A PROJECT REPORT ON

NEURAL IMAGE CAPTION GENERATION WITH DEEP LEARNING AND COMPUTER VISION

SUBMITTED IN PARTIAL FULFILLMENT FOR AWARD DEGREE OF

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

HRISHABH RAJ

(1906943 / 338)

UNDER THE GUIDANCE OF

ER. HARJASDEEP SINGH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MALOUT INSTITUTE OF MANAGEMENT & INFORMATION TECHNOLOGY, MALOUT

Aug-Dec 2022

DECLARATION

I hereby declare that the project entitled **“Image Caption Generation with Deep Learning and Computer Vision”** submitted for the B.Tech. CSE 7th is my original work and the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

HRISHABH RAJ (Signature)

Place: MIMIT MALOUT

Date:______________

CERTIFICATE

We hereby certify that the work which is being presented in the Project-I report entitled **“Image Caption Generation with Deep Learning and Computer Vision”** by me in partial fulfilment of requirements for the award of degree of B.Tech. (CSE) submitted in the Department of Computer Science & Engineering at **MIMIT Malout** under **IKG PUNJAB TECHNICAL UNIVERSITY, KAPURTHALA** is an authentic record of my/our own work carried out during a period from “17TH AUG 2022” to “19TH NOV 2022”. The matter presented in this Project report has not been submitted by me in any other University / Institute for the award of any Degree or Diploma.

Signature of the Student/s

HRISHABH RAJ (1906943 / 338)

This is to certify that the above statement made by the candidate/s is correct to the best of my knowledge.

Signature of the Project-I Guide

Er. HARJASDEEP SINGH

Assistant Professor (CSE)

ABSTRACT

We see an image and our brain can easily tell what the image is about, but can a computer tell what the image is representing? Computer vision researchers worked on this a lot and they considered it impossible until now! With the advancement in Deep learning techniques, availability of huge datasets and computer power, we can build models that can generate captions for an image.

This is what we are going to implement in this Python based project where we will use deep learning techniques of Convolutional Neural Networks and a type of Recurrent Neural Network (LSTM) together.

Image caption generation is a task that involves computer vision and natural language processing concepts to recognize the context of an image and describe them in a natural language like English.

In this Python project, we will be implementing the caption generator using **CNN** (Convolutional Neural Networks) and **LSTM** (Long short-term memory). The image features will be extracted from Residual Network Model (Res-Net50) which is a CNN model trained on the **imagenet dataset (Flicker_8D)** and then we feed the features into the LSTM model which will be responsible for generating the image captions.

ACKNOWLEDGEMENT

The authors are highly grateful to the **Dr. JASKARAN SINGH BHULLAR Director, MIMIT Malout**, for providing this opportunity to carry out the present Project-I work.

The constant guidance and encouragement received from**, Dr. SONIA SHARMA**, **HEAD CSE DEPARTMENT, MIMIT Malout**, has been of great help in carrying out the present work and is acknowledged with reverential thanks.

The authors would like to express a deep sense of gratitude and thanks profusely to **Er. HARJASDEEP SINGH**, who was our Project guide. Without the wise counsel and able guidance, it would have been impossible to complete the project in this manner.

The author express gratitude to other faculty members of Computer Science & Engineering Department, MIMIT Malout for their intellectual support throughout the course of this work.

Finally, the authors are indebted to all whosoever have contributed in this Project work.

Name of the Student/s

Hrishabh Raj

TABLE OF CONTENS

LIST OF TABLES

LIST OF FIGURES

CHAPTER - 1: INTRODUCTION

Every day, we encounter a large number of images from various sources such as the internet, news articles, document diagrams and advertisements. These sources contain images that viewers would have to interpret themselves. Most images do not have a description, but the human can largely understand them without their detailed captions. However, machine needs to interpret some form of image captions if humans need automatic image captions from it.

Image captioning is important for many reasons. Captions for every image on the internet can lead to faster and descriptively accurate images searches and indexing.

Ever since researchers started working on object recognition in images, it became clear that only providing the names of the objects recognized does not make such a good impression as a full human-like description. As long as machines do not think, talk, and behave like humans, natural language descriptions will remain a challenge to be solved.

Image captioning has various applications in various fields such as biomedicine, commerce, web searching and military etc. Social media like Instagram, Facebook etc can generate captions automatically from images.

Image caption generation is a task that involves computer vision and natural language processing concepts to recognize the context of an image and describe them in a natural language like English. The goal of image captioning is to convert a given input image into a natural language description.

1.1 MOTIVATION

Whenever an image appears in front of us our brain is capable of annotating or labelling it. But, what about computers? How can a machine process an image and label it with a highly relevant and accurate caption? It seemed quite impossible a few years back, but now with the enhancement of Computer Vision and Deep learning algorithms, availability of relevant datasets, and AI models, it becomes easier to build a relevant caption generator for an image. Even Caption generation is becoming a growing business in the world, and many data annotation firms are earning billions from this. In this guide, we are going to build one such annotation tool which is capable of generating very relevant captions for the image with the help of datasets.

Generating captions for images is a vital task relevant to the area of both Computer Vision and Natural Language Processing. Mimicking the human ability of providing descriptions for images by a machine is itself a remarkable step along the line of Artificial Intelligence. The main challenge of this task is to capture how objects relate to each other in the image and to express them in a natural language (like English). Traditionally, computer systems have been using pre-defined templates for generating text descriptions for images. However, this approach does not provide sufficient variety required for generating lexically rich text descriptions. This shortcoming has been suppressed with the increased efficiency of neural networks. Many state of art models use neural networks for generating captions by taking image as input and predicting next lexical unit in the output sentence. This is what we are going to implement in this Python based project where we will be implementing the caption generator using **CNN** (Convolutional Neural Networks) and **LSTM** (Long short-term memory).

1.2 General Introduction:

Generating accurate captions for an image has remained as one of the major challenges in Artificial Intelligence with plenty of applications ranging from robotic vision to helping the visually impaired. Long term applications also involve providing accurate captions for videos in scenarios such as security system. “Image caption generator”: the name itself suggests that we aim to build an optimal system which can generate semantically and grammatically accurate captions for an image. Researchers have been involved in finding an efficient way to make better predictions, therefore we have discussed a few methods to achieve good results. We have used the deep neural networks and machine learning techniques to build a good model. We have used

Flickr 8k dataset which contains around 8000 sample images with their five captions for each image.

