Show and Tell model	37
5.4	Loss in Show and Tell model	38

# CHAPTER 1

## INTRODUCTION

### 1.1 Project Statement:

In this project, two state-of-the-art image captioning models are used. One is Show and Tell [1], and the other one is Show, Attend and Tell [2] model. Both of this model uses different CNN architectures which consists of many layers and also LSTM [3] for the sequential data. Due to this training, a deep learning model requires a massive amount of time and resources. On average running, a model with 32 epochs took about 14-15 hours. During training a model, hyperparameters tuning is an integral part of any deep learning project. For example, if someone is creating a new model, then he/she has to find a suitable hyperparameters combination to get better accuracy and optimally use the resources. As mentioned before, it takes a lot of time and resources, and these hyperparameters act differently based on the language, model, and dataset in the case of image captioning. To minimize these problems, we want to analyze those changes based on different hyperparameters combinations and languages and give a rough estimate of how hyperparameters behave with different combinations.

### 1.2 Motivation:

We started our initial work on Bangla image captioning using the Chittrun [4] dataset. Chittrun is an image captioning dataset in the Bangla language, and the images are based in Bangladesh. When training this new dataset, we counter several problems, and most of them are regarding finding a suitable hyperparameter combination. First of all, the language is different, and also the dataset is different. So, we had to go through many trials and errors to find the best hyperparameter combination to get the highest accuracy. As we were training, we realized that choosing/finding the best hyperparameter is a challenging task itself. So, we decided to expand our research to counter this problem and analyze changes in hyperparameter tuning.

### 1.3 Objective:

Our goal is to analyze those changes in hyperparameter tuning based on the language perspective. For that, we will analyze multiple combinations of hyperparameters for different datasets in English, Chinese, and Bangla language. Then, find the optimum hyperparameter setting for the best accuracy in multiple languages. Finally, generate a comprehensive comparison of the hyperparameter analysis.

### 1.4 Summary:

In this chapter, we have discussed our motivation behind the hyperparameter analysis and why it is crucial to train the model with an optimum hyperparameter combination. Also, we have mentioned our objective to provide an extensive analysis of hyperparameters in image captioning.

## **CHAPTER 2**

### **RELATED WORKS**

#### **2.1 Introduction:**

In any deep learning project tuning the hyperparameter takes a huge amount of time mainly because Hyperparameters act differently on different datasets and different deep learning models. In most of the works done in image captioning, researchers mainly focused on that particular model and its best hyperparameter, which they have found by trial and error. In this project, we want to bring everything together and conduct an extensive analysis of Hyperparameters based on three languages English, Bangla, and Chinese, with some state-of-the-art models.

#### **2.2 Works Related to Our Project:**

The number of researches done solely on Hyperparameter Analysis on a large scale is very low. One of the reasons might be that it takes a massive amount of time to tune them and to train the model, no need to mention that these trainings are very resource-intensive. One of the notable works done by Amish Patel and Aravind Varier [5]. In this research, they have mainly focused on different encoder-decoder architecture, and for the encoder, they have used different layered combinations of ResNet [6] (ResNet-18, ResNet-50, ResNet-101) architecture. For the decoder, they have used LSTM [3] and Transformer [7]. They have conducted the whole research on Flickr8k [8] datasets. There are other research projects out there, but they are only focusing on a particular hyperparameter. For instance, Yanzhao Wu et al. [9] have presented an extensive study of 13 learning rate functions and also their associated LR policies. They have done it on the MNIST and CIFAR-10 dataset using the LeNet [10], 3-layer CNN [13], and ResNet [6] architecture. Samuel L. Smith et al. [11] presented how the increased batch size and decreased learning rates and vice versa can impact accuracy. They showed that decreasing the learning rate and increasing the batch size in training have the same effect. Also, scaling the batch size and increasing the effective learning rate can reduce the number of parameter updates. Yanzhao Wu et al. [12] presented analysis on MNIST, CIFAR-10, and ImageNet [14] datasets. They have used four frameworks, such as TensorFlow [15], Caffe [13], PyTorch [16], and Theano [17] using both CPU and GPU. As for the optimizer, SGD was used, and they run all of the training on Server.

#### **2.3 Problems with those projects:**

These research projects focus on particular sets of hyperparameters. Some focus on encoder-decoder variation, batch size variations, different learning rate variations, etc. This hinders getting the whole picture of hyperparameter analysis. Also, these analyses are done using the English language only, which can be a strong bias. Language plays a vital role in conveying or describing the right message in image captioning. So, the message can be different based on the language. Testing it only in English is not enough. So, further testing is needed to analyze the hyperparameters in different languages to understand its effect in different languages fully.

## **2.4 Proposed Solution:**

In those related works, other researches did cover some crucial analysis of the hyperparameters but only focused on the English language, and some of them only cover 1 or 2 hyperparameters. In this project, the analysis has been conducted on a large scale on different hyperparameters. Moreover, we have tested those hyperparameters in different languages, such as in English, Bangla, and Chinese, and tried to give a full overview of the hyperparameters' changes so that others can easily tune their models' hyperparameters to gain higher accuracy in a short amount of time. Also, different state-of-the-art models have been used to test these hyperparameters.

## **2.5 Summary:**

In this section, we have discussed some of the research related to our project and how they are different from our approach. Also, we have presented some of the aspects of those related projects that could have been done.

## CHAPTER 3

### PROJECT DESIGN

#### 3.1 Framework:

Table 3.1: Description of Framework

Framework	What it does	Why selected this framework	Similar Framework
PyTorch [16]	PyTorch is a very well-known framework for any deep learning project developed by Facebook in 2016. It makes it easier to create a deep learning model quickly without raw python code using NumPy.	PyTorch is mainly used for research purposes then in production. It has better memory organization and optimization. Also, it has better compatibility with NumPy. Moreover, it is easier for a beginner to adapt to the PyTorch framework.	TensorFlow [15]

#### 3.2 Model Used:

##### 3.2.1 Show and Tell:

This model was first introduced in 2015 by Vinyals et al. [1] and it was the state-of-the-art model of that time. They have tested their model on Flickr8k [8], Flickr30k [18][19], Pascal [20], and MSCOCO [21] dataset.

The goal was to maximize the probability of the correct caption/description of that particular image, and for that, they have used this formula.

$$\theta^* = \arg \max_{\theta} \sum_{(I,S)} \log p(S|I; \theta) \quad (1)$$

Here,  $\theta$  = Parameters of the model,

I = The given image to the model,

S = Correct description of that image

For the sequence tasks, they have used LSTM [3] (Long-Short Term Memory). This model uses a multilayered LSTM. One LSTM is used to read the input sequence to obtain a large fixed-dimensional

vector, and the other LSTM is used to extract the output sequence from that vector. It can learn data with a long-range temporal dependency.

To extract the features of the images, they have used the Convolutional Neural Network (CNN). CNN is very powerful when it comes to image feature extraction. CNN uses different sizes of filters to identify the salient object in an image. In this model, the encoded image representation is only shown once in the 1st time step of the LSTM. Then, the LSTM uses its gated cell to add, update, and delete data based on the importance of information. Finally, the model creates a caption for an image.

The following image depicts the whole process of caption generation for an image:

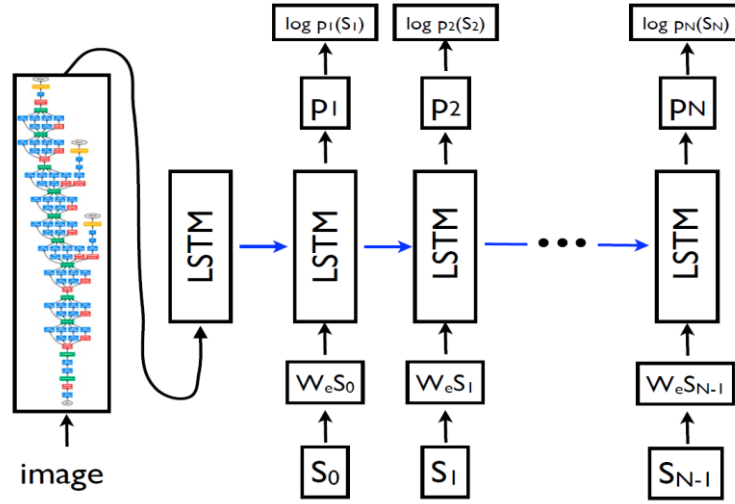


Figure 1: Image captioning using Show and Tell model [1]

### 3.2.2 Show, Attend, and Tell:

The show, Attend and Tell model was first introduced by Xu et al. [2] in 2016. This model is an extension of the Show and Tell [1] model. It uses visual attention mechanisms that focus on the different parts of an image during caption generation by LSTM [3].

This attention framework does not use any object detectors to detect the salient objects in an image. It learns latent alignments from scratch. This enables the model to learn to attend to abstract concepts. The difference between the Show and Tell and Show, Attend and Tell model is that the image is shown at the first-time step to the LSTM to generate captions in the show and tell model. On the other hand, the Show, Attend, and Tell model uses different parts of the images in every time step while generating a caption.

Figure 2 depicts the image caption generation process in the Show, Attend, and Tell model. This model did not use any object detector; instead, the model learns latent alignment from scratch. This will help the model to attend to different abstract concepts.

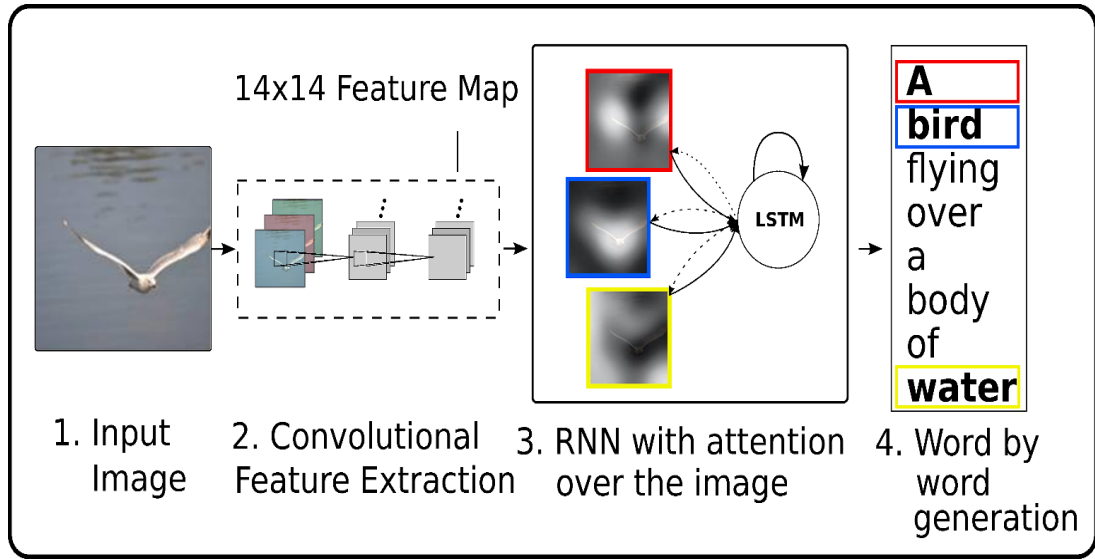


Figure 2: Image captioning using Show, Attend and Tell model [2]

At first, the model takes a single raw image and generates a caption  $y$ , which consists of encoded-words.

$$y = \{y_1, \dots, y_c\}, y_i \in R^K \quad (1)$$

Here,  $K$  is the size of the vocabulary created from the captions, and  $C$  is the length of the caption.

