AUTOMATIC IMAGE CAPTIONING USING DEEP LEARNING

Submitted in partial fulfillment of the
requirements for the degree of
Bachelor of Engineering
Synopsis Report- Stage-II
by

Solanki Dhruv
Roll No.49
Supe Tejal
Roll No.50
Undirkar Siddhi
Roll No.54

Under the Supervision of

Prof. J. P. Patil

DEPARTMENT OF INFORMATION TECHNOLOGY
KONKAN GYANPEETH COLLEGE OF
ENGINEERING KARJAT-410201
JUNE 2021

Certificate

This is to certify that the project entitled Automatic Image Captioning Using Deep Learning is a bonafide work of SOLANKI DHRUV (Roll No.49), SUPE TEJAL (Roll No.50), UNDIRKAR SIDDHI (Roll No.54) submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of Undergraduate in DEPARTMENT OF INFORMATION TECHNOLOGY.

Supervisor/Guide

Prof. J. P. Patil

Department of Information Technology

Head of Department

Department of Information Technology

Principal

Dr. M. J. Lengare

Konkan Gyanpeeth College of Engineering

Project Report Approval for B.E.

This thesis / dissertation/project report entitled Automatic Image Captioning Using Deep Learning by SOLANKI DHRUV (Roll No.49), SUPE TEJAL (Roll No.50), UNDIRKAR SIDDHI (Roll No.54) is approved for the degree of DEPARTMENT OF INFORMATION TECHNOLOGY.

Examiners

1.....

2.....

Date.

Place.

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature
(Solanki Dhruv) Roll No.49

Signature

(Supe Tejal) Roll No.50

Signature

(Undirkar Siddhi) Roll No.54

Date.

Abstract

Image captioning means automatically generating a caption for an image. Automatically creating the description of an image using any natural language sentences is a very challenging task. It requires expertise of both image processing as well as natural language processing. Sharing photos through the internet (e.g., Instagram, Facebook, etc.), which becomes a common practice, leads to archives in the order of millions of images. Manual annotation is time consuming and extremely labor-intensive work. Thereafter, computer-assisted system are desired to lesson these difficulties by automation using machine learning. The annotated image can be used for retrieval purposes, in image processing algorithms, intelligent scanner, digital cameras, photocopiers, and printers. Our goal is to create system which will annotate images based on previously learned datasets.

Acknowledgements

Success is nourished under the combination of perfect guidance, care blessing. Acknowledgment is the best way to convey. We express deep sense of gratitude brightness to the outstanding permutations associated with success. Last few years spend in this estimated institution has molded us into condent and aspiring Engineers. We express our sense of gratitude towards our project guide Prof. J. P. Patil. It is because of his valuable guidance, analytical approach and encouragement that we could learn and work on the project. We will always cherish the great experience to work under their enthusiastic guidance. We are also grateful to our principle Dr. M.J. Lengare who not only supporting us in our project but has also encouraging for every creative activity. We extend our special thanks to all teaching and non-teaching staff, friends and well-wishers who directly or indirectly contributing for the success of our maiden mission. Finally, how can we forget our parents whose loving support and faith in us remains our prime source of inspiration. Lastly, we would like to thank all those who directly and indirectly helping to complete this project. We would also like to acknowledge with much appreciation the crucial role of the staff of Information Technology Department, who gave the permission to use the all required software/hardware and the necessary material to completing to the project.

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Introduction

Automatically generating captions to an image shows the understanding of image by computers. For a caption model, it not only need to find which objects are contained in the image but also need to be able to expressing their relationships in a natural language such as English. Given an image, find the most probable sequence of words (sentence) describing the image. CNN paired along with RNN provided a good method to do the task. Our model contains CNN and RNN based LSTM algorithm for generating captions. CNN used for decoding the image and LSTM to encode the image. The explosive growth of image data leads to the research and development of Content Based Image Retrieval (CBIR) systems. CBIR systems extract and retrieve images by their low-level features, such as color, texture, and shape. However, these visual contents do not allow users to query images by semantic meanings. Image annotation systems, a solution to solve the inadequacy of CBIR systems, aim at automatically annotating image with some controlled keywords.

Image annotation refers to the tagging of images with appropriate keywords. The process of annotating images manually is time-consuming and extremely labor-intensive work. Thereafter, computer-assisted systems are desired to lessen these difficulties by automation. The automatic image annotation is targeted to assign a few relevant text keywords to an input image that reflects its visual content. Machine learning techniques are used to develop the image annotation systems to map the low-level (visual) features to high-level concepts or semantics.

1.2 Objectives

- The objective of our project is to extract the features of a given image and to automatically caption the objects present in the image.
- To build a system that will identify multiple objects from a given image.
- To create a system that will be able to caption the objects present in the image.

- To be able to train the model created to identify and caption the images.
- To be able to test and evaluate the performance of the system of both the training and testing phase.

1.1 Purpose, Scope, and Applicability

Purpose, Scope and Applicability: The description of Purpose, Scope, and Applicability are given below:

1.1.1 Purpose

The main purpose of image annotations is to highlight or capture the targeted object in a picture to make it recognizable for machines. As large collection of data is required for technologies based on machine learning, it is crucial that the data available is highly precise in terms of correctness and its relevance with real world entities. This kind of data can be obtained by manually tagging images, this task is very tedious as well as time consuming and labor intensive. It is not practical. So, a method is required to carry the same automatically.

1.1.2 Scope

- The scope of this project is to classify the images from the different classes available which makes it easy to search an image based on a keyword.
- The system developed in this project is such that it will add a caption to multiple objects present in an image.
- The images after been identified and captioned will be saved into the separate database.

1.1.3 Applicability

Image annotation system have immense areas of applicability, we list some of them that are practical interest to us:

- Content based image retrieval.
- Improve data-sets for training.
- Improve image-based search engines.
- Create problem specific image data-sets.