There are two phases, feature extraction from the image using Convolutional Neural Networks (CNN) and generating sentences in natural language based on the image using Recurrent Neural Networks (RNN).

For our image-based model– we used **CNN**,

and for language-based model — we used **LSTM** (RNN).

This model consists of Convolutional Neural Network (CNN) as well as Recurrent Neural Network (RNN). The CNN is used for feature extraction from image and RNN is used for sentence generation. The model is trained in such a way that if input image is given to model it generates captions which nearly describes the image.

CHAPTER - 2: LITERATURE SURVEY

2.1 Related Work:

There have been several attempts at providing a solution to this problem including template-based solutions which used image classification i.e. assigning labels to objects from a fixed set of classes and inserting them into a sample template sentence. But more recent work has focused on Recurrent Neural Networks. RNNs are already quite popular with several Natural Language Processing tasks such as machine translation where a sequence of words is generated. Image caption generator extends the same application by generating a description for an image word by word.

The computer vision reads an image considering it as a two-dimensional array. Therefore, we describe image captioning as a language translation problem. Previously language translation was complicated and included several different tasks but the recent work has shown that the task can be achieved in a much efficient way using Recurrent Neural Networks. But, regular RNNs suffer from the vanishing gradient problem which was vital in case of our application. The solution for the problem is to use LSTMs and GRUs which contain internal mechanisms and logic gates that retain information for a longer time and pass only useful information.

Image captioning has recently gathered a lot of attention specifically in the natural language domain. There is a pressing need for context based natural language description of images, however, this may seem a bit farfetched but recent developments in fields like neural networks, computer vision and natural language processing has paved a way for accurately describing images i.e. representing their visually grounded meaning. We are leveraging state-of-the-art techniques like Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and appropriate datasets of images and their human perceived description to achieve the same. We demonstrate that our alignment model produces results in retrieval experiments on datasets such as Flicker.

2.2 IMAGE CAPTIONING PROCESS: -

Image Captioning is the process of generating textual description of an image. It uses both Natural Language Processing and Computer Vision to generate the captions. Image captioning is a popular research area of Artificial Intelligence (AI) that deals with image understanding and a language description for that image. Image understanding needs to detect and recognize objects. It also needs to understand scene type or location, object properties and their interactions. Generating well-formed sentences requires both syntactic and semantic understanding of the language. Understanding an image largely depends on obtaining image features. For example, they can be used for automatic image indexing. Image indexing is important for Content-Based Image Retrieval (CBIR) and therefore, it can be applied to many areas, including biomedicine, commerce, the military, education, digital libraries, and web searching. Social media platforms such as Facebook and Twitter can directly generate descriptions from images. The descriptions can include where we are (e.g., beach, cafe), what we wear and importantly what we are doing there. **Techniques:-** The techniques used for this purpose can be broadly divided into two categories: (1) Traditional machine learning based techniques and (2) Deep machine learning based techniques.

In traditional machine learning, hand crafted features such as Local Binary Patterns (LBP), Scale-Invariant Feature Transform (SIFT), the Histogram of Oriented Gradients (HOG), and a combination of such features are widely used. In these techniques, features are extracted from input data. They are then passed to a classifier such as Support Vector Machines (SVM) in order to classify an object. Since hand crafted features are task specific, extracting features from a large and diverse set of data is not feasible. Moreover, real world data such as images and video are complex and have different semantic interpretations.

On the other hand, in deep machine learning based techniques, features are learned automatically

from training data and they can handle a large and diverse set of images and videos. For example, Convolutional Neural Networks (CNN) are widely used for feature learning, and a classifier such as SoftMax is used for classification. CNN is generally followed by Recurrent Neural Networks (RNN) or Long Short-Term Memory Networks (LSTM) in order to generate captions. Deep learning algorithms can handle complexities and challenges of image captioning quite well.

The previous image captioning methods can generate only one caption for the whole image. Here is a proposed image captioning method called Dense Captioning. This method localizes all the salient regions of an image and then it generates descriptions for those regions.

A typical method of this category has the following steps:

(1) Region proposals are generated for the different regions of the given image.

(2) CNN is used to obtain the region-based image features.

(3) The outputs of Step 2 are used by a language model (LSTM) to generate captions for every region.

A block diagram of a typical dense captioning method is given in Figure below :

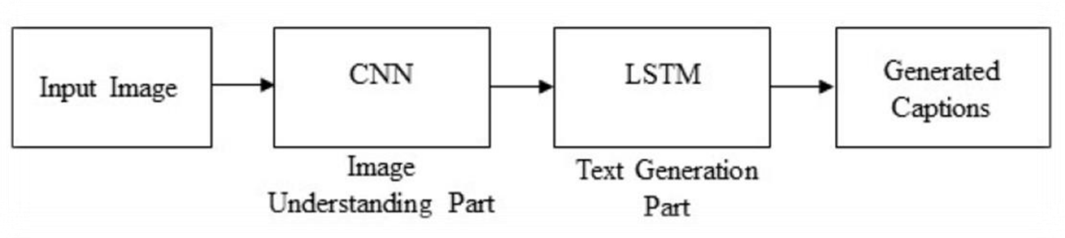


Figure.2.1. A block diagram of simple Encoder-Decoder architecture-based image captioning.

Image captioning intersects computer vision and natural language processing (NLP) research. NLP tasks, in general, can be formulated as a sequence-to-sequence learning.

Several neural language models such as neural probabilistic language model, log-bilinear models, skip-gram models, and recurrent neural networks (RNNs) have been proposed for learning sequence to sequence tasks. RNNs have widely been used in various sequence learning tasks. However, traditional RNNs suffer from vanishing and exploding gradient problems and cannot adequately handle long-term temporal dependencies.

LSTM networks are a type of RNN that has special units in addition to standard units. LSTM units use a memory cell that can maintain information in memory for long periods of time. In recent years, LSTM based models have dominantly been used in sequence-to-sequence learning tasks. Another network, Gated Recurrent Unit (GRU) has a similar structure to LSTM but it does not use separate memory cells and uses fewer gates to control the flow of information.