As for the decoder part, they have used LSTM. They used CNN for extracting the features of the images. In this model, they have described two types of attention mechanisms: soft attention and the other is hard attention. Soft attention is when we calculate the context vector as a weighted sum of the encoder hidden states. On the other hand, hard attention is when, instead of weighted average of all hidden states, we use attention scores to select a single hidden state.

They have used Flickr8k [8], Flickr30k [18][19], and MSCOCO [21] datasets for the evaluation, and the highest scores were achieved when they applied hard attention.

### 3.3 Hyperparameter Used:

**Table 3.3: Hyperparameters used in our project**

Dataset	Encoder	Decoder	LSTM size	Learning rate	Optimizer	Batch size
Flickr8k EN [8]	VGG16	LSTM	256	X	Adam	8
Flickr8k CN [24]	Resnet	X	512	X	SGD	16
Chittron [4]	X	X	1024	0.0001	X	32
X	X	X	X	X	X	64

For the encoder, we have used VGG16 [22], a 16-layer network, and ResNet152 [6], a 152-layer network. The reason behind choosing VGG16 and ResNet152 because both of them scored higher in the ImageNet [14] competition. We have used the LSTM [3] with 256, 512, 1024 sizes hidden layers for the decoder. LSTM is very commonly used in image captioning, and it is very good at handling past dependencies. As for the LSTM sizes, due to lack of resources, we could not go past 1024. For the learning rate, we have used 0.0001 with Adam [23] and SGD optimizer. We could have used other LR such as 0.01 or 0.1, but before starting the analysis, we tested those LR, and it seems 0.0001 performs relatively better than the others.

Furthermore, we have trained our model in different batch sizes [11] starting from 8 and then increasing them to the power of 2. For example, 8, 16, 32, 64 batch size has been used throughout the training. However, we have used 64 as the largest batch size in this project because this is the best we could achieve with our hardware limitation.

Based on these variations, we have collected our data in excel sheets and then analyze every specification by plotting them using matplotlib.



# CHAPTER 4

## APPROACH

### 4.1 Dataset:

**Table 4.1: Datasets used in this project**

Datasets	Total Images	Train	Validation	Test	Caption per Image
Flickr8k (English)	8000	6000	1000	1000	5
Flickr8k (Chinese)	8000	6000	1000	1000	5
Chittrun (Bangla)	9154	6408	1946	800	2

#### 4.1.1 Flickr8k Dataset (English):

There are 8000 images in total in JPEG format in different sizes. Among them, 6000 are for training, 1000 for validation, and another 1000 for testing. All of them have 5 captions in English. These captions are given in a text format file named Flickr8k.token.txt. These train, validation, and test splits are already given with the dataset in a text file. So, we did not have to split them up. We ran a python code to make a JSON file to feed those captions and images to the model.

#### 4.1.2 Flickr8k Dataset (Chinese):

The only difference here is the style of captioning, meaning they are in the Chinese language. In the Chinese language, sentences have no spaces between them. We tried to split them using NLTK [25] and it seemed that using another package called jieba [26] was easier, which is used to tokenize a sentence in the Chinese language.

#### 4.1.3 Chittrun Dataset (Bangla):

Chittrun is the first image captioning dataset for the Bangla language. It has a total of 9154 images with 2 captions per image. We randomly split the whole dataset into 6408 train, 1946 validation, and 800 test images. We have used the Bengali Natural Language Processing (BNLP) for tokenizing the Bangla sentences instead of using the NLTK [25] package. We found that the NLTK package did not understand

the Dari (ل), which is full-stop(.) in the English language. These images mainly focus on Bangladeshi places and people and their day to day life.

## 4.2 Encoder:

In any deep learning model, encoder refers to the part where features of images are being extracted so that it can be used in training. Mostly it refers to a convolutional neural network or CNN model. We have used the VGG [22] and ResNet [6] model as encoders in our analysis.

### 4.2.1 VGG:

VGG a CNN architecture first introduced by Simonyan et al. [22] in 2014 in a paper named "Very Deep Convolutional Networks for Large Scale Image Recognition." It was an improvement over AlexNet [27], which uses large-sized kernels. Large kernels/filters are very resource hungry. So, VGG replaced those large sized kernels with 3x3 sized filters and stacked them up one after another. That way, calculations got more efficient, and fewer resources were needed.

The input image size given in the CONV1 layer is 224 X 224 for the VGG network, then going through the layers, the images get reduced in size, and features are being extracted. In the VGG network, there are 138 million parameters that are needed to learn during training.

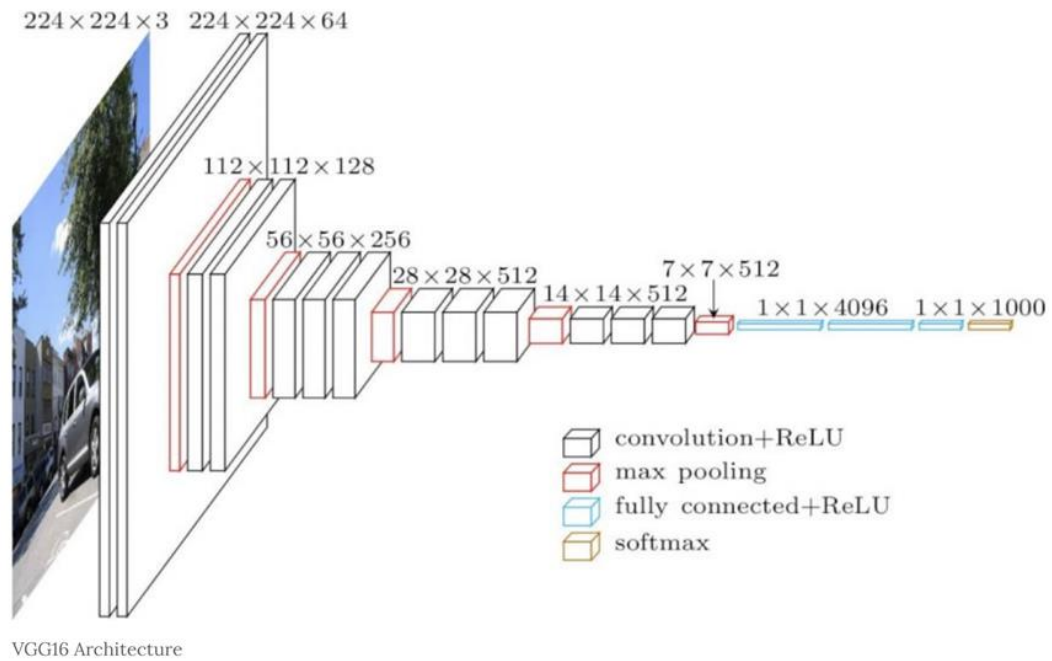


Figure 3: VGG16 Architecture Used in ImageNet

As for the filters, a small 3 X 3 filter is used to learn the images' patterns and features. The convolutional stride refers to how much the filters are going to slide in an image. In the case of VGG, it is fixed to 1 pixel. The padding is 1 pixel in VGG, which is used to maintain the image information. In the case of pooling, max pooling is applied with a 2x2 pixel window and stride 2. In every layer, for the non-linearity, ReLU or rectified linear unit is used.

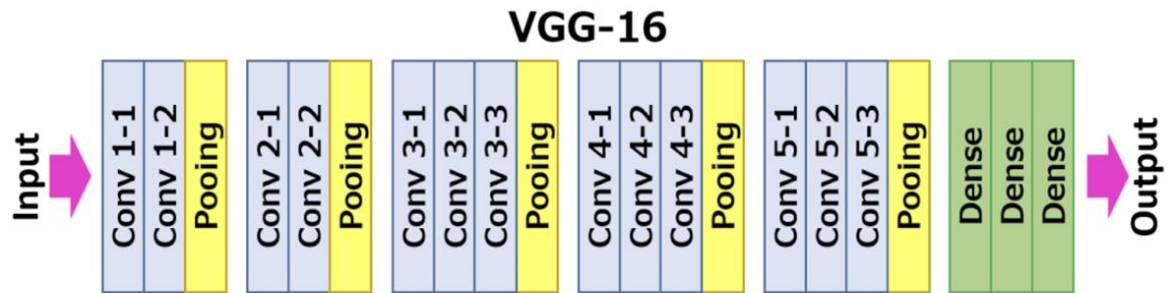


Figure 4: Layer formation in VGG16

There are two major drawbacks in VGG; one is it is very slow to train the whole model, and another is the weights of the VGG architecture are quite large

#### 4.2.2 ResNet:

ResNet was introduced by Kaiming He et al. [6] during their work at Microsoft in a paper called "Deep Residual Learning for Image Recognition" in 2015. ResNet won the famous ImageNet competition called ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2015, with an error rate of only 3.57%. This is the first CNN architecture to successfully train a neural network by using more than 100 layers. Previously, with any deep neural network, as the network goes deeper and deeper, the performance gets saturated, but it is not caused due to overfitting. Also, adding more layers to it increases the error rate. [28][29]

ResNet overcame this problem caused by deeper neural networks by initializing better weight, by adding a new activation function, by using new normalization layers and the skip-connections. These skip connections or residual connections are used to pass any information from the previous layer to the next layer, which is very helpful if the network is very long.