1.4 Organization of Report

The paper is organized as follows: Chapter 1 includes introduction, scope, objectives and applicability of our project. Chapter 2 focuses on the previous work done in the field of study and gives a brief comparison of the papers referred. Chapter 3 defines the problem statement and system requirements of our project. Chapter 4 gives general idea of system's work flow and design. Chapter 5 gives general idea of implementation and testing done. Chapter 6 concludes with future scope of the project.

Chapter 2

LITERATURE SURVEY

1 IMAGE ANNOTATION USING SVM

Author: Claudio Cusano, Gianluigi Ciocca, Raimondo Schettini

In this paper, author suggested an image annotation tool using SUPPORT VECTOR MACHINE for classifying image regions in one of seven classes - sky, skin, vegetation, snow, water, ground, and buildings - or as unknown. The central idea of SVM is to adjust a discriminating function so that it makes optimal use of the separability information of boundary cases. Using SVM for classification of image the results are satisfactory. The accuracy in the classification has increased for each class (more than 50% for the ground class). Some errors are still present, ground tiles in particular have often been misclassified as vegetation tiles (5,4% of cases) and vice-versa (5%), snow tiles have been misclassified as sky tiles (5,4%), and buildings tiles as ground tiles (4,5%).

2 AUTOMATICALLY ANNOTATING IMAGES WITH KEYWORDS: A REVIEW OF IMAGE ANNOTATION SYSTEM.

Author: Chih-Fong Tsai, Chihli Hung. (2007).

This paper briefly describes commonly used image segmentation methods including global and local feature extraction. This paper is organized as follows. Section 2 briefly describes commonly used image segmentation methods including global and local feature extraction. Section 3 covers the low-level features which are generally extracted from images. Section 4 overviews the mostly used supervised learning methods for the task of image annotation. Some representative image annotation systems are also described. Section 5 presents advanced techniques by combining multiple classifiers and hybrid classifiers for image annotation. Section 6 provides a comparison of related image annotation systems in terms of their feature representation, classifier used, classification scale.

3 DEEP CAPTIONING WITH MULTIMODAL RECURRENT NEURAL NETWORKS (M-RNN).

Author: JunhuaMao. (2015).

This paper presents a m-RNN model for generating novel image captions, over benchmark datasets ((IAPR TC-12, Flickr 8K, Flickr 30K and MS COCO). In addition, the m-RNN model is applied to retrieval tasks for retrieving images or sentences. Using this model sentence level description can be obtained. In this work, author propose a multimodal Re- current Neural Networks (m-RNN) model to address both the task of generating novel sentences descriptions for images, and the task of image and sentence retrieval. The whole m-RNN model contains a language model part, a vision part and a multimodal part.

The language model part learns a dense feature embedding for each word in the dictionary and stores the semantic temporal context in recurrent layers. The vision part contains a deep Convolutional Neural Network (CNN) which generates the image representation. Results are shown for three tasks: 1. generating novel sentences,2. retrieving images given a sentence and 3. retrieving sentences given an image. Results on IAPR TC-12 are as follows: The result shows that 20.9are ground truth. Result on FLICKR30K and MS COCO are as follows: 71generated sentences for MSCOCO datasets are novel (i.e., different from training sentences).

4 AUTOMATIC IMAGE ANNOTATION USING NEURAL NETWORKS

Author: Aanchan K Mohan, Marwan A. Tokri.

This paper talks about the training process to achieve automatic image annotation using neural networks. This report is organized as follows: Section 2 describes the training process in great detail and the steps followed therein, Section 3 summarizes the results obtained following our proposed approach, and Section 4 concludes this report with a discussion about future work that this project could be extended to. The Training Process,

2.1. The Training Database: The IAPR-TC 12 Benchmark image database was used for this project. This database consists of 20000 naturals still in ages (plus 20000 thumbnails). Each image is associated with a free-flowing text caption describing the image. For this project, about 17558 images were provided as a part of the training set along with image keywords(nouns) extracted from the free-flowing text, and the free-flowing text itself. 2.2 The Proposed Approach: the procedure that is followed consists of doing some preprocessing on the image database to extract

a dictionary of key words. 2.3 The Preprocessing: created a dictionary of keywords based on their frequency of occurrence in the image database. We got rid of words which appeared to be outliers, and kept only those keywords whose frequency of occurrence was greater than 10. 2.4 Feature Extraction: The process of feature extraction involves the extraction of a 6-element vector for each block of the image after having divided the entire image into regions of blocks of size 4X4.The feature extraction for the color information is done in the L, u, v space. This involves in converting the image from the default RGB color space to the L u, v color space. 2.5 Region Segmentation: The second step in training approach is to cluster the image into a set of 16 clusters that describe the image. We used the K-means clustering algorithm to perform the task of clustering to limit our output clusters to 16 clusters as author mentioned. 2.6 Artificial Neural Networks: The last step in training approach is to build and train a neural network to learn relationship between image segments and the annotation keywords. Results shows that using this method even in low end computers the process of training takes few minutes. The accuracy rate and the recall rate are both greater than 19%, which is quite high.

5 OBJECTIVE-GUIDED IMAGE ANNOTATION

Author: Ivor Wai-Hung Tsang, Qi Mao. (2013).

This paper talks about various methodologies to assign multiple tags for images. The rest of this paper is organized as follows. Related work is brief discussed in Section II. The objective-guided performance measures and its property analysis are given in Section III and Section IV. Section V illustrates the united multi-label learning framework. Experimental results are shown in Section VI. To address the issue that many image annotation methods neglect optimizing the objective-guided performance measures, in this paper, we attempt to optimize a variety of objective specic measures in a unied multi-label learning framework. We present a multilayer hierarchical structure of learning hypothesis for multi-label problems based on which a variety of loss functions with respect to objective guided measures are dened. And then, the unied learning framework is presented. Our analysis reveals that macro-averaging measures are very sensitive to infrequent keywords, micro averaging measures are time-consuming, and hamming measure is easily affected by skewed distributions. The experimental results on four image annotation datasets demonstrate that optimizing the objective-guided performance measure is able to improve this performance measure, especially for F1 score, which consistently shows very competitive results over three measures on all four datasets.