CHAPTER - 3: METHODOLOGY

PROBLEM FORMULATION

3.1 PROBLEM IDENTIFICATION

Despite the successes of many systems based on the Recurrent Neural Networks (RNN) many issues remain to be addressed. Among those issues the following two are prominent for most systems.

1. The Vanishing Gradient Problem.

2. Training an RNN is a very difficult task.

A recurrent neural network is a deep learning algorithm designed to deal with a variety of complex computer tasks such as object classification and speech detection. RNNs are designed to handle a sequence of events that occur in succession, with the understanding of each event based on information from previous events.

Ideally, we would prefer to have the deepest RNNs so they could have a longer memory period and better capabilities. These could be applied for many real-world use-cases such as stock prediction and enhanced speech detection. However, while they sound promising, RNNs are rarely used for real-world scenarios because of the vanishing gradient problem.

3.1.1 THE VANISHING GRADIENT PROBLEM:

This is one of the most significant challenges for RNNs performance. In practice, the architecture of RNNs restricts its long-term memory capabilities, which are limited to only remembering a few sequences at a time. Consequently, the memory of RNNs is only useful for shorter sequences and short time-periods. Vanishing Gradient problem arises while training an Artificial Neural Network. This mainly occurs when the network parameters and hyperparameters are not properly set. The vanishing gradient problem restricts the memory capabilities of traditional RNNs—adding too many time-steps increases the chance of facing a gradient problem and losing information when you use backpropagation.

3.2 PROPOSED WORK:

The main aim of this project is to get a little bit of knowledge of deep learning techniques. We use two techniques mainly CNN and LSTM for image classification.

So, to make our image caption generator model, we will be merging these architectures. It is also called a CNN-RNN model.

- CNN is used for extracting features from the image. We will use the pre-trained Residual Network model Res-Net50.
- LSTM will use the information from CNN to help generate a description of the image.

3.3 CONVOLUTIONAL NEURAL NETWORK:

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms.

Convolutional Neural networks are specialized deep neural networks which can process the data that has input shape like a 2D matrix. Images are easily represented as a 2D matrix and CNN is very useful in working with images.

It scans images from left to right and top to bottom to pull out important features from the image and combines the feature to classify images. It can handle the images that have been translated, rotated, scaled and changes in perspective.

3.4 LONG SHORT TERM MEMORY (LSTM):

LSTM stands for Long short-term memory, they are a type of RNN (recurrent neural network) which is well suited for sequence prediction problems. Based on the previous text, we can predict what the next word will be. It has proven itself effective from the traditional RNN by overcoming the limitations of RNN which had short term memory. LSTM can carry out relevant information throughout the processing of inputs and with a forget gate, it discards non-relevant information.

LSTMs are designed to overcome the vanishing gradient problem and allow them to retain information for longer periods compared to traditional RNNs. LSTMs can maintain a constant error, which allows them to continue learning over numerous time-steps and backpropagate through time and layers.

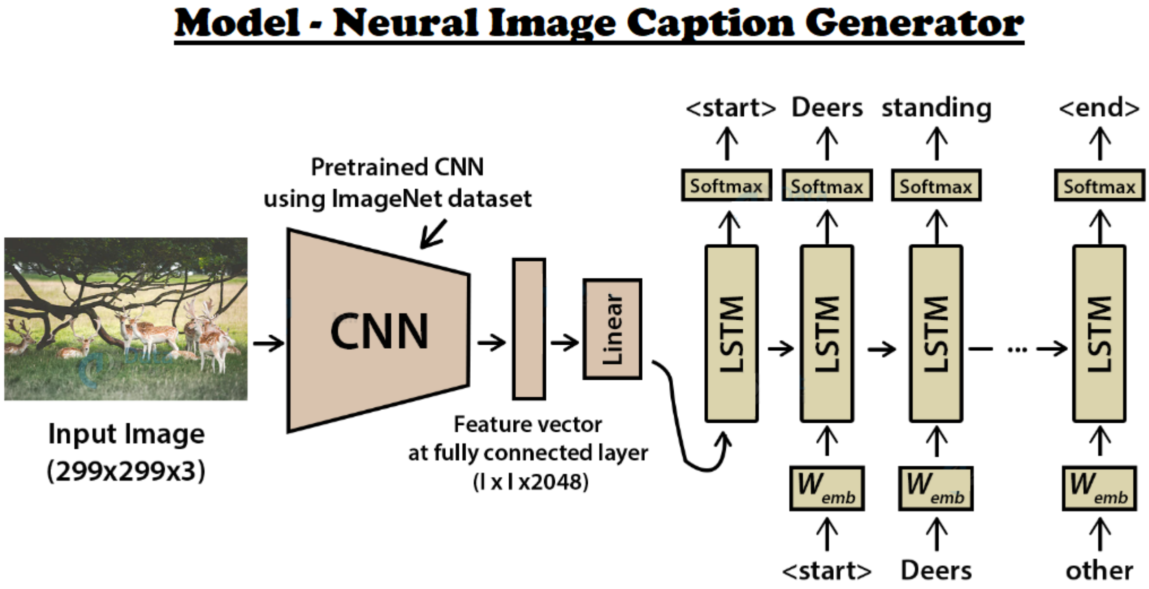


Fig 3.1. Methodology for Image Caption Generation using Neural Networks

Proposed working methodology are following as under.

This project requires a dataset which have both images and their caption. The dataset should be able to train the image captioning model.

3.5 FLICKR8K DATASET: Flickr8k dataset is a public benchmark dataset for image to sentence description. This dataset consists of 8000 images with five captions for each image. These images are extracted from diverse groups in Flickr website. Each caption provides a clear description of entities and events present in the image. The dataset depicts a variety of events and scenarios and doesn’t include images containing well-known people and places which makes the dataset more generic. The dataset has 6000 images in training dataset, 1000 images in development dataset and 1000 images in test dataset.

Features of the dataset making it suitable for this project are:

- Multiple captions mapped for a single image makes the model generic and avoids overfitting of the model.
- Diverse category of training images can make the image captioning model to work for multiple categories of images and hence can make the model more robust.