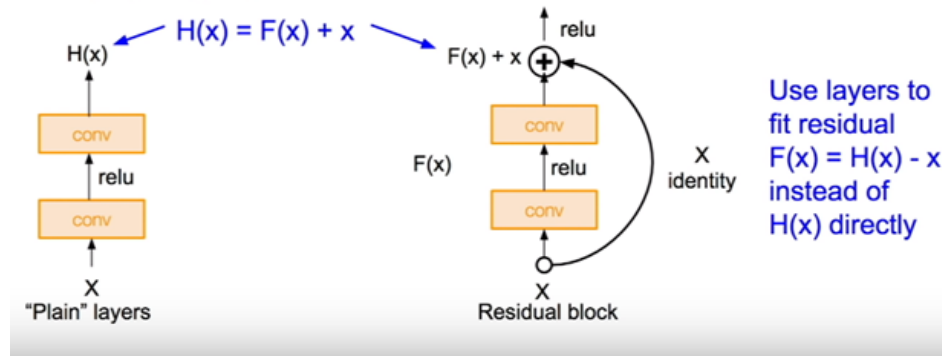


Figure 5: Skip connections in ResNet

In figure 4, the plain network passes the information  $x$  from conv1 to conv2 through a rectified linear unit (ReLU) activation function. So, the information gets processed in the way. But in the ResNet [6] architecture, the processed data passes from one layer to the next, and also with that, the original data from the previous layer passes to the next layer without any processing. In that way, the information does not get lost when moving to deeper layers. The information is always passing from one layer to another by using a skip connection or residual connection.

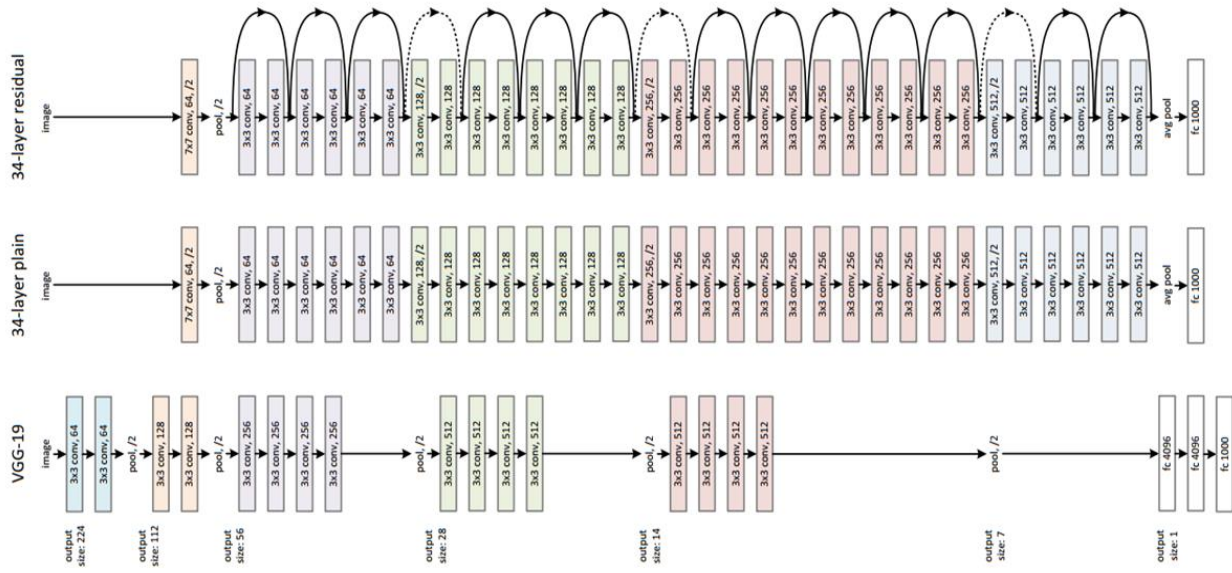


Figure 6: Comparison between ResNet, plain network, and vgg19 network

As we can see in figure 5 that VGG [22] and plain both the network lacks the skip connections. Whereas ResNet uses it after every 2 layers.

### 4.3 Decoder:

In our project for the decoder, we have used Long Short-Term Memory networks (LSTM). LSTM is a special kind of RNN. LSTM has the capability to learn long-term dependencies. They were introduced by Hochreiter & Schmidhuber in 1997 in their paper called Long short-term memory [3].

When training using vanilla RNN, the problem is that it cannot remember the information for a longer period of time. This is why data gets missing; thus, missing information cannot be used for sequential learning. The vanilla RNN (Figure 7) contains a single layer. Within that layer, there is a single  $\tanh$  layer that takes the input  $x_t$  and previous information and then passes it to the output and to the next cell/iteration.

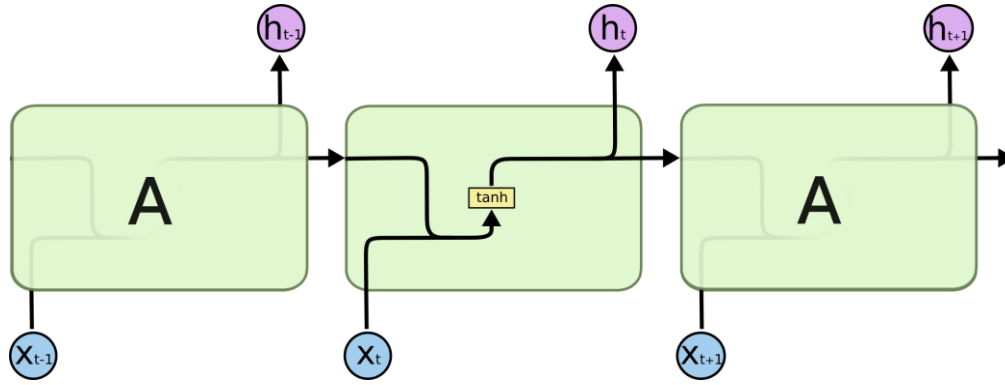


Figure 7: Structure of Standard RNN

On the other hand, LSTM uses a gated cell (Figure 8) which is a chain-like structure. Comparing the vanilla/standard RNN, the LSTM uses 4 intersecting neural network layers.

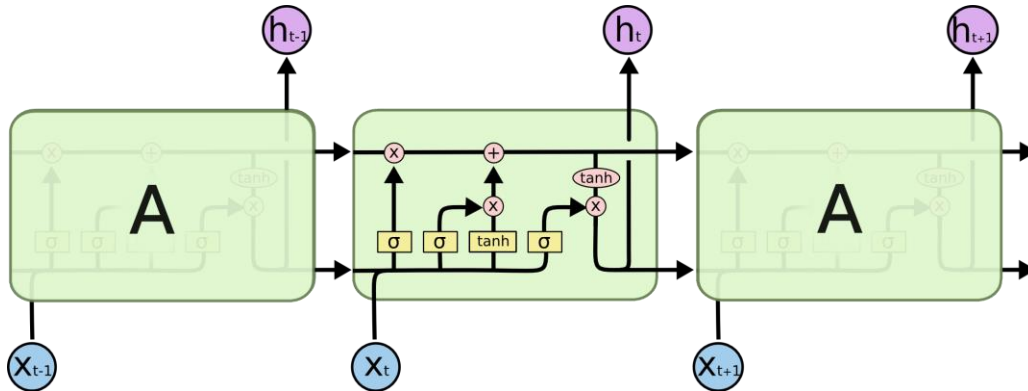


Figure 8: LSTM Structure

The most important part of the LSTM is the cell state, which is a single line on top of the cell. It is used to pass information from one cell to another. Also, gated cells are used, such as the forget gate, which is a sigmoid layer, the input gate, which is also a sigmoid layer. This has 2 parts: the first sigmoid layer is applied with the previous data and the present data, and then the tanh layer is applied. By combining them

both, the cell will decide which information should be added to the cell. After that cell state will be updated, deciding what parts of the cell state are going to get passed on. Then, the cell state will go through a  $\tanh$  layer to normalize the values between  $-1$  to  $1$  and, then, it will be multiplied by a sigmoid gate.

## 4.4 Evaluation Metrics:

Evaluating a model after each epoch in training will give an idea of how the model is doing. Is it overfitting, or is it underfitting, or is it going to the optimum accuracy? Evaluation of the validation set for a particular dataset also gives the idea of how much the dataset performs with a specific model. We have used BLEU (1-4) [30], METEOR [31], CIDEr [32], and ROUGE\_L [33] evaluation matrices to evaluate our different hyperparameter settings in our image captioning task.

### 4.4.1 BLEU:

The most used evaluation matrix in Natural Language Processing (NLP) is the Bilingual Evaluation Understudy (BLEU). BLEU was first introduced in 2002 by Kishore Papineni et al. [30] in a paper called "BLEU: A Method for Automatic Evaluation of Machine Translation" They made this evaluation metric at IBM T. J. Watson Research Center for evaluating machine-generated translation. Later, it is being used in other NLP tasks. In image captioning, it is considered to be the main evaluating factor.

BLEU measures how much the captions generated by humans and the machine are close based on the differences among word choice and word order. BLEU compares the n-gram of the human-generated caption and the n-gram of the machine-generated captions and counts the number of matches between them. The n-gram refers to a sequence of words that occur within a given window, where n is the window size.

For example, if we consider the sentence, "Deep Learning tasks are interesting."

when  $n = 1$ , n-gram of this sentence will be (Deep, Learning, tasks, are, interesting), it is called unigram  
when  $n = 2$ , n-gram of this sentence will be (Deep Learning, Learning tasks, tasks are, are interesting) it is called bigram.

BLEU uses modified n-gram precision to correctly distinguish how many words match with the human caption. At first, it calculates the count clip for any n-gram by counting the maximum number of times a generated caption's n-gram occurs in any single human caption; this is known as Count. After that, it takes the maximum number of n-grams occurrences in any human caption count. This is known as Max\_Ref\_Count.

Then by taking the minimum of the Count and the Max\_Ref\_Count, we can calculate the Count Clip.

$$Count_{clip} = \min(Count, Max\_Ref\_Count) \quad (3)$$

Finally, the modified precision  $P_n$  is calculated by dividing the summation of the clipped n-gram counts for all the candidate sentences in the corpus by the number of generated caption's n-grams.

$$P_n = \frac{\sum_{C \in \text{Candidates}} \sum_{n\text{-gram} \in C} \text{Count}_{\text{Clip}}(n\text{-gram})}{\sum_{C' \in \text{Candidates}} \sum_{n\text{-gram}' \in C'} \text{Count}(n\text{-gram}')} \quad (4)$$

Furthermore, if the captions are too small, they added Brevity Penalty (BP) to handle this. BP will be 1.0 when the generated caption length is the same as the human caption length. Basically, Brevity Penalty is an exponential decay and calculated using the #words in the human caption and #words in the generated caption.

$$\text{BP} = \begin{cases} 1, & c > r \\ e^{(1-r/c)}, & c \leq r \end{cases} \quad (5)$$

Here,  $c$  is the #words in human captions, and  $r$  is the #words in generated captions

Finally, using BP and  $P_n$  we can calculate the BLEU score.