6 IMAGE CAPTIONING WITH SEMANTIC ATTENTION

Author: Quanzeng You, Hailin Jin

In this paper, author propose a new image captioning approach that combines the top down and bottom-up approaches through a semantic attention model. Author's definition of semantic attention in image captioning is the ability to provide a detailed, coherent de- scription of semantically important objects that are needed exactly when they are needed. The semantic attention model has the following properties: 1) able to attend to a semantically important concept or region of interest in an image, 2) able to weight the relative strength of attention paid on multiple concepts, and 3) able to switch attention among concepts dynamically according to task status. The model is built on top of a Recurrent Neural Network (RNN), whose initial state captures global information from the top-down feature. As the RNN state transits, it gets feedback and interaction from the bottom-up attributes via an attention mechanism enforced on both network state and output nodesPerformance on MS-COCO The two model variants are trained on MS COCO dataset using the ground-truth visual attributes, and compared in Table 4. The performance of using output attention is slightly better than only using input attention on some metrics. However, the combination of this two attentions improves the performance by several percent's on almost every metric.

This can be attributed to that fact that at tention mechanisms at input and output layers are not the same, and each of them attend to different aspects of visual attributes. Therefore, combining them may help provide a richer interpretation of the context and thus lead to improved performance. The above results are tested on other datasets such as Flickr30, the results are the same, also the accuracy are high.

7 DEEP VISUAL-SEMANTIC ALIGNMENTS FOR GENERATING IMAGE DESCRIPTIONS

Author: Andrej Karpathy, Li Fei-Fei. (2015)

The paper introduces a model that generates natural language descriptions of image regions based on weak labels in form of a dataset of images and sentences, and with very few hardcoded assumptions. The approach features a novel ranking model that aligned parts of visual and language modalities through a common, multimodal embedding. This model provides state of the art

performance on image sentence ranking experiments. Second, we described a Multimodal Recurrent Neural Network architecture that generates descriptions of visual data. The model is tested and trained on MSCOCO, Flickr8K datasets. Author evaluated its performance on both full frame and region-level experiments and showed that in both cases the Multimodal RNN outperforms retrieval baselines. There are some limitations to this model. First, the model can only generate a description of one input array of pixels at a xed resolution. Additionally, the RNN receives the image information only through additive bias interactions, which are known to be less expressive than more complicated multiplicative interaction.

8 DEEP CONVOLUTIONAL RANKING FOR MULTILABEL IMAGE ANNOTATION

Author: YunchaoGong, ThomasK.Leung. (2014).

In this paper for multilabel image annotation, architecture approached as basic frame- work is of convolutional ranking and mainly focused on training the network with loss functions tailored for multi-label prediction tasks. The paper mainly focuses on Multilabel Ranking Losses. The first loss is SoftMax, it has been used for multilabel annotation in Tagprop, and is also used in single label image classification. The second loss function is Pairwise Ranking, it directly models the annotation problem. One limitation of this loss is that it optimizes the area under the ROC curve (AUC) but does not directly optimize the top-k annotation accuracy. Because for image annotation problems we were mostly interested in top-k annotations, this pairwise ranking loss did not best-t the purpose. The third loss function is Weighted Approximate Ranking (WARP), It specifically optimizes the top-k accuracy for annotation by using a stochastic sampling approach. Author performed experiments on the largest publicly available multilabel dataset, NUS-WIDE. This dataset contains 269,648 images downloaded from Flickr that have been manually annotated, with several tags (2-5 on average) per image. After ignoring the small subset of the images that are not annotated by any tag, we had a total of 209,347 images for training and testing. We used a subset of 150,000 images for training and used the rest of the images for testing. The tag dictionary for the images contains 81 different tags. Using Visual features and other machine learning algorithm tests on dataset gives the following results, the CNN+SoftMax method out performs the Visual Feature+SVM baseline by about 10%.

9 AUTOMATIC IMAGE ANNOTATION USING MULTI-OBJECT IDENTIFICATION

Author: Yin-Fu Huang, Hsin-Yun Lu. (2010).

In this paper author suggest in order to improve annotation accuracy, the irrelevant areas such as background can be eliminated from an image before extracting the main features from the image, and then classifiers are built using these extracted features. The system framework consists of two phases: training phase and annotation phase. The training phase could be divided into three stages: 1) main object training, 2) background object training, and 3) object association analysis. First, a main object detection method is proposed to segment the main object from a training image.
Then, the feature vectors of the main object are extracted to train its classifier. The classifier could determine what class an image belongs to. Second, we also segment training images and extract the features vectors from the background objects to train background classifiers. Finally,the main task is to perform the association analysis by using a probability model called GMM. The object association analysis is to find the associations between main objects (or image classes) and image background objects, and build the association knowledge base. The purpose is to eliminate the irrelevant model testing in the annotation phase so that annotation accuracy could be improved. In the annotation phase, we segment a test image and extract the feature vectors from a main object. The main object classifier would identify the class of the test image. Then, the system can retrieve the relevant backgrounds of the detected image class from the association knowledge base, and the relevant background classifiers would be used to detect whether these backgrounds appear in the test image. Main and background object training is done next. The following step is object association analysis. The purpose of object association analysis is to speed up the process time in the annotation phase and improve annotation accuracy. Gaussian mixture model: The GMM is an efficient method to precisely describe the sample clustering in the feature space. Through the training, proper parameters could be obtained to fit with target statistics. The final step is Annotation phase. First, we segment and detect the main object out of a test image. Then, the features of the main object are extracted to identify its class, using the built main object models. According to the identified class, we can get all related background models from the association knowledge base. Next, the background objects are discriminated one by one from the remainder of the image. Then, the features of the background objects are extracted to detect the backgrounds, using these related background models. Finally, the annotation of an image, including the class and backgrounds, can be achieved. This results in a very good detection of subject and specific annotation of individual objects. This increases accuracy.