3.6 IMAGE DATA PREPARATION: The image should be converted to suitable features so that they can be trained into a deep learning model. Feature extraction is a mandatory step to train any image in deep learning model. The features are extracted using **Convolutional Neural Network (CNN) with Residual Network (Res-Net50)** model. It has been concluded recently that ResNet50 is the best architecture to classify the images into one among the 1000 classes given in the challenge. Hence, this model is ideal to use for this project as image captioning requires identification of images.

In ResNet50, there are 50 weight layers in the network and the deeper number of layers help in better feature extraction from images. The ResNet50 network uses 3*3 convolutional layers making its architecture simple and uses max pooling layer in between to reduce volume size of the image. The last layer of the image which predicts the classification is removed and the internal representation of image just before classification is returned as feature.

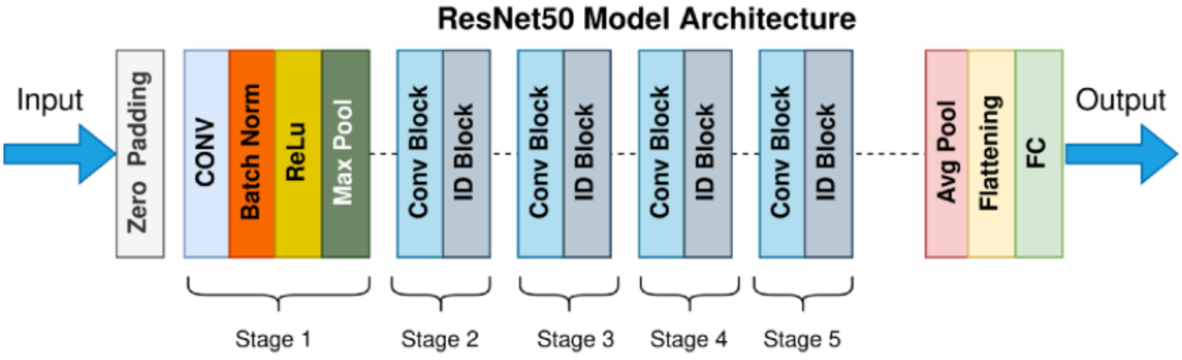


Fig 3.2. ResNet50 Model Architecture

3.7 CAPTION DATA PREPARATION:

Flickr8k dataset contains multiple descriptions described for a single image. In the data preparation phase, each image id is taken as key and its corresponding captions are stored as values in a dictionary.

3.7.1 DATA CLEANING: In order to make the text dataset work in machine learning or deep learning models, raw text should be converted to a usable format.

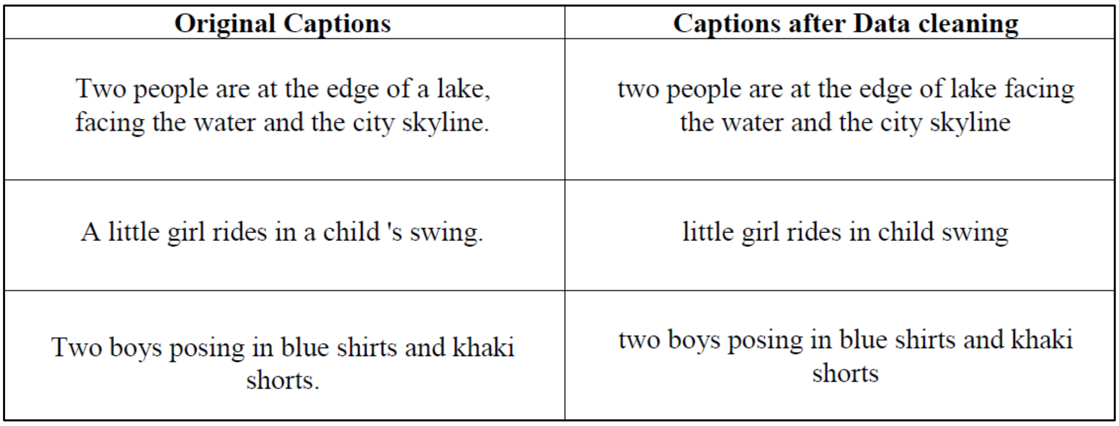
The following text cleaning steps are done before using it for the project:

- Removal of punctuations.
- Removal of numbers.
- Removal of single length words.
- Conversion of uppercase to lowercase characters.

Stop words are not removed from the text data as it will hinder the generation of a grammatically complete caption which is needed for this project.

This is done with the help of Flicker_8K_image_text_data **“descriptions.txt”**, pre-processing files (cleaning, encoding, lemmatization etc.) can be found in the “/text-files” directory of this Project Folder.

Table 3.1. Data Cleaning of Captions



Original Captions	Captions after Data cleaning
Two people are at the edge of a lake, facing the water and the city skyline.	two people are at the edge of lake facing the water and the city skyline
A little girl rides in a child 's swing.	little girl rides in child swing
Two boys posing in blue shirts and khaki shorts.	two boys posing in blue shirts and khaki shorts

3.8 IMPLEMENTATION:

3.8.1. Frameworks, Libraries & Languages

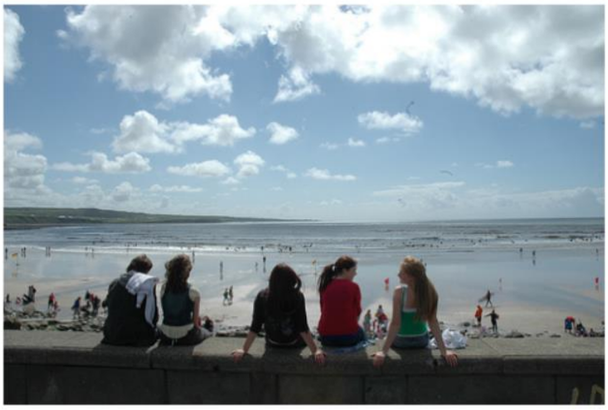
- Keras Functional API
- TensorFlow
- Python3
- NumPy
- Matplotlib
- Pickle

3.8.2 Loading the training set

In our “/text-files” folder, we have **descriptions.txt** file that contains a list of 6000 image names that we will use for training. First, we must load the prepared photo and text data so that we can use it to fit the model. We are going to train the data on all of the photos and captions in the training dataset. While training, we are going to monitor the performance of the model on the development dataset and use that performance to decide when to save models to file. The train development dataset has been predefined in the *descriptions.txt*, that contain lists of image file names. From these filenames, we can extract the photo identifiers and use these identifiers to filter photos and descriptions for each set.