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^N w_n \log P_n\right) \quad (6)$$

Here,  $N$  refers to the n-gram number, normally  $n = 1, 2, 3, 4$  are being used when evaluating an image caption.  $P_n$  is the Modified Precision and  $w_n$  refers to the weight for each modified precision.

#### 4.4.2 METEOR:

METEOR is another evaluation metric mainly proposed for machine translation (MT) by Satanjeev Banerjee and Alon Lavie [31] from Language Technologies Institute, Carnegie Mellon University in a paper called "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments." They proved that METEOR has a better correlation with human judgment.

The main problem with BLEU [30] is that when the generated captions are too short in length, then the Brevity Penalty (BP) is applied, and because of that, the whole sentence's score gets reduced even if the caption is accurate. They counter this problem by modifying the precision and recall computations in METEOR by calculating the F-score. F-score is based on the mapping of unigrams and a penalty function for incorrect word order.

Here, the weighted F-score is measured by finding the largest subset of mappings between the generated caption and human sentence. To do that, we have to look for exact matches for the alignment.

F-score is calculated by

$$F = \frac{PR}{\alpha P + (1 - \alpha)R} \quad (7)$$

This equation also can be written like this for unigram matches. Here, P = precision, R = recall

$$F_{mean} = \frac{10PR}{9P + R} \quad (8)$$

Furthermore, in the case of longer matches, METEOR computes a penalty for a given alignment using #chunks divided by the #unigram\_matched.

$$\text{Penalty} = 0.5 * \left( \frac{\#chunks}{\#unigram\_matched} \right) \quad (9)$$

Finally, using the Fmean and the Penalty METEOR score is calculated

$$\text{Score} = F_{mean} * (1 - \text{Penalty}) \quad (10)$$

#### 4.4.3 CIDEr:

CIDEr was first introduced in 2015 by Ramakrishna Vedantam et al. [32] in a paper called "CIDEr: Consensus-based Image Description Evaluation." CIDEr measures the similarity of a generated captions vs. a set of humans generated captions. CIDEr gives more weight-age to important n-grams which were not present in the BLEU [30] evaluation metric. It also shows a higher correlation with human consensus scores compared to other metrics such as the BLEU.

At first, they calculated the Term Frequency Inverse Document Frequency (TF-IDF) weighting for each n-gram. The reason to measure TF-IDF is that sometimes n-grams in the generated captions are present in the human-generated captions. Also, the n-grams that are not present in the human-generated caption should not be in the machine-generated captions.



$$g_k(s_{ij}) = \frac{h_k(s_{ij})}{\sum_{w_l \in \Omega} h_l(s_{ij})} \log \left( \frac{|I|}{\sum_{l_p \in I} \min(1, \sum_q h_k(s_{pq}))} \right) \quad (11)$$

Here, the number of times an n-gram  $w_k$  occurs in a reference sentence  $s_{ij}$  is denoted by  $h_k(s_{ij})$ .

Then, the CIDEr score is computed by using the average cosine similarity between the machine-generated caption and the human-generated caption. It also takes precision and recall into account during the calculation.

$$CIDER_n(c_i, S_i) = \frac{1}{m} \sum_j \frac{g^n(c_i) \cdot g^n(s_{ij})}{\|g^n(c_i)\| \|g^n(s_{ij})\|} \quad (12)$$

#### 4.4.4 ROUGE\_L:

Another evaluation metric we used in our analysis was ROUGE\_L. It was introduced by Chin-Yew Lin [33] in a paper called "ROUGE: A Package for Automatic Evaluation of Summaries." As it is mentioned in the paper, ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It determines the quality of a summary by comparing it to other summaries created/generated by humans. To do that, it considers counting the number of overlapping units such as n-gram, word sequences, and word pairs between the machine-generated summary and the human-generated summary.

In this paper, Chin-Yew Lin presented four different ROUGE measures: ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S. But in image captioning, ROUGE\_L is commonly used for evaluation. Basically, ROUGE is a modified version of BLEU.

As for the recall, in the case of ROUGE, it refers to how well the reference summary (in our case, human captions) is captured by the system summary (in our case, generated captions). It can be calculated by

$$\frac{\#overlapping\ words}{total\ \#words\ in\ reference\ summary} \quad (13)$$

However, the problem with machine-generated summary or captions is they can be very long. Also, some of the words can be unnecessary, meaning they do not express accurate messages. That is why precision is considered. It means how much of the generated caption is relevant to describe the image.

ROUGE\_L measures the longest matching sequence of words, and for that, it uses the Longest Common Subsequence (LCS). LCS does not need any consecutive matches but in-sequence matches that reflect sentence-level word order as n-grams. There is another advantage, and that is it automatically includes the longest in-sequence common n-grams. Thus, no predefined n-gram length is needed.

When applying the sentence level LCS, the main idea is that the longer the LCS is for both the summary/captions (machine-generated and human-generated), the more similar they are. The paper proposed an LCS-based F-measure to estimate the similarity between two summaries (in our case, captions). By considering all of these, the following equations are applied to compute the ROUGE\_L score.

$$R_{lcs} = \frac{LCS(X, Y)}{m} \quad (14)$$

$$P_{lcs} = \frac{LCS(X, Y)}{n} \quad (15)$$

Here, X = Human-generated caption, and m is the length of that caption

Y = Machine-generated caption, and n is the length of that caption

Then, using the  $P_{lcs}$  and  $R_{lcs}$  LCS-based F-measure is calculated, which is the desired ROUGE\_L score.

Here,  $\beta$  is calculated using  $\beta = P_{lcs}/R_{lcs}$

$$F_{lcs} = \frac{(1 + \beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \quad (16)$$

# CHAPTER 5

## EXPERIMENTAL ANALYSIS

### 5.1 Introduction:

In this section, we have presented the results of our hyperparameter analysis and also discussed the results' variation on different combinations of hyperparameters. We have divided the whole analysis based on datasets and then divide them up based on different encoder and decoder combinations.

For all the analysis:

For the optimizer, we have used Adam [23] with a learning rate of 0.0001. We have used batch sizes of 8, 16, 32, and 64 and also used different LSTM [3] sizes of 256, 512, and 1024. We have compared the Flickr8k (English) [8] against the Flickr8k (Chinese) [24] dataset and presented 4 evaluation metric scores BLEU (1-4) [30], METEOR [31], CIDEr [32], and ROUGE\_L [33].

### 5.2 Show and Tell:

#### 5.2.1 ResNet + LSTM:

For this analysis, we have used ResNet [6] as the encoder and the LSTM [3] as the decoder. Among the other ResNet versions, we have chosen the ResNet152 because of its deeper layers.

**Table 5.2.1(a): Comparative analysis for Flickr8K(English) with Resnet-152 in Show and Tell model. Red, Blue, and Green colors represent the top, second top, and third top scores respectively.**

Hyperparameters			Flickr8K (English)							
Encoder	LSTM Size	Batch Size	Loss	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE_L	CIDEr
Resnet-152	256	8	2.47312	60.87	42.49	28.97	19.759	18.813	44.27	45.663
		16	2.4704	61.97	43.69	30.13	20.6	19.22	45.328	48.14
		32	2.5682	60.44	41.64	27.7	18.55	18.501	43.8	42.5
		64	2.6932	58	38.85	25.36	16.7	17.21	41.4	35.83
	512	8	2.4682	61.19	42.76	29.33	19.79	18.95	44.39	45.66
		16	2.44	62.84	44	30.07	20.28	19.42	45.66	47.2
		32	2.493	62.64	44.1	30.23	20.632	19.44	45.36	50.17
		64	2.5054	62.05	43.39	29.66	19.83	19.42	45.356	47.79
	1024	8	2.499	61.58	43.12	29.27	19.71	19.07	44.64	46.58
		16	2.4683	61.93	43.61	29.94	20.12	19.42	44.945	50.17
		32	2.49644	61.64	44	30.66	21.15	19.64	45.66	51.1
		64	2.5598	62.84	44.52	30.69	20.622	19.516	45.755	49.77

As we can see from the table, the larger LSTM size of 1024 gives the best accuracy in BLEU (1-4), METEOR, ROUGE\_L, CIDEr compared to the other LSTM sizes 256 and 512. The reason for that, the deeper the hidden layer size, the more information stores. As we have seen before, like the ResNet, the deeper architecture can manage to learn information more accurately. Thus, a larger LSTM size helps in this case.

Also, we can see that larger batch sizes are responsible for getting the highest accuracy. In batch size 64, we are getting 4 of the highest evaluation scores, and in batch size 32, we are getting the rest of them. All of them are getting these accuracies in the LSTM size of 1024. Larger batch sizes help the model to converge to the optimum minima, thus getting the highest accuracy. So, higher batch sizes combined with higher LSTM layers gives the best accuracy.

These findings also presented in Figure 9 for a better understanding of the analysis