Experimental results show that the final annotation of most classes achieves more than 85they also validate that the system would not annotate incorrect backgrounds in an image even if its image class implies these backgrounds in the association knowledge base.

10 AUTOAMTIC MULTI-LABLE IMAGE ANNOTATION FOR SMART CITIES

Author: Gyayak Sanghi, Nalin Kanungo. (2017).

In this author used ML-KNN algorithm to classify specific objects from dataset. In this case objects that are seen in a city are expected as output. Several images of a landmark are extracted from flicker dataset. Every new image is compared with all the images and its nearest neighbors are determined, then the image is compared to the visually closest image of the landmark. Finally, the image is annotated with a particular label only if it matches above a particular threshold. A kernel method formultilabeled classification.which is basically a slight modification of the KNN approach discussed in a research work. Here the images are categorized according to the keywords and for each keyword the k-nearest neighbors to the input image are found. Finally, a weighted sum is taken over the samples to assign a rank to each label so obtained. A research work proposed use of maximum entropy for automatic image annotation which divides each training image into visterms or rectangular regions having a label associated to it. Then for any test image, the closest neighbors are found foreach visterm and probabilistic approach is used to as- sign the label to the test image. Methodology used is ML-KNN algorithm on own dataset which is composed of 100 training images and 50 testing images all of urban cities. Images are represented by feature vectors with 78 features each. These 78 features are composed of colour, edge, texture features. The values of result parameters obtained above are comparable to the value of same parameters obtained when this algorithm is applied on benchmark dataset (corel5kdataset). Using Benchmark dataset there is an improvement in the value of hamming loss and coverage parameters.

11 AUTOMATED ANNOTATION OF NATURAL IMAGES USING AN EXTENDED ANNOTATION MODEL

Author: GABRIEL MIHAI, LIANA STANESCU. (2012).

The annotation process implemented in this system is based on CMRM (Cross Media Relevance Model). The annotation model is based on object-oriented approach. The paper describes the extension of an image annotation model that can be used for annotating natural images. The CMRM annotation model has proved to be very efficient by several studies.

This model learns the joint probability of concepts and blobs based on a well know benchmark: SAIAPR TC-12. This benchmark contains a large-size image collection comprising diverse and realistic images, includes an annotation vocabulary having a hierarchical organization, well defined criteria for the objective segmentation and annotation of images. Because the quality of an image region and the running time of the segmentation process are two important factors for the annotation process that have been used, uses a segmentation algorithm based on a hexagonal structure which was proved to satisfy both requirements: a better quality and a smaller running time. Each new image was annotated with concepts taken from an ontology created starting from the information provided by the benchmark: the hierarchical organization of the vocabulary and the spatial relationships between regions. For storing the information required by the annotation process it used an object-oriented database called db4o. The object-oriented approach has simplified the way of describing the modified version. The experimental results realized from two perspectives (annotation and retrieval) have proved that the proposed modified model produces better results than the initial model. Experimental results (1) Annotation perspective. In order to evaluate the annotation system author has used a testing set of 400 images that were manually annotated and not included in the training set used for the CMRM model. This set was segmented using the segmentation algorithm described above and a list of concepts having the joint probability greater than a threshold value was assigned to each image. Then the number of relevant concepts automatically assigned by the annotation system was compared against the number of concepts manually assigned by computing an accuracy value for both modules. The average accuracy value obtained for the initial model was 0.46 The annotation process implemented in this system is based on CMRM (Cross Media Relevance Model). The annotation model is based on object oriented approach. The paper describes the extension of an image annotation model that can be used for annotating natural images. The CMRM annotation model has proved to be very efficient by several studies. This model learns the joint probability of concepts and blobs based on a well know benchmark: SAIAPR TC-12. This benchmark contains a large-size image collection comprising diverse and realistic images, includes an annotation vocabulary having a hierarchical organization, well defined criteria for the objective segmentation and annotation of images. Because the quality of an image region and the running time of the segmentation process are two important factors for the annotation process that have been used, uses a segmentation algorithm based on a hexagonal structure which was proved to satisfy both requirements: a better quality and a smaller running time. Each new image was annotated with concepts taken from an ontology created starting from the information provided by the benchmark: the hierarchical organization of the vocabulary and the spatial relationships between regions. For storing the information required by the annotation process it used an object-oriented database called db4o.

The object-oriented approach has simplified the way of describing the modified version. The experimental results realized from two perspectives (annotation and retrieval) have proved that the proposed modified model produces better results than the initial model. Experimental results (1) Annotation perspective. In order to evaluate the annotation system author has used a testing set of 400 images that were manually annotated and not included in the training set used for the CMRM model. This set was segmented using the segmentation algorithm described above and a list of concepts having the joint probability greater than a threshold value was assigned to each image. Then the number of relevant concepts automatically assigned by the annotation system was compared against the number of concepts manually assigned by computing an accuracy value for both modules. The average accuracy value obtained for the initial model was 0.46 and the average accuracy value obtained for the modified model was 0.54. (2) Retrieval perspective: After computing the precision and recall values for all concepts it was computed a mean precision equal to 0.38 (0.34 obtained using the standard version) and a mean recall equal to 0.44 (0.36 obtained using the standard ver sion). It can be observed that the values corresponding to the proposed modified model are always greater than the values of the initial model.