3.8.2.1 Samples of Training Data

```
CAPTIONS -
a man uses ice picks and crampons to scale ice
an ice climber in a blue jacket and black pants is s
an ice climber scaling a frozen waterfall
a person in blue and red ice climbing with two picks
climber climbing an ice wall
```



```
CAPTIONS -
a couple of several people sitting on a ledge overlooking the beach
a group of people sit on a wall at the beach
a group of teens sit on a wall by a beach
crowd of people at the beach
several young people sitting on a rail above a crowded beach
```

Fig 3.3. Sample of Image Data being used for training the model

3.8.3 Tokenizing the vocabulary

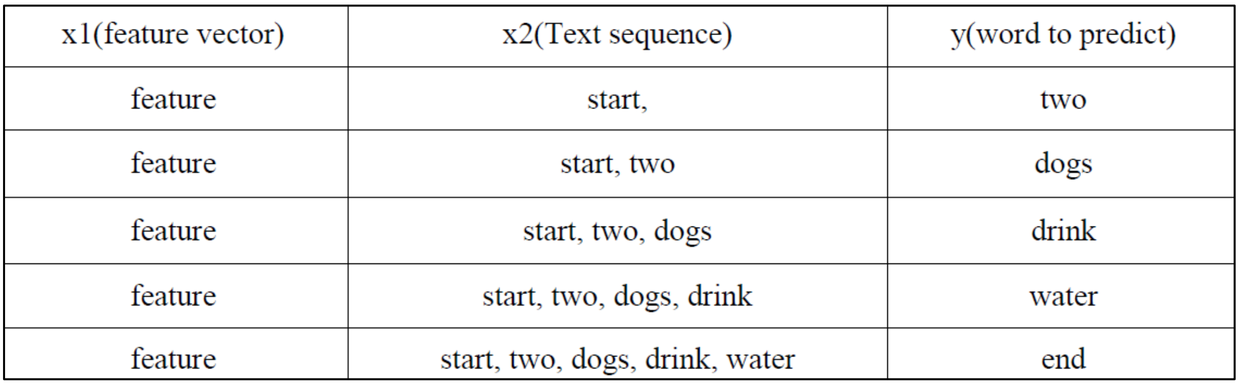
Computers don’t understand English words, for computers, we will have to represent them with numbers. So, we will map each word of the vocabulary with a unique index value. Keras library provides us with the tokenizer function that we will use to create tokens from our vocabulary and save them to a “word_to_idx.pkl”pickle file.

3.8.4 Create Data generator:

Let us first see how the input and output of our model will look like. To make this task into a supervised learning task, we have to provide input and output to the model for training. We have to train our model on 6000 images and each image will contain 2048 length feature vector and caption is also represented as numbers. This amount of data for 6000 images is not possible to hold into memory so we will be using a generator method that will yield batches. The generator will yield the input and output sequence.

generator = data_generator(train_content, train_encoding, word_to_index, max_len, batch_size)

Table 3.2. Word Prediction Generation (Step-by-Step)



x1(feature vector)	x2(Text sequence)	y(word to predict)
feature	start,	two
feature	start, two	dogs
feature	start, two, dogs	drink
feature	start, two, dogs, drink	water
feature	start, two, dogs, drink, water	end

3.8.5. Defining The Model:

This project uses the ResNet50 architecture for obtaining the image features. Res-Nets (short for Residual Networks) have been classic approach for many Computer Vision tasks, after this network won the 2015 ImageNet Challenge. Res-Nets showed how even very Deep Neural Networks (the original Res-Net was around 152 layers deep!) can be trained without worrying about the vanishing gradient problem. The strength of a Res-Net lies in the use of Skip Connections - these mitigate the vanishing gradient problem by providing a shorter alternate path for the gradient to flow through.

ResNet50 which is used in this project is a smaller version of the original ResNet152. This architecture is so frequently used for Transfer Learning that it comes preloaded in the Keras framework, along with the weights (trained on the ImageNet dataset). Since we only need this network for getting the image feature vectors, so we remove the last layer (which in the original model was used to classify input image into one of the 1000 classes). The encoded features for training and test images are stored at "encoded_train_features.pkl" and "encoded_test_features.pkl" respectively.

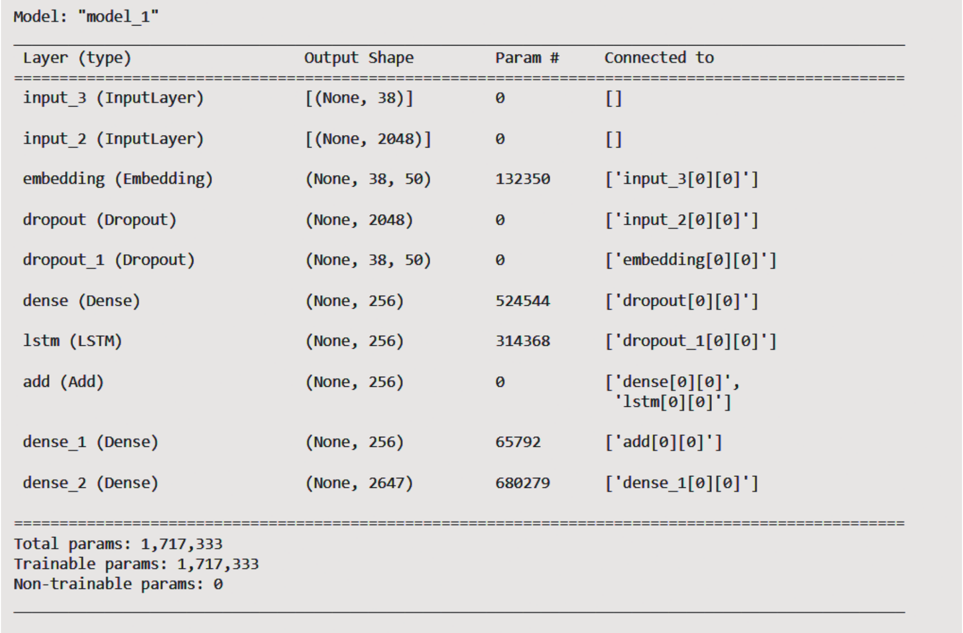
Feature Extractor: The feature extracted from the image has a size of 2048, with a dense layer, we will reduce the dimensions to 256 nodes.