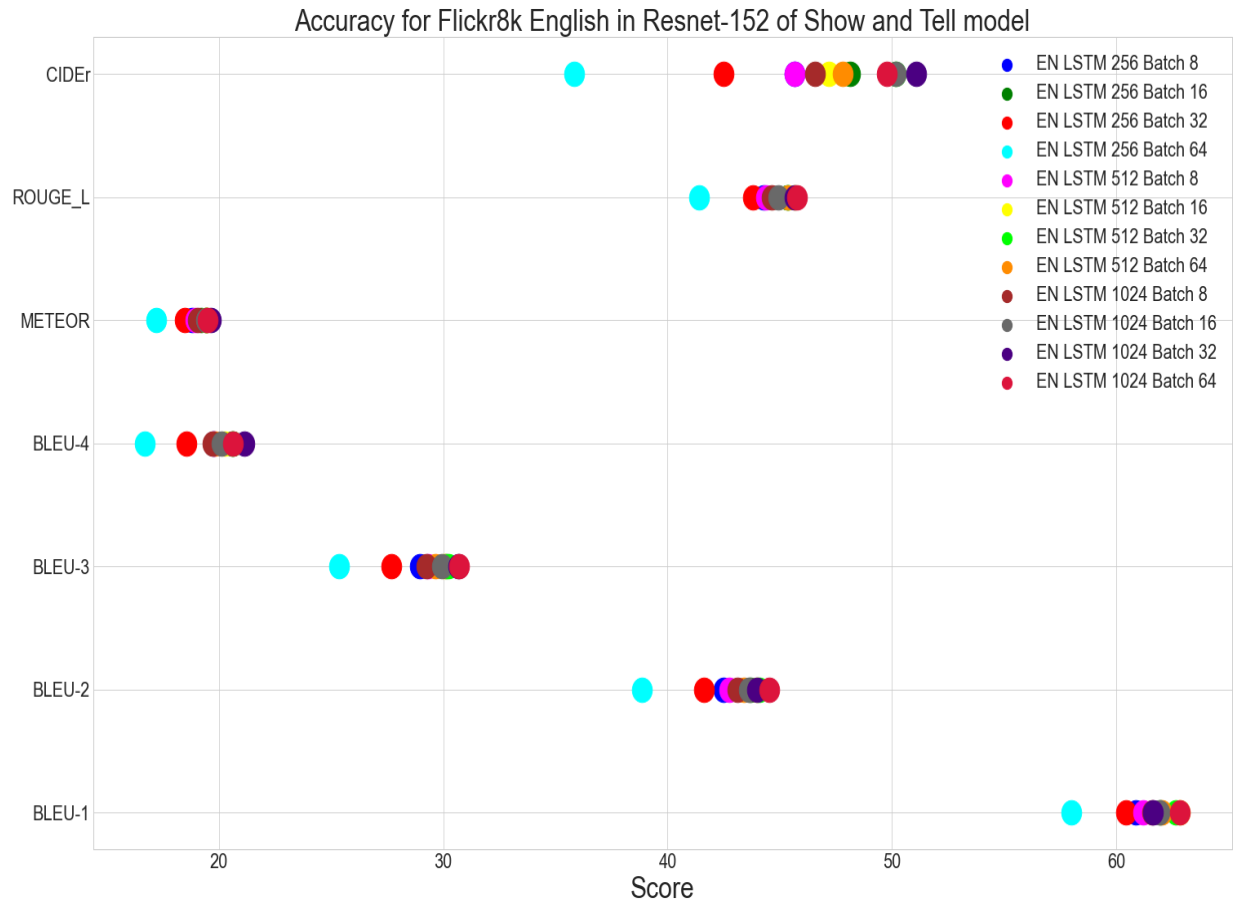


Figure 9: Accuracy for Flickr8k English in Resnet-152 of Show and Tell model

As for the Flickr8k Chinese language, the findings are given below

**Table 5.2.1(b): Comparative analysis for Flickr8K(Chinese) with Resnet-152 in Show and Tell model. Red, Blue, and Green colors represent the top, second top, and third top scores respectively.**

Hyperparameters			Flickr8K (Chinese)							
Encoder	LSTM Size	Batch Size	Loss	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE_L	CIDEr
Resnet-152	256	8	1.7188	68	50	39	29	30	60	88
		16	1.7859	68	48	37	27	30	60	84
		32	1.7642	69	50	38	28	30	60	88
		64	1.7971	67.47	47.67	36.26	26.26	29.71	59.6	82.85
	512	8	1.7094	68.69	51.11	40.58	31.07	30.49	60.66	93.01
		16	1.7363	68.74	49.59	38.12	28.05	30.54	60.33	90.69
		32	1.753	69.08	50.72	39.67	29.99	31.13	60.91	93.36
		64	1.7746	68.64	50.38	39.65	30.399	30.84	60.46	94.76
	1024	8	1.7438	69.91	51.03	39.54	29.29	31.01	61.12	91.06
		16	1.7719	68.06	49.25	38.01	28.28	30.12	59.69	90.07
		32	1.7791	70.33	52.15	40.93	30.59	31.02	61.17	93.45
		64	1.825	67.08	48.31	36.63	27.01	30.31	59.28	84.86

For the Chinese language, here we can see the same pattern of accuracy considering the LSTM size of 1024. However, in this case, we are getting some of the evaluation metrics' top accuracy at LSTM size 1024, and some are at the LSTM size 512. Also, a large change in the other evaluation matrices has been seen in the Flickr8k Chinese language. Here, also we do not get any considerable score in the smaller LSTM size 256.

These findings also presented in Figure 10 for a better understanding of the analysis

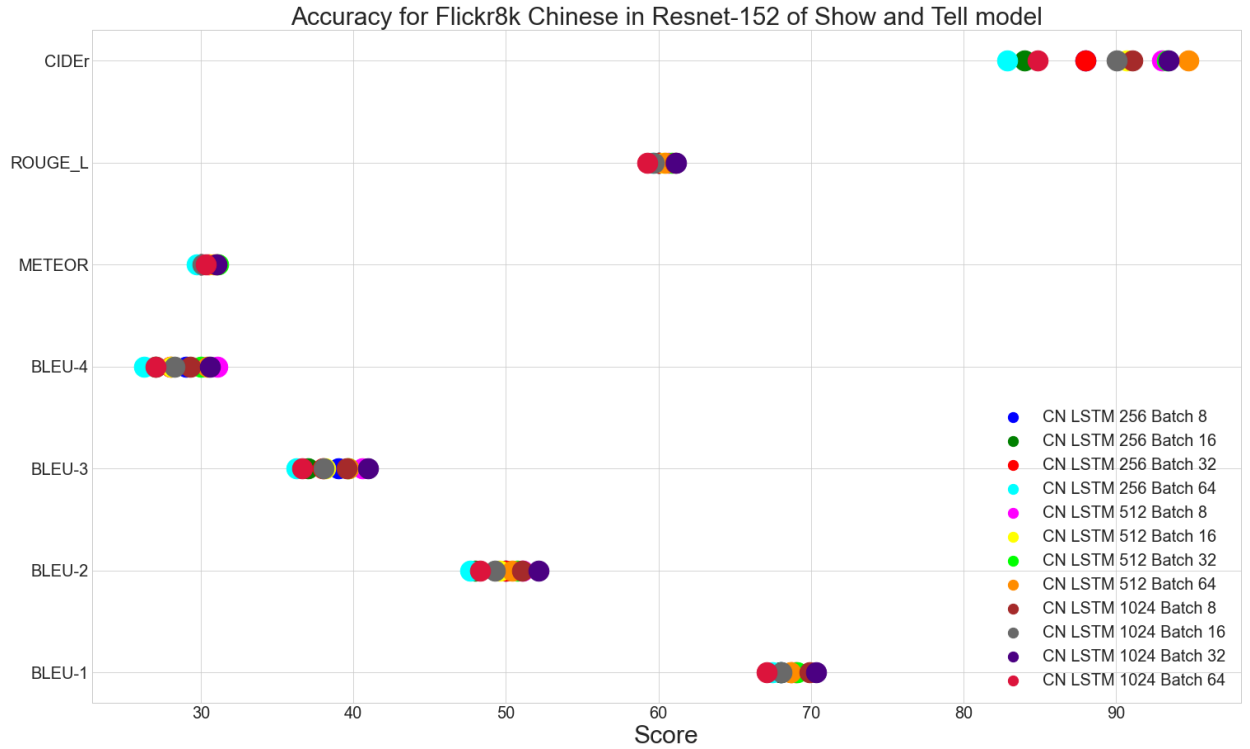


Figure 10: Accuracy for Flickr8k Chinese in Resnet-152 of Show and Tell model

### 5.2.2 VGG + LSTM:

For this analysis, we have used VGG [22] as the encoder and the LSTM [3] as the decoder. We have chosen the VGG16 rather than the VGG19 because the difference in the case of image feature extraction is negligible.

As we can see from Table 5.2.2(a), like the ResNet152, VGG also gives the highest accuracy for a large number of LSTM layers of 1024. However, the accuracy of the evaluation matrices is lower than the Resnet152. The reason behind this is that ResNet152 has a gradient descent flow that is absent in VGG16. Another problem with VGG is that it takes a longer period of time to complete its training, whereas ResNet is faster.

As for the Flickr8k English language, the findings are given below:

**Table 5.2.2(a): Comparative analysis for Flickr8K(English) with VGG16 in Show and Tell model. Red, Blue, and Green colors represent the top, second top, and third top scores respectively.**

Hyperparameters			Flickr8K(English)							
Encoder	LSTM Size	Batch Size	Loss	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE_L	CIDEr
VGG16	256	8	2.6082	58.29	39.11	25.47	16.683	17.407	42.04	37.53
		16	2.64384	57.05	37.92	24.26	15.35	17.46	41.93	36.21
		32	2.6635	58.05	38.67	25.07	16.16	17.48	42.11	36.49
		64	2.723	57.65	38.17	24.49	15.57	17.23	41.46	34.62
	512	8	2.5907	57.97	38.76	24.93	16.07	17.873	42.32	37.78
		16	2.64075	58.07	39.08	25.53	16.546	17.495	41.966	36.86
		32	2.63619	59.69	40.524	26.9	17.699	18.208	43.21	41.36
		64	2.67392	57.11	38.46	25.319	16.667	17.612	41.561	37.521
	1024	8	2.6096	59.75	41.075	27.659	18.5	18.411	43.99	42.448
		16	2.58709	60.39	41.88	28.14	18.69	18.55	43.84	43.3
		32	2.6124	60.25	41.31	27.55	18.473	18.5	43.72	41.819
		64	2.6466	58.42	39.64	26.33	17.203	17.926	42.54	41.383 7



These findings also presented in Figure 11 for a better understanding of the analysis