12 A COMPARATIVE IMAGE AUTO-ANNOTATION

Author: Mahdia Bakalem, Nadjia Benblidia. (2013).

The image annotation process presented in this paper is: The first annotation process based on texture parameters allows extracting texture visual features. The second annotation process based on color parameters extracts visual features of color. The latest process based on fusion of texture and color parameters permits to extract visual features of texture and color at the same time. The image auto-annotation process consists of two main steps: a training step and a new image processing step. Training Step: This step consists of regrouping the similar visual regions in classes called Blobs and annotating them. 1) Visual Space Preparation. The aim of this part is to construct the blobs that represent visual space. 2) Visual-Textual Correlation. We correlate between two aspects visual and semantic in order to annotate the blobs constructed in the first one. For each blob, author used a blob annotation algorithm which permits to inherit the image keys words of regions belong to the blob by selective heritage. B. New image processing step: This step permits to annotate automatically any new image; the first task is the segmentation of a new image into regions, followed by the extraction of the visual features of each region and by the affectation of regions to the blobs defined in the previous step (training step).

The new image will be annotated by the selective heritage of the key words of the blobs to which its regions belong. In order to improve the annotation of image, we conduct our experiments on the annotated image data set referred to as Corel, consists of 16.000 images annotate manually with 1 to 5 keywords. Author used 1801 images in the training process, which have been considered in order to construct the annotated blobs. For each region, a visual features vector is defined, depending on the system. First vector is defined by texture parameters. The second is defined by color parameters (RGB space: the average, the variance and the moments order 3 of each component of RGB) and the last vector is defined by fusing the parameters of texture and color. The results show that the annotation by latent space is more promising than the annotation by textual space. Author suggests improvements in the system by using special images and also by refining the training process.

Chapter 3

REQUIREMENTS AND ANALYSIS

3.1 Problem Definition

In this section we define the problem on which we are working in the project. Details are provided of the overall problem and then divided the problem in to sub-problems.

a) Problem Definition:

To build a system that will generate an automatic caption of the image by CNN, LSTM and trained VGG16 model. The image will be of the dataset provided to the system. The caption will get generate by identifying various objects in the image.

b) Sub-problem:

- To assign tags to images based on their semantic features. To build a system that can perform annotation automatically.
- To extract features using the favorable method.
- To classify multiple objects from a single image.
- To test and evaluate the performance of the system.

3.2 Requirements Specification

In this phase we define the requirements of the system. The Requirements Specification describes the things in the system and the actions that can be done on these things. The requirements of the system are:

1. An image dataset (Flicker 8k), this dataset should have minimum two batches including one for training of the NN and second for testing.
2. A Neural Network Model (ResNet-50), for classification of the images.
3. High Level API such as Keras or TensorFlow.

3.3 Planning and Scheduling

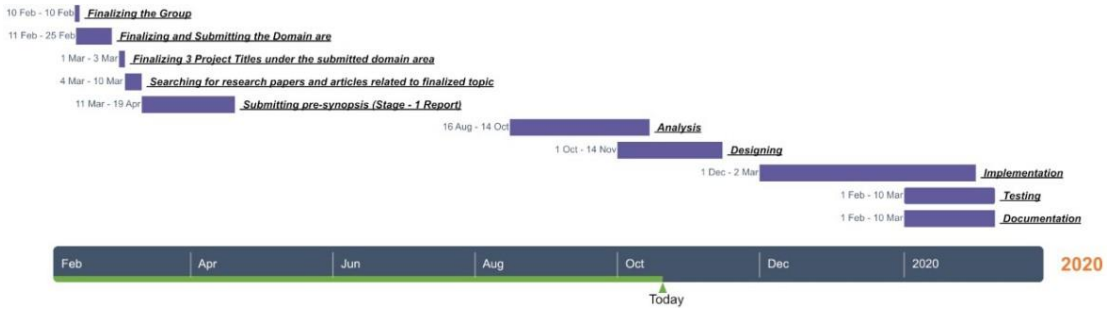


FIGURE 3.1: Gantt Chart

3.4 Software and Hardware Requirements

Hardware Requirement: A computer system having a multi-core processor, minimum of 8GB RAM, storage of min. 500GB and input and output peripherals.

Software Requirements: A web browser to run Google Colab.

Chapter 4

SYSTEM DESIGN

4.1 Basic Modules

Our project consists of these following modules 1. Image Dataset, 2. Neural Network Model with convolution layers to extract features. 3.DataBase to store outputs. Image Dataset is required for the purpose of training and validation of the neural network. Neural Network is required for making classification of images. Database to store the output tagged images.