Sequence Processor: An embedding layer will handle the textual input, followed by the LSTM layer.

Decoder: By merging the output from the above two layers, we will process by the dense layer to make the final prediction. The final layer will contain the number of nodes equal to our vocabulary size. Image Feature Extractor model expects input photo features to be a vector of 4,096 elements. These are processed by a Dense layer to produce a 256-element representation of the photo. The Sequence Processor model expects input sequences with a pre-defined length (34 words) which are fed into an Embedding layer that uses a mask to ignore padded values. This is followed by an LSTM layer with 256 memory unit. Both the input models produce a 256-element vector. Further, both input models use regularization in the form of 50% dropout. This is to reduce overfitting the training dataset, as this model configuration learns very fast.

The Decoder model merges the vectors from both input models using an addition operation. This is then fed to a Dense 256 neuron layer and then to a final output Dense layer that makes a SoftMax prediction over the entire output vocabulary for the next word in the sequence.

Table 3.3. Model Summary



```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 38)]         0           []

 input_2 (InputLayer)           [(None, 2048)]       0           []

 embedding (Embedding)          (None, 38, 50)       132350      ['input_3[0][0]']

 dropout (Dropout)              (None, 2048)         0           ['input_2[0][0]']

 dropout_1 (Dropout)            (None, 38, 50)       0           ['embedding[0][0]']

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

 lstm (LSTM)                    (None, 256)          314368      ['dropout_1[0][0]']

 add (Add)                      (None, 256)          0           ['dense[0][0]',
                                                                  'lstm[0][0]']

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

 dense_2 (Dense)                (None, 2647)         680279      ['dense_1[0][0]']

==================================================================================================
Total params: 1,717,333
Trainable params: 1,717,333
Non-trainable params: 0
__________________________________________________________________________________________________
```

GloVe vectors were used for creating the word embeddings for the captions. The version used in this project contains 50-dimensional embedding vectors for 6 billion English words. It can be downloaded from glove.6B.50d. These Embeddings are not processed (fine-tuned using the current data) further during training time.

The neural network for generating the captions has been built using the Keras Functional API. The features vectors (obtained from the ResNet50 network) are processed and combined with the caption data (which after converting into Embeddings, have been passed through an LSTM layer). This combined information is passed through a Dense layer followed by a SoftMax layer (over the vocabulary words). The model was trained for 20 epochs, and at the end of each epoch, the model was saved in the "/Models" directory. Each epoch in his process took about half an hour (approximately 10 hours of Computation).

3.8.6. Training the model:

To train the model, we will be using the 6000 training images by generating the input and output sequences in batches and fitting them to the model using “model.fit(generator, steps_per_epoch=steps)” method.

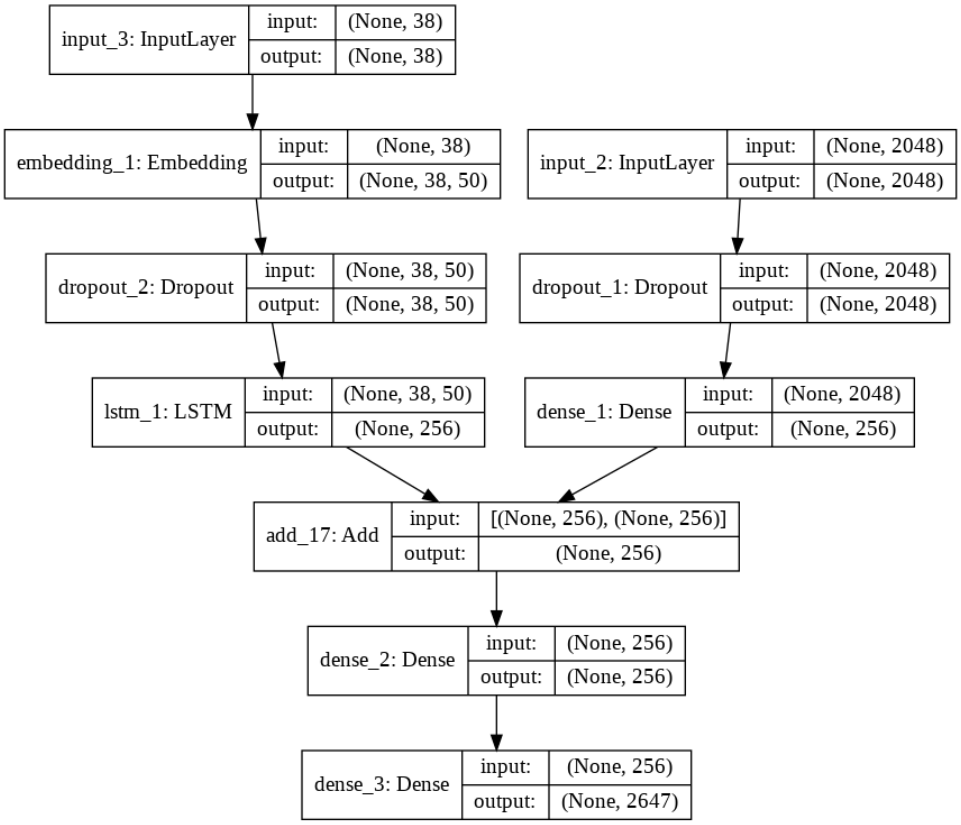
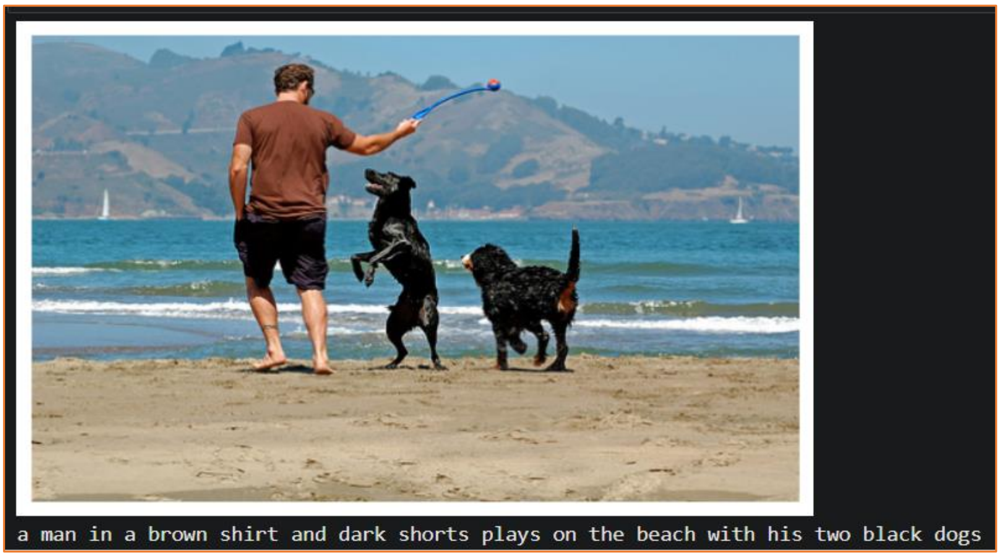