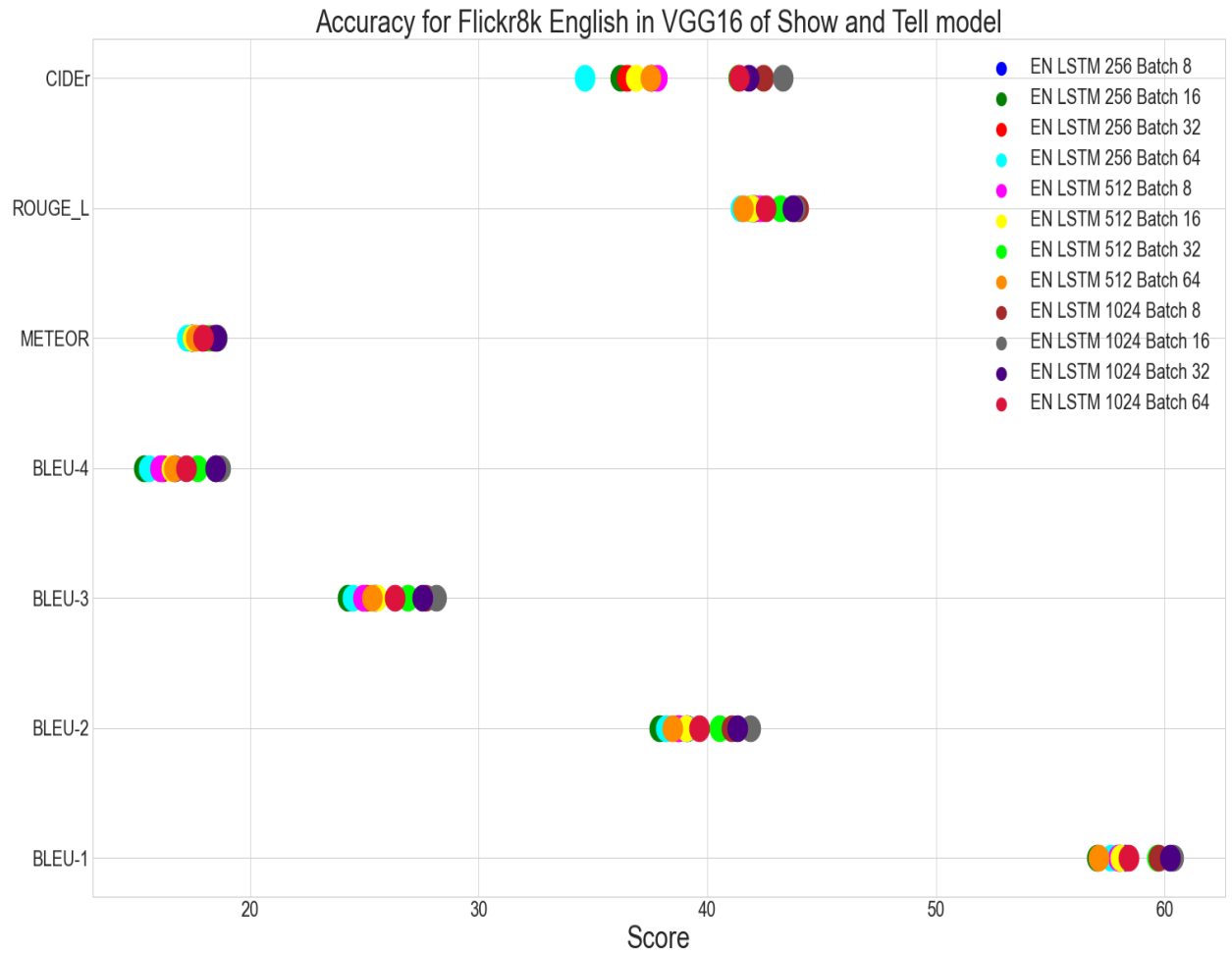


Figure 11: Accuracy for Flickr8k English in VGG16 of Show and Tell model

For the Flickr8k Chinese language, the findings are given below:

**Table 5.2.2(b): Comparative analysis for Flickr8K(Chinese) with VGG16 in Show and Tell model. Red, Blue, and Green colors represent the top, second top, and third top scores respectively.**

Hyperparameters			Flickr8K(Chinese)							
Encoder	LSTM Size	Batch Size	Loss	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE_L	CIDEr
VGG16	256	8	1.927	65	44.6	32.9	22.94	27.7	56.8	68.46
		16	1.912	63.9	43.1	31.7	22.2	27.6	56.2	67.3
		32	1.971	64.2	44.1	32.6	23.04	27.5	56.3	67
		64	1.978	63.9	43.4	32.1	22.7	27.62	56.1	68.55
	512	8	1.864	66.38	46.3	35	25.5	29	57.76	77.9
		16	1.894	66.2	46.01	34.3	24.5	28.6	57.76	76.42
		32	1.921	65.1	45	33.7	23.8	28.5	57.3	72.7
		64	1.949	64.4	43.2	31.35	21.8	27.92	56.1	67.922
	1024	8	1.829	66.18	46.2	34.7	24.8	28.8	58.3	76.9
		16	1.9429	65.15	45.6	33.7	23.7	28.4	57.3	75.5
		32	1.8937	66.4	46.65	35.2	25.3	29.24	58.3	76.9
		64	1.9255	66.9	47.09	35.52	25.7	28.8	58.2	78.24

For the Flickr8k (Chinese) [24] analysis with VGG16, here also we can see that a larger LSTM size gives higher accuracy. But as compared to the ResNet152 with the Chinese language, we are still getting lower scores in every evaluation metric with the VGG16.

These findings also presented in Figure 12 for a better understanding of the analysis

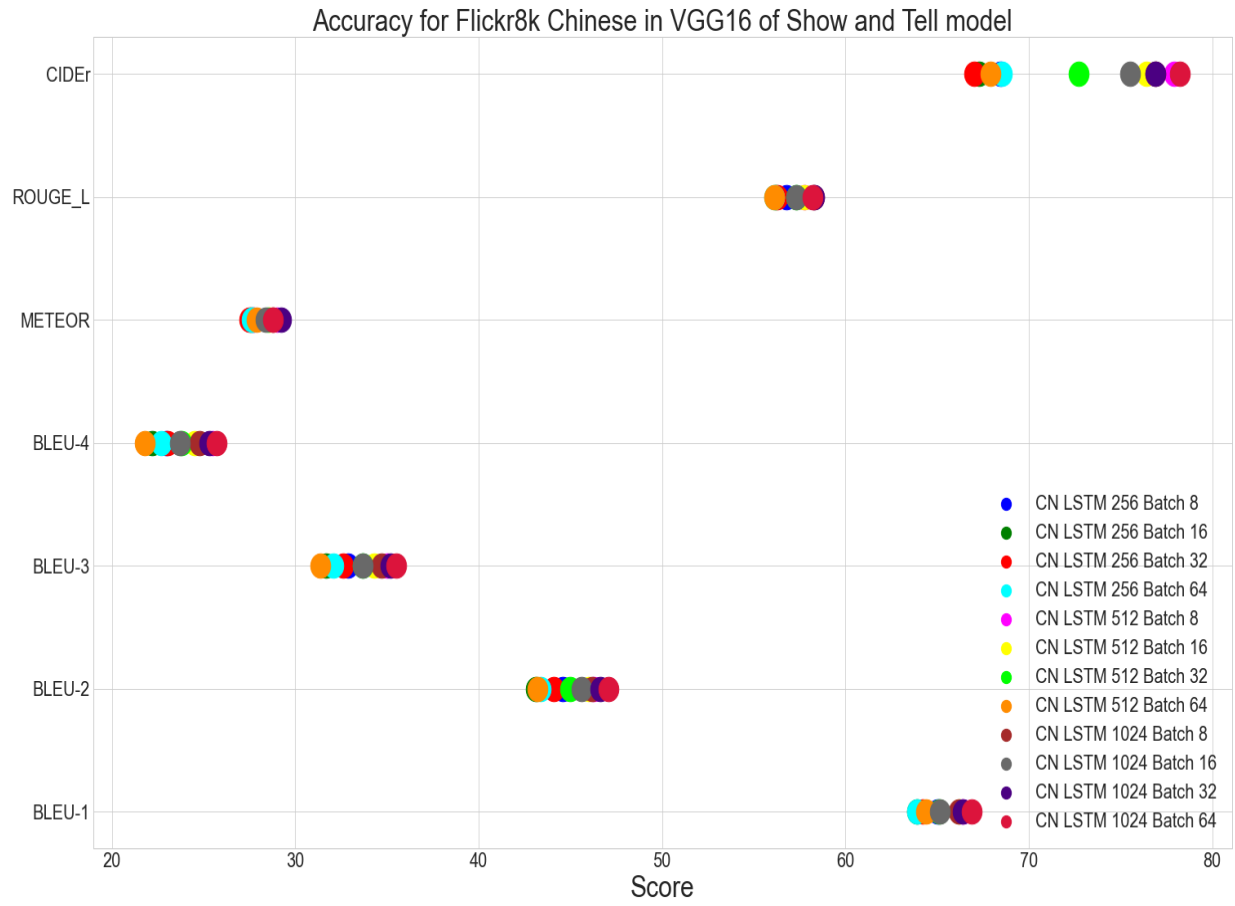


Figure 12: Accuracy for Flickr8k Chinese in VGG16 of Show and Tell model

## 5.4 Comparative Analysis on Show and Tell:

Combining all of our findings together, we have created some plots depicting differences in accuracy for different hyperparameter combinations.

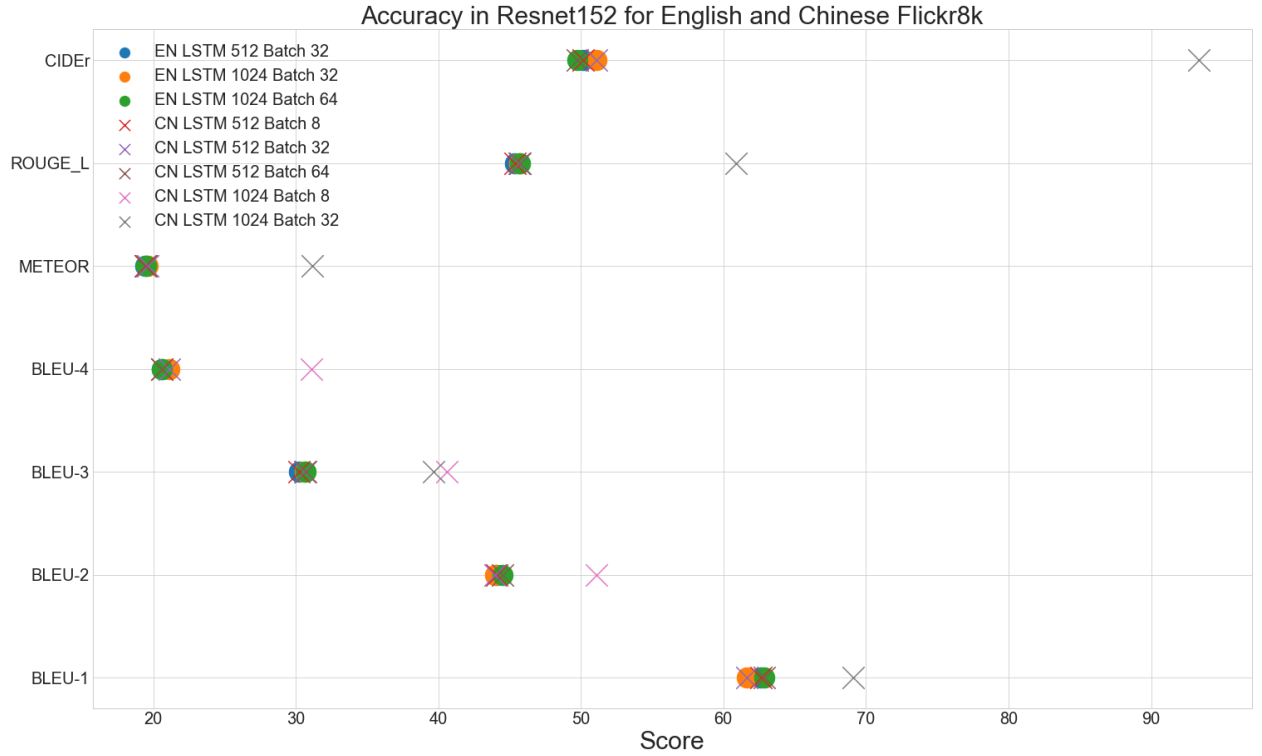


Figure 13: Accuracy in Resnet152 for English and Chinese Flickr8k

Figure 13 shows accuracy differences between English and Chinese languages based on different hyperparameter combinations in Resnet152. There are some noticeable changes in accuracy based on the language perspective. As we know, conveying messages is different in different parts of the world because of the language. Our analysis shows that BLEU (1-4) [30] accuracy is almost 10% higher than the English language.