4.2 Logic Diagrams

4.2.1 Use Case Diagram

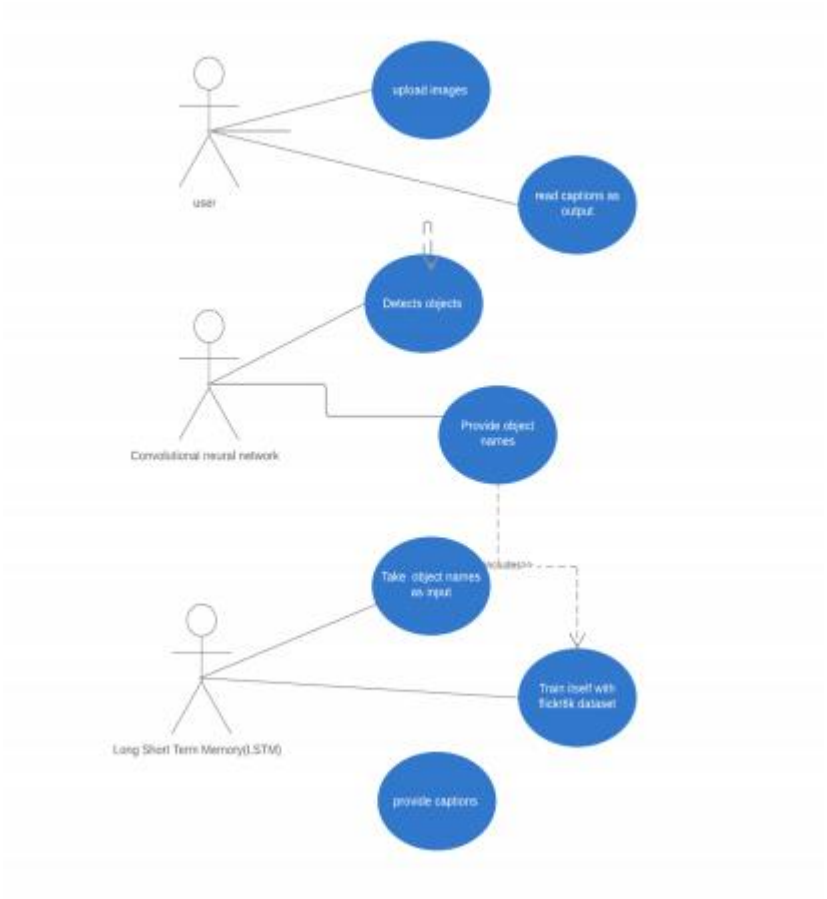


FIGURE 4.1: UseCase Diagram

4.2.2 Activity Diagram

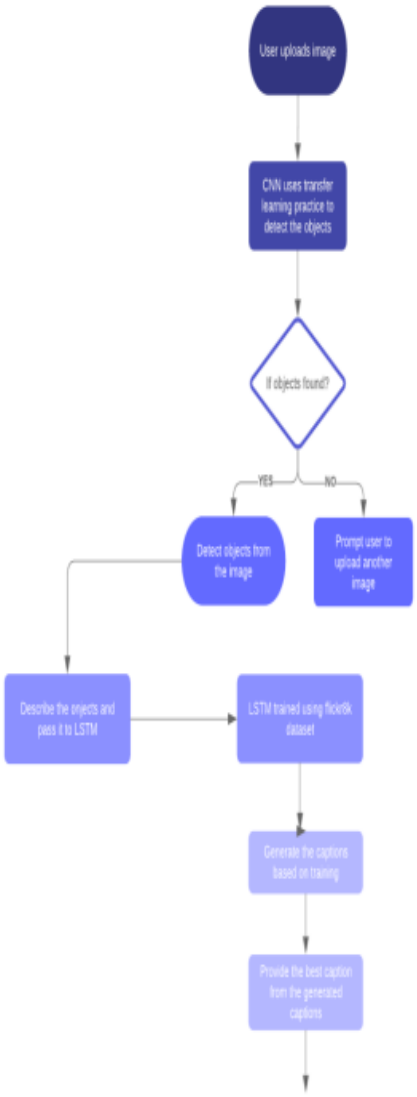
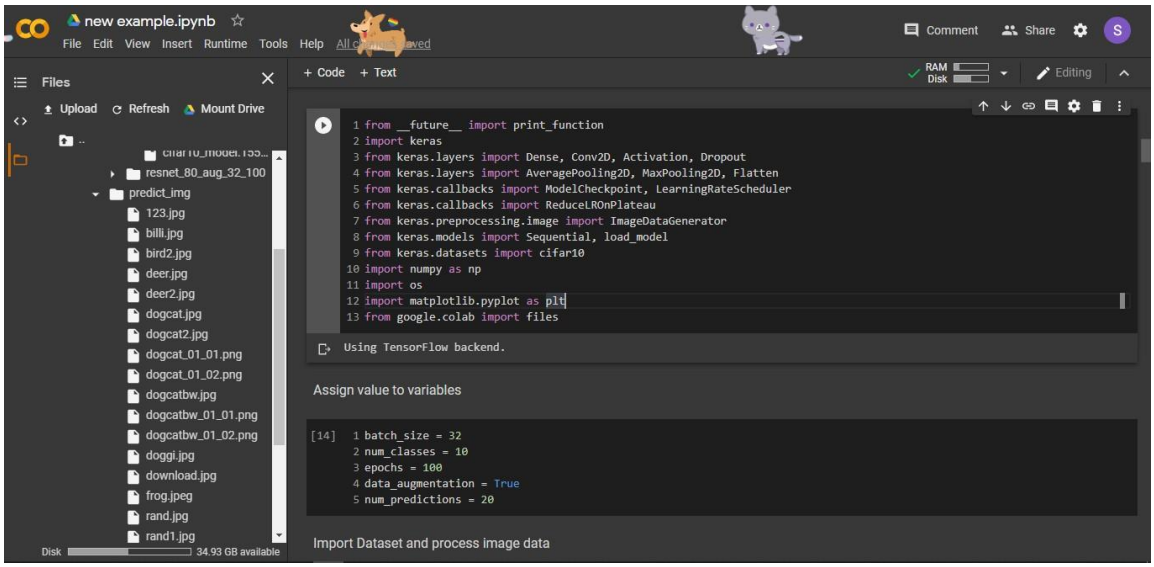