input_3: InputLayer | input: (None, 38) | output: (None, 38)
embedding_1: Embedding | input: (None, 38) | output: (None, 38, 50)
input_2: InputLayer | input: (None, 2048) | output: (None, 2048)
dropout_2: Dropout | input: (None, 38, 50) | output: (None, 38, 50)
dropout_1: Dropout | input: (None, 2048) | output: (None, 2048)
lstm_1: LSTM | input: (None, 38, 50) | output: (None, 256)
dense_1: Dense | input: (None, 2048) | output: (None, 256)
add_17: Add | input: [(None, 256), (None, 256)] | output: (None, 256)
dense_2: Dense | input: (None, 256) | output: (None, 256)
dense_3: Dense | input: (None, 256) | output: (None, 2647)

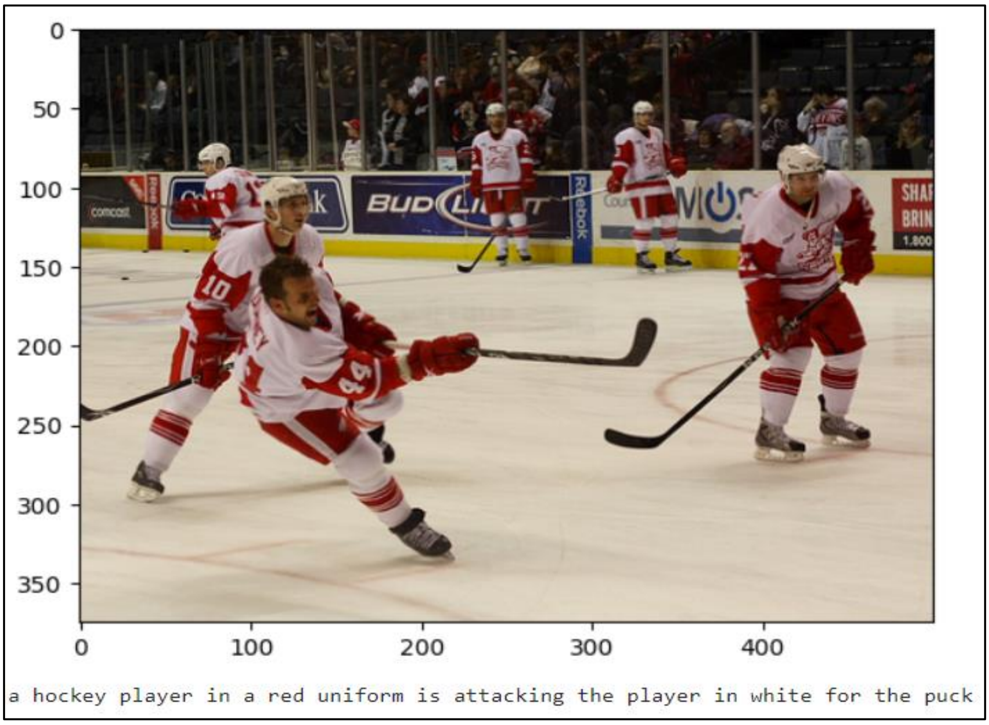
Fig 3.4. A structural plot of the Model Architecture

3.8.7 Testing the model:

The model has been trained, now, we will make a separate file "_Generate_Caption_for_Image.ipynb" which will load the model and generate predictions. The predictions contain the max length of index values so we will a "idx_to_word.pkl"pickle file to get the words from their index values.



```
a man in a brown shirt and dark shorts plays on the beach with his two black dogs
```



```
a hockey player in a red uniform is attacking the player in white for the puck
```

Fig 3.5. Output Caption of a given Image

CHAPTER - 4: RESULTS and Limitations

4.1 Results:

The proposed system is developed using python, TensorFlow, Keras framework and the performance of the designed system is evaluated based on the Bilingual Evaluation Understudy (FLICKR8K) dataset. The proposed approach shows an accuracy of 88.75% and it is tested upon the dataset which is a combination of a benchmark dataset and a real time dataset.