Figure 14 shows accuracy differences between English and Chinese languages based on different hyperparameter combinations in VGG16. Here also, we can see the higher accuracy for a larger LSTM size of 1024.

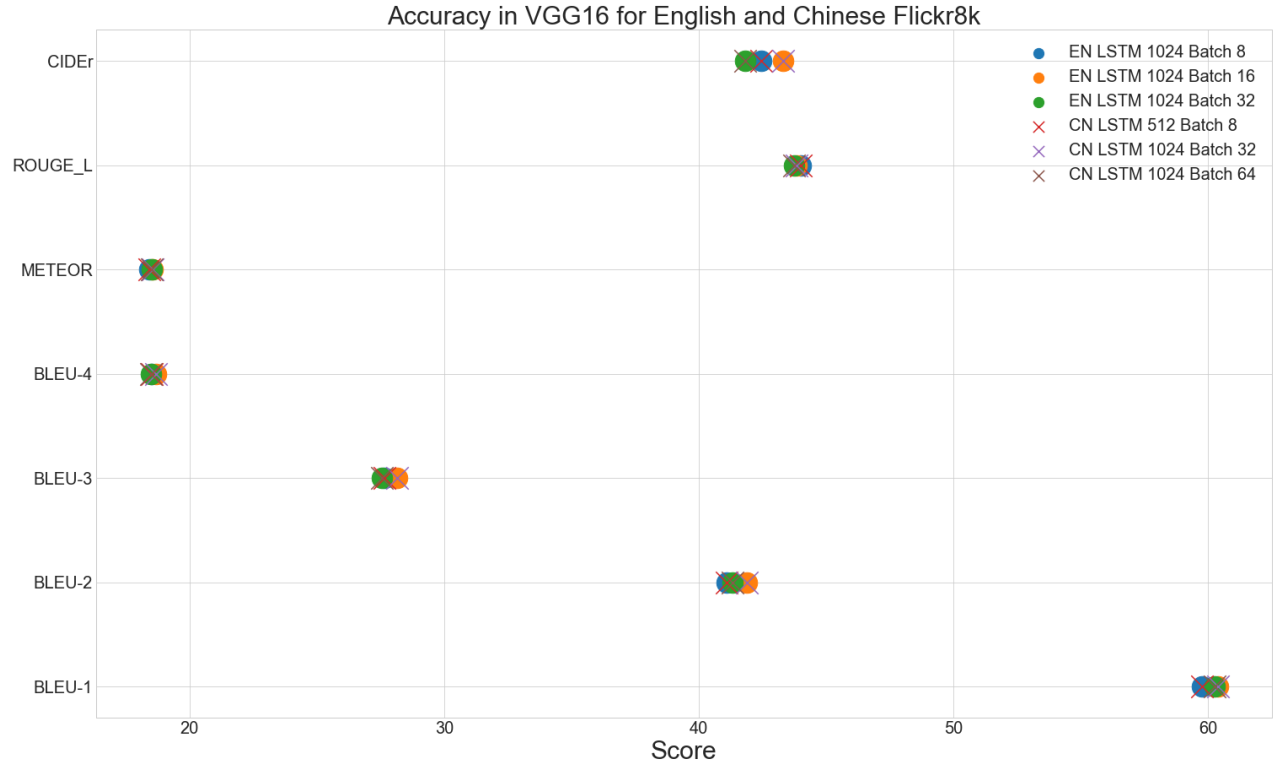


Figure 14: Accuracy in VGG16 for English and Chinese Flickr8k

The difference of the top highest and smallest accuracy score for VGG16 and Resnet-152 in the Show and Tell model is given below to evaluate the accuracy based on different evaluation metrics.

**Table 5.4(a): Difference of top highest and smallest accuracy score for VGG16 and Resnet-152 in Show and Tell model**

Encoder	Flickr8K(English)							Flickr8K (Chinese)						
	B-1	B-2	B-3	B-4	METEOR	ROUGE_L	CIDEr	B-1	B-2	B-3	B-4	METEOR	ROUGE_L	CIDEr
VGG16	3.34	3.96	3.88	3.34	1.32	2.53	8.68	3	3.99	4.17	3.9	1.74	2.2	11.24
Resnet-152	4.84	5.67	5.33	4.45	2.43	4.355	15.27	3.25	4.48	4.67	4.81	1.42	1.89	11.91

We also plotted the effects on evaluation metrics in the Show and Tell model, to visualize how the evaluation metrics react to different languages and different encoder models.

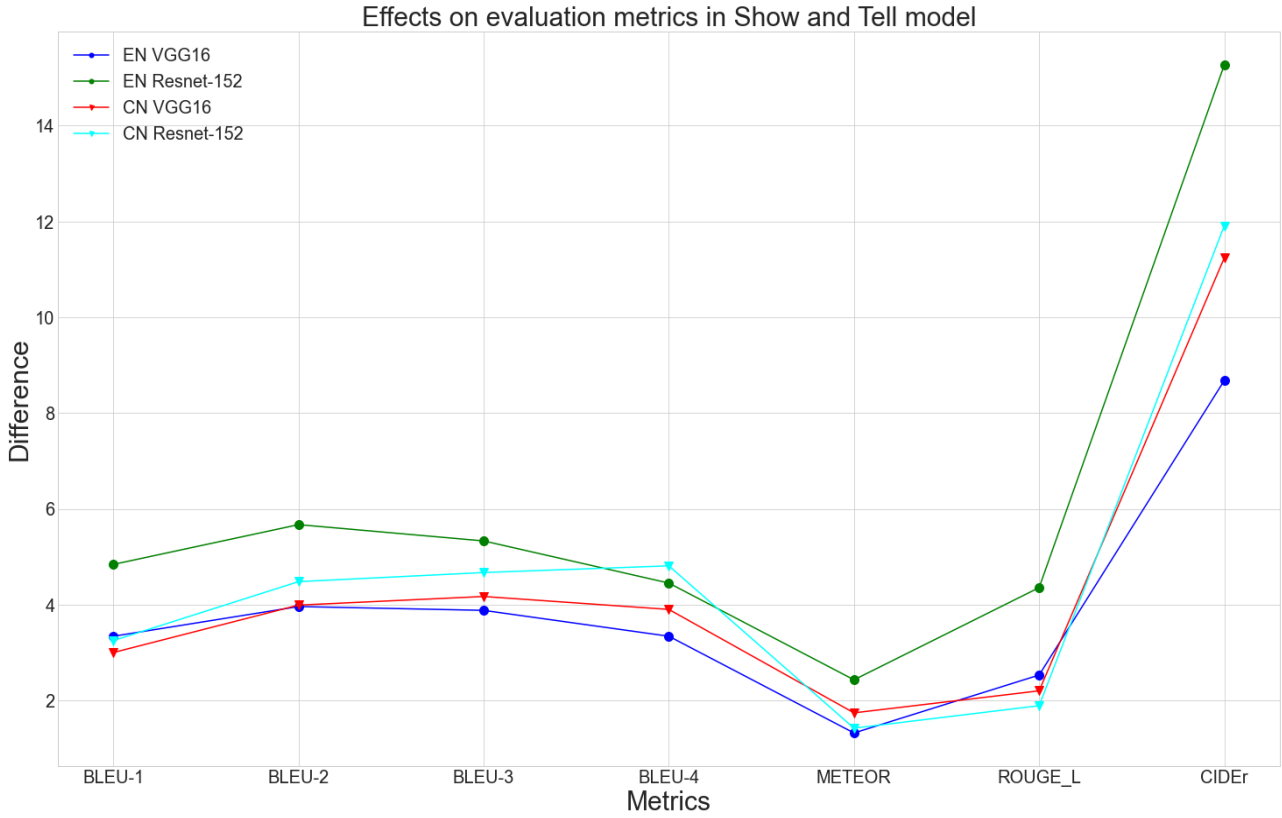


Figure 15: Effects on evaluation metrics in Show and Tell model

Now for the loss in different languages and encoder models, the plot is given below.