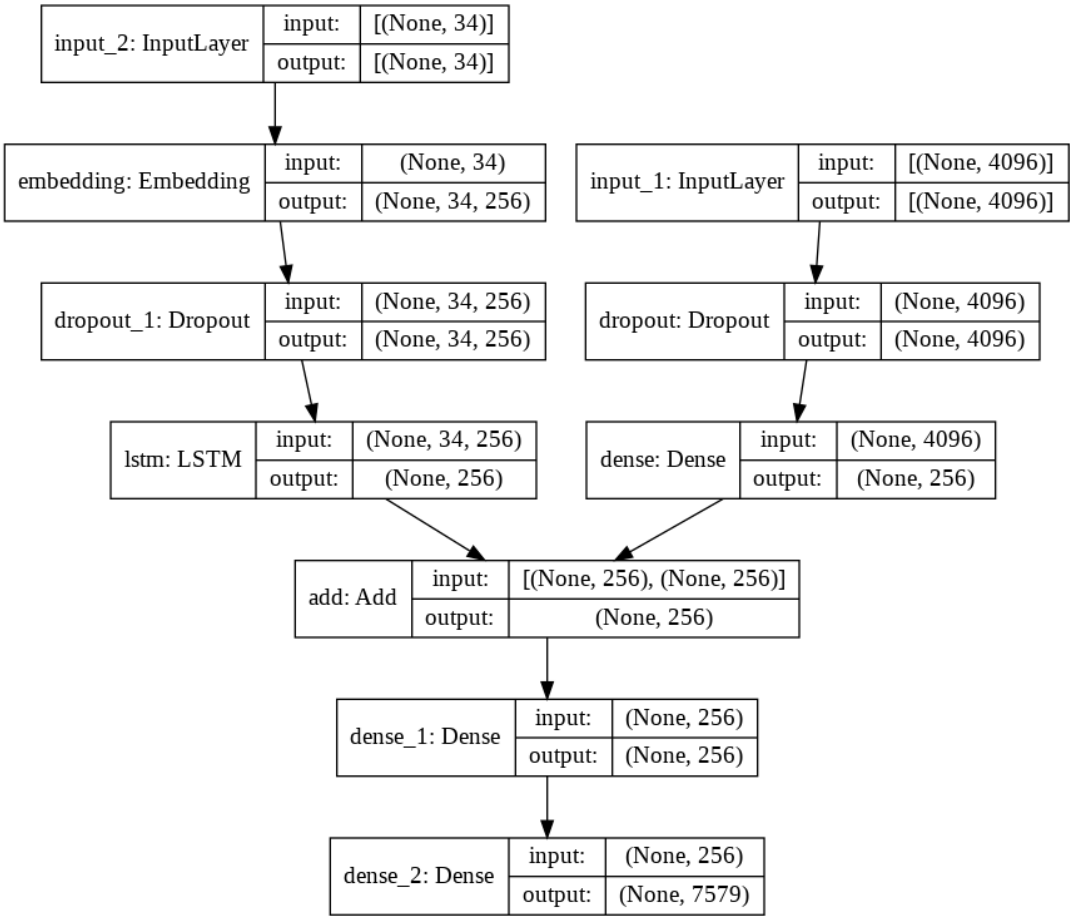
FIGURE 4.2: Activity Diagram

4.3 User interface design

Colaboratory is a research tool for machine learning education and research. It's a Jupyter notebook environment that requires no setup to use. Colaboratory is a research project that is free to use. Colaboratory allows you to use and share Jupyter notebooks with others without having to download, install, or run anything on your own computer other than a browser. Colaboratory provides with computational resources (CPU, GPU) for free. Code is executed in a virtual machine dedicated to user's account. User can upload files to googles cloud.



4.4 Model Architecture:



Model Architecture

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Implementation Approaches

This project is implemented in googlecolab. The main focus is done for development of a system that can classify and annotate images. The system is able to classify images from 10 different classes. To implement this, we used **keras** API. To achieve this the plan was to train a neural network model which will be used to classify the input image form the ten classes. A model with high accuracy is required. A image data set for the training process. Annotation of this classified images can be done by some simple code once images are accurately classified.

5.2 Coding Details and Coding Efficiency

```
# extract features from each photo in the directory
def extract_features(directory):
  # load the model
  model = VGG16()
  # re-structure the model
  model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
  # summarize
  print(model.summary())
  # extract features from each photo
  features = dict()
  for name in listdir(directory):
    # load an image from file
    filename = directory + '/' + name
    image = load_img(filename, target_size=(224, 224))
    # convert the image pixels to a numpy array
    image = img_to_array(image)
    # reshape data for the model
    image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
    # prepare the image for the VGG model
    image = preprocess_input(image)
    # get features
    feature = model.predict(image, verbose=0)
    # get image id
    image_id = name.split('.')[0]
    # store feature
    features[image_id] = feature
    print('>%s' % name)
  return features
```

Figure 1 . Loading VGG16 Model

```
# fit a tokenizer given caption descriptions
def create_tokenizer(descriptions):
  lines = to_lines(descriptions)
  tokenizer = Tokenizer()
  tokenizer.fit_on_texts(lines)
  return tokenizer

# calculate the length of the description with the most words
def max_length(descriptions):
  lines = to_lines(descriptions)
  return max(len(d.split()) for d in lines)

# create sequences of images, input sequences and output words for an image
def create_sequences(tokenizer, max_length, desc_list, photo, vocab_size):
  X1, X2, y = list(), list(), list()
  # walk through each description for the image
  for desc in desc_list:
    # encode the sequence
    seq = tokenizer.texts_to_sequences([desc])[0]
    # split one sequence into multiple X,y pairs
    for i in range(1, len(seq)):
      # split into input and output pair
      in_seq, out_seq = seq[:i], seq[i]
      # pad input sequence
      in_seq = pad_sequences([in_seq], maxlen=max_length)[0]
      # encode output sequence
      out_seq = to_categorical([out_seq], num_classes=vocab_size)[0]
      # store
      X1.append(photo)
      X2.append(in_seq)
      y.append(out_seq)
  return array(X1), array(X2), array(y)
```