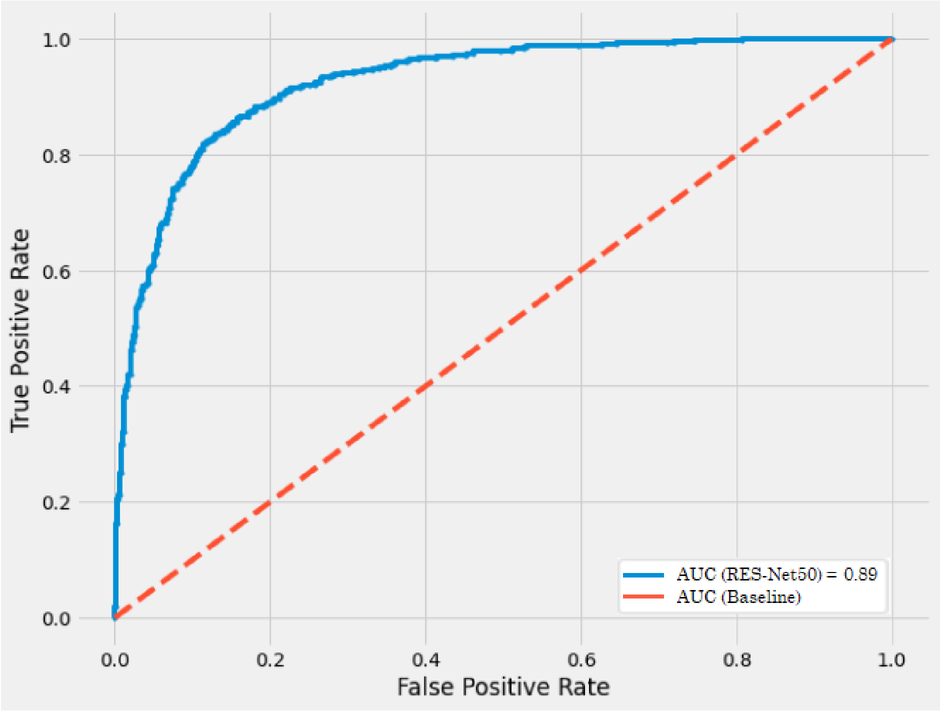


Fig 4.1. ROC curve to demonstrate the accuracy of the Model

4.2 Limitations:

The neural image caption generator gives a useful framework for learning to map from images to human-level image captions. By training on large numbers of image caption pairs, the model learns to capture relevant semantic information from visual features.

However, with a static image, embedding our caption generator will focus on features of our images useful for image classification and not necessarily features useful for caption generation.

To improve the amount of task-relevant information contained in each feature, we can train the image embedding model (the ResNet50 network used to encode features) as a piece of the caption generation model, allowing us to fine-tune the image encoder to better fit the role of generating captions. Also, if we actually look closely at the captions generated, we notice that they are rather mundane and commonplace.

4.3 Extensions:

This section lists some ideas for extending the model that be explored for better accuracy.

Alternate Pre-Trained Image Models: A 50-layer Res-Net model was used for feature extraction. Consider exploring larger models that offer better performance on the ImageNet dataset, such as Inception.

Reduce Vocabulary Size: A larger vocabulary of nearly eight thousand words was used in the development of the model. Many of the words supported may be misspellings or only used once in the entire dataset. Refine the vocabulary and reduce the size, perhaps by half.

Tune Model: The configuration of the model was not tuned on the problem. Explore alternate configurations and see if you can achieve better performance.

CHAPTER - 5: CONCLUSION AND FUTURE WORK

5.1 CONCLUSION:

In this project, we have developed deep learning-based image captioning method. We have given a taxonomy of image captioning techniques, shown generic block diagram of the major groups and highlighted their pros and cons. We discussed different evaluation metrics and datasets with their strengths and weaknesses. A brief summary of experimental results is also given.

We briefly outlined potential research directions in this area. Although deep learning-based image captioning methods have achieved a remarkable progress in recent years, a robust image captioning method that is able to generate high quality captions for nearly all images is yet to be achieved. With the advent of novel deep learning network architectures, automatic image captioning will remain an active research area for some time.

We have used Flickr_8k dataset which includes nearly 8000 images, and the corresponding captions are also stored in a text file descriptions.txt. Although deep learning -based image captioning methods have achieved a remarkable progress in recent years, a robust image captioning method that is able to generate high quality captions for nearly all images is yet to be achieved. With the advent of novel deep learning network architectures, automatic image captioning will remain an active research area for some time. The scope of image-captioning is very vast in the future as the users are increasing day by day on social media and most of them would post photos. So, this project will help them to a greater extent.

5.2 FUTURE SCOPE:

Future work Image captioning has become an important problem in recent days due to the exponential growth of images in social media and the internet. This report discusses the various research in image retrieval used in the past and it also highlights the various techniques and methodology used in the research. As feature extraction and similarity calculation in images are challenging in this domain, there is a tremendous scope of possible research in the future.

Current image retrieval systems use similarity calculation by making use of features such as colour, tags, IMAGE RETRIEVAL USING IMAGE CAPTIONING histogram, etc. There can't be completely accurate results as these methodologies do not depend on the context of the image.

Hence, complete research in image retrieval making use of context of the images such as image captioning will facilitate to solve this problem in the future. This project can be further enhanced in future to improve the identification of classes which has a lower precision by training it with more image captioning datasets.

This methodology can also be combined with previous image retrieval methods such as histogram, shapes, etc. and can be checked if the image retrieval results get better.

REFERENCES

A. Books and Research Papers:

[1] Ahmet Aker and Robert Gaizauskas. 2010.

Generating image descriptions using dependency relational patterns.

In Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, 1250–1258.

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2017.

Bottom-up and top-down attention for image captioning and vqa. arXiv preprint arXiv :1707.07998 (2017).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015.

Neural machine translation by jointly learning to align and translate.

In International Conference on Learning Representations (ICLR).

[4] Raffaella Bernardi, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, Barbara Plank, et al. 2016.

Automatic Description Generation from Images: A Survey of Models, Datasets, and Evaluation Measures.

Journal of Artificial Intelligence Research (JAIR) 55, 409–442.

[5] Jyoti Aneja, Aditya Deshpande, and Alexander G Schwing. 2018.

Convolutional image captioning.

IEEE Conference on Computer Vision and Pattern Recognition. 5561–5570.

B. Websites:

1. Implementation Guidance:

   https://www.analyticsvidhya.com/blog/2021/12/step-by-step-guide-to-build-image-caption-generator-using-deep-learning/

2. CNN:
   https://www.analyticsvidhya.com/blog/2021/07/convolution-neural-network-better-understanding/

3. LSTM:
   https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/

4. Residual Networks (Computer Vision):
   https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33

5. https://keras.io/guides/serialization_and_saving/

6. https://keras.io/guides/functional_api/

7. https://arxiv.org/abs/1512.03385

8. https://jair.org/index.php/jair/article/view/10833

9. https://arxiv.org/abs/1411.4555v2