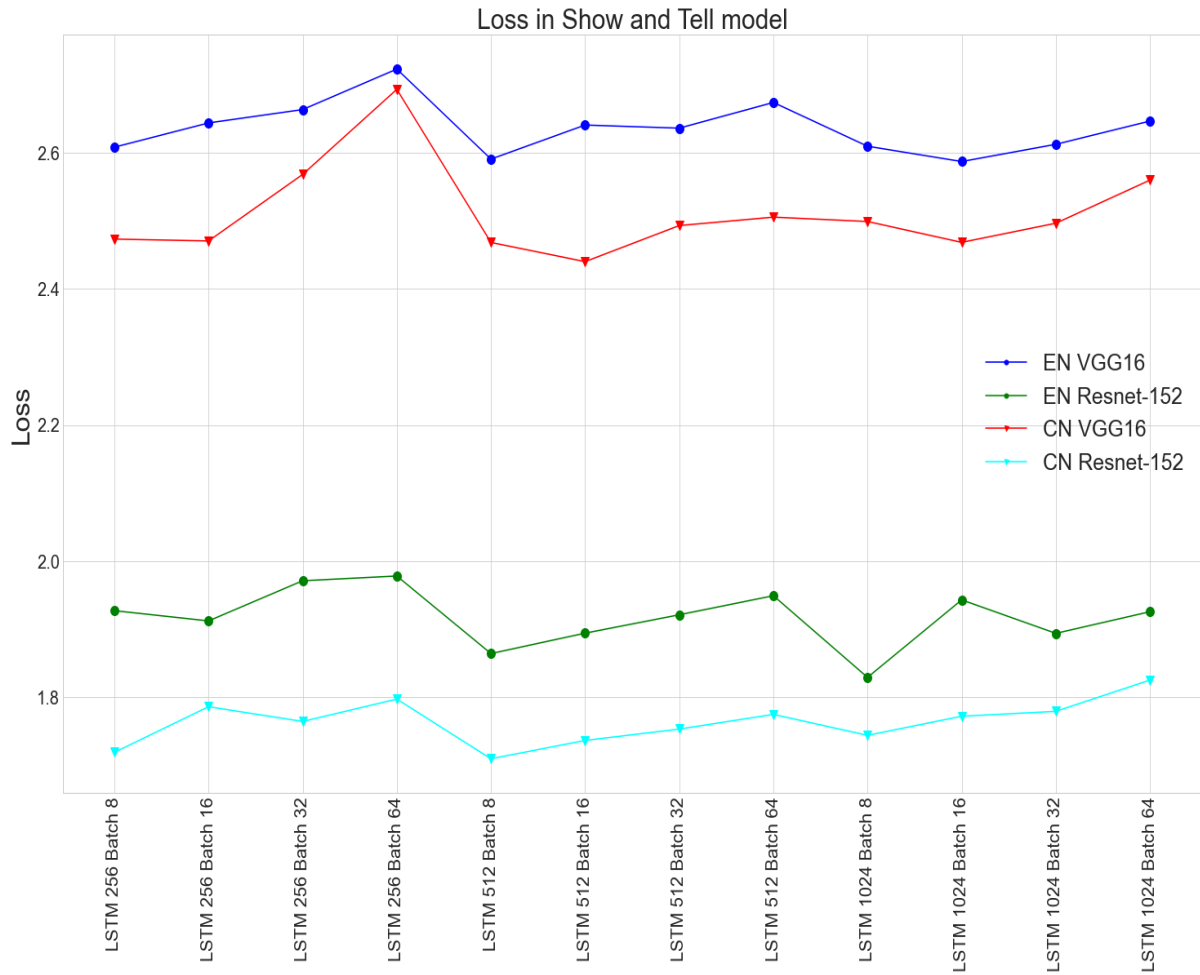


Figure 16: Loss in Show and Tell model

For the Bangla language, we have done comparative analysis using the Chittrion [4] dataset based on VGG16 [22], VGG19 [22], and ResNet152 [6] encoder architecture. With these encoders, we have used LSTM [3] as the decoder. We have tested different kinds of vocab building methods because commonly used NLTK [25] does not get the Bangla character accurately. For the optimizer, along with Adam [23], we also used the SGD optimizer.

Our findings are given below:


**Table 5.4(b): Comparative analysis for Chittrion dataset in Show and Tell model. Red color represents the top scores.**

Encoder	Vocab	Optimizer	Chittrion							
			Loss	B-1	B-2	B-3	B-4	METEOR	ROUGE_L	CIDEr
VGG16	NLTK	Adam	0.65	45.27	29.98	18.4	11.8	28.7	44.3	46.5
VGG16	NLTK	SGD	0.91	44.9	30.5	15.8	8.6	24.6	42.6	25.4
VGG19	NLTK		0.68	44.1	30.07	17.6	10.7	25.6	44.29	37.68
Resnet-152 (Edited captions)	NLTK		0.68	42.3	26.1	15.2	9.8	27.4	39.9	52.68
Resnet-152	BNLP		0.67	<b>49.1</b>	33.5	20.8	13.5	<b>28.7</b>	<b>48.2</b>	55.4
Resnet-152 (Basic)	BNLP		0.62	49.01	<b>34.02</b>	<b>21.58</b>	<b>14.3</b>	28.4	48.09	<b>58.2</b>

Here, all of this training is done with an LSTM size of 512, and BNLP is a vocab building method mainly used for the Bangla language. Here, we can see the same pattern of using ResNet architecture with an LSTM size of 512.

We also compared our Bangla image captioning model against the original implementation on the Chittrion dataset. Some of the captions are given side by side for comparison.

**Table 5.4(c): Caption analysis for the Chittrion dataset between Original Chittrion Implement and Our Model**

Image	Chittrion	Our Model
	<p><b>Human Caption:</b> হলুদ সরিষা ক্ষেতের মাঝে দিয়ে হেটে যাচ্ছে একজন মহিলা</p> <p><b>Model Caption:</b> সরিষা ক্ষেতের মাঝে একজন মহিলা দাড়িয়ে আছে</p> <p><b>BLEU: 0.11</b></p>	<p><b>Human Caption:</b> হলুদ সরিষা ক্ষেতের মাঝে দিয়ে হেটে যাচ্ছে একজন মহিলা</p> <p><b>Model Caption:</b> সরিষা ক্ষেতের মাঝ দিয়ে একজন মানুষ হেঁটে যাচ্ছে ।</p> <p><b>BLEU: 0.596</b></p>



	<p><b>Human Caption:</b> জমির পাশ দিয়ে ২টি হালের গরু নিয়ে হেটে জাচ্ছেলুঙ্গি পরা একজন পুরুষ মানুষ</p> <p><b>Model Caption:</b> একজন লোক গরু নিয়ে যাচ্ছে</p>	<p><b>Human Caption:</b> জমির পাশ দিয়ে ২টি হালের গরু নিয়ে হেটে জাচ্ছেলুঙ্গি পরা একজন পুরুষ মানুষ</p> <p><b>Model Caption:</b> একজন পুরুষ ও একটি গরু আছে।</p>
	<p><b>BLEU: 0.029</b></p>	<p><b>BLEU: 0.182</b></p>
	<p><b>Human Caption:</b> একজন লোক রাস্তায় দাঁড়িয়ে ব্যায়াম করছে আর আশে পাশে কয়েকটা কাক দাঁড়িয়ে আছে</p> <p><b>Model Caption:</b> ২ জন বাচ্চা ছেলে পাড় পাড় পাড় দিয়ে</p>	<p><b>Human Caption:</b> একজন লোক রাস্তায় দাঁড়িয়ে ব্যায়াম করছে আর আশে পাশে কয়েকটা কাক দাঁড়িয়ে আছে</p> <p><b>Model Caption:</b> কয়েকজন মানুষ আছে।</p>
	<p><b>BLEU: 0.0</b></p>	<p><b>BLEU: 0.04</b></p>

As we can see from some of the images and their comparisons, our model gives better accuracy and better descriptions. Here, we have presented the BLEU-1 score for both of these models. This is because we have used the ResNet-152 as the encoder for training our model instead of VGG16/19. And for the vocab builder, BNLP was used, which is better in terms of recognizing Bangla words.

## 5.5 Future Direction:

We have started our training with the same hyperparameter combinations for the Show, Attend and Tell model [2]. So far, we have done only 12 combinations with the VGG16 [22] on the English language of the Flickr8k [8] dataset. In the future, we want to use one other model based on Transformer [7], which uses the concept of multi-head. Also, we will use two other encoder architectures, Densnet-161 [34] and Inception V3 [35].

## 5.6 Conclusion:

Our project reports that a larger LSTM size in the decoder and a larger batch size gives better performance for both languages. From the language perspective, there is less difference in choosing a hyperparameter to tune. However, the decoder has a more significant effect than the encoder. There is a possibility of overfitting because the dataset is small. So, the continuation of work can be done for a larger dataset, though it might be challenging to find a larger dataset for other languages. The project can be further explored for other image captioning state-of-the-art models like Neural Baby Talk [36], which requires visual bounding boxes.

## REFERENCES

---

- [1] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [2] K. Xu *et al.*, “Show, attend and tell: Neural image caption generation with visual attention,” *arXiv [cs.LG]*, 2015.
- [3] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [4] M. Rahman, N. Mohammed, N. Mansoor, and S. Momen, “Chittron: An automatic Bangla image captioning system,” *Procedia Comput. Sci.*, vol. 154, pp. 636–642, 2019.
- [5] A. Patel and A. Varier, “Hyperparameter Analysis for Image Captioning,” *arXiv [cs.CV]*, 2020.
- [6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [7] A. Vaswani *et al.*, “Attention is all you need,” *arXiv [cs.CL]*, 2017.
- [8] M. Hodosh, P. Young, and J. Hockenmaier, “Framing image description as a ranking task: Data, models and evaluation metrics,” *J. Artif. Intell. Res.*, vol. 47, pp. 853–899, 2013.
- [9] Y. Wu *et al.*, “Demystifying learning rate policies for high accuracy training of deep neural networks,” in *2019 IEEE International Conference on Big Data (Big Data)*, 2019.
- [10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE Inst. Electr. Electron. Eng.*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [11] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, “Don’t decay the learning rate, increase the batch size,” *arXiv [cs.LG]*, 2017.
- [12] Y. Wu *et al.*, “A comparative measurement study of deep learning as a service framework,” *IEEE trans. serv. comput.*, pp. 1–1, 2019.
- [13] Y. Jia *et al.*, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the ACM International Conference on Multimedia - MM ’14*, 2014.

- [14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.
- [15] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," *arXiv [cs.DC]*, 2016.
- [16] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," *arXiv [cs.LG]*, 2019.
- [17] The Theano Development Team *et al.*, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv [cs.SC]*, 2016.
- [18] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions," *Trans. Assoc. Comput. Linguist.*, vol. 2, pp. 67–78, 2014.
- [19] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik, "Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models," *Int. J. Comput. Vis.*, vol. 123, no. 1, pp. 74–93, 2017.
- [20] R. Mottaghi *et al.*, "The role of context for object detection and semantic segmentation in the wild," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
- [21] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Computer Vision – ECCV 2014*, Cham: Springer International Publishing, 2014, pp. 740–755.
- [22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv [cs.CV]*, 2014.
- [23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv [cs.LG]*, 2014.
- [24] X. Li, W. Lan, J. Dong, and H. Liu, "Adding Chinese captions to images," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval - ICMR '16*, 2016.
- [25] E. Loper and S. Bird, "NLTK: The natural language toolkit," in *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics -*, 2002.
- [26] J. L. Junfeng Jiang, *Constructing Financial Sentimental Factors in Chinese Market Using Natural Language Processing*. arXiv:1809.08390, 2018.
- [27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [28] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

- [29] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, 2015.
- [30] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, 2001.
- [31] A. Lavie and A. Agarwal, "Meteor: An automatic metric for MT evaluation with high levels of correlation with human judgments," in *Proceedings of the Second Workshop on Statistical Machine Translation - StatMT '07*, 2007.
- [32] R. Vedantam, C. L. Zitnick, and D. Parikh, "CIDEr: Consensus-based image description evaluation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [33] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," *In Text Summarization Branches Out*, pp. 74–81, Jul. 2004.
- [34] H. G. & L. Z. & van der Maaten Laurens & Weinberger Kilian, "Densely Connected Convolutional Networks," *CVPR*, 2017.
- [35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [36] J. Lu, J. Yang, D. Batra, and D. Parikh, "Neural Baby Talk," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.