Figure 2. Creating sequences of images, input sequences and output words for an image

```python
# define the captioning model
def define_model(vocab_size, max_length):
  # feature extractor model
  inputs1 = Input(shape=(4096,))
  fe1 = Dropout(0.5)(inputs1)
  fe2 = Dense(256, activation='relu')(fe1)
  # sequence model
  inputs2 = Input(shape=(max_length,))
  se1 = Embedding(vocab_size, 256, mask_zero=True)(inputs2)
  se2 = Dropout(0.5)(se1)
  se3 = LSTM(256)(se2)
  # decoder model
  decoder1 = add([fe2, se3])
  decoder2 = Dense(256, activation='relu')(decoder1)
  outputs = Dense(vocab_size, activation='softmax')(decoder2)
  # tie it together [image, seq] [word]
  model = Model(inputs=[inputs1, inputs2], outputs=outputs)
  # compile model
  model.compile(loss='categorical_crossentropy', optimizer='adam')
  # summarize model
  model.summary()
  plot_model(model, to_file='model.png', show_shapes=True)
  return model
```

Figure 3. Defining the caption Model

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_2 (InputLayer)            [(None, 34)]         0
__________________________________________________________________________________________________
input_1 (InputLayer)            [(None, 4096)]       0
__________________________________________________________________________________________________
embedding (Embedding)           (None, 34, 256)      1940224     input_2[0][0]
__________________________________________________________________________________________________
dropout (Dropout)               (None, 4096)         0           input_1[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 34, 256)      0           embedding[0][0]
__________________________________________________________________________________________________
dense (Dense)                   (None, 256)          1048832     dropout[0][0]
__________________________________________________________________________________________________
lstm (LSTM)                     (None, 256)          525312      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, 7579)         1947803     dense_1[0][0]
==================================================================================================
Total params: 5,527,963
Trainable params: 5,527,963
Non-trainable params: 0
```

Figure 4. Model

```python
# define the model
model = define_model(vocab_size, max_length)
# train the model, run epochs manually and save after each epoch
epochs = 20
steps = len(train_descriptions)
for i in range(epochs):
  # create the data generator
  generator = data_generator(train_descriptions, train_features, tokenizer, max_length, vocab_size)
  # fit for one epoch
  model.fit_generator(generator, epochs=1, steps_per_epoch=steps, verbose=1)
  # save model
  model.save('model_' + str(i) + '.h5')
```

Figure 5. Training the Model

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:1844:
  warnings.warn('`Model.fit_generator` is deprecated and '
6000/6000 [==============================] - 854s 141ms/step - loss: 5.1216
6000/6000 [==============================] - 844s 141ms/step - loss: 3.8869
6000/6000 [==============================] - 840s 140ms/step - loss: 3.6258
6000/6000 [==============================] - 831s 139ms/step - loss: 3.4869
6000/6000 [==============================] - 823s 137ms/step - loss: 3.3856
6000/6000 [==============================] - 831s 138ms/step - loss: 3.3052
6000/6000 [==============================] - 826s 138ms/step - loss: 3.2558
6000/6000 [==============================] - 835s 139ms/step - loss: 3.2147
6000/6000 [==============================] - 844s 141ms/step - loss: 3.1806
6000/6000 [==============================] - 839s 140ms/step - loss: 3.1653
6000/6000 [==============================] - 839s 140ms/step - loss: 3.1324
6000/6000 [==============================] - 832s 139ms/step - loss: 3.1243
6000/6000 [==============================] - 817s 136ms/step - loss: 3.0970
6000/6000 [==============================] - 809s 135ms/step - loss: 3.0873
6000/6000 [==============================] - 842s 140ms/step - loss: 3.0798
6000/6000 [==============================] - 835s 139ms/step - loss: 3.0689
6000/6000 [==============================] - 828s 138ms/step - loss: 3.0640
6000/6000 [==============================] - 849s 141ms/step - loss: 3.0509
6000/6000 [==============================] - 846s 141ms/step - loss: 3.0518
6000/6000 [==============================] - 840s 140ms/step - loss: 3.0482
```

Figure 6. Trained Models

```python
#the below function evaluates the skill of the model
def evaluate_model(model, descriptions, photos, tokenizer, max_length):
    actual, predicted = list(), list()
    for key, desc_list in descriptions.items():
        prediction = generate_desc(model, tokenizer, photos[key], max_length)
        actual_desc = [d.split() for d in desc_list]
        actual.append(actual_desc)
        predicted.append(prediction.split())

    print('BLEU-1: ', corpus_bleu(actual, predicted, weights=(1.0, 0, 0, 0)))
    print('BLEU-2: ', corpus_bleu(actual, predicted, weights=(0.5, 0.5, 0, 0)))
    print('BLEU-3: ', corpus_bleu(actual, predicted, weights=(0.3, 0.3, 0.3, 0)))
    print('BLEU-4: ', corpus_bleu(actual, predicted, weights=(0.25, 0.25, 0.25, 0.25)))

def max_length(descriptions):
    lines = to_lines(descriptions)
    return max(len(d.split()) for d in lines)
```

Figure 7. Evaluating the skills pf model using BLEU

```python
def generate_desc(model, tokenizer, photo, max_length):
    #start the generation process
    in_text = 'startseq'
    #iterating over the max_length since the maximum length of the description can be that only
    for i in range(max_length):
        #integer ncoding input sequence
        sequence = tokenizer.texts_to_sequences([in_text])[0]
        #padding the input
        sequence = pad_sequences([sequence], maxlen=max_length)
        #predicting next word
        #the predict function will return probability
        prob = model.predict([photo,sequence], verbose=0)
        #converting the probability to integer
        prob = argmax(prob)
        #calling the word_for_id function in order to map integer to word
        word = word_for_id(prob, tokenizer)
        #breaking if word cannot be mapped
        if word is None:
            break
        #appending as input
        in_text += ' ' + word
        #break if end is predicted
        if word == 'endseq':
            break
    return in_text
```

Figure 8. Results of BLEU

```
# load the tokenizer
tokenizer = load(open('tokenizer.pkl', 'rb'))
# pre-define the max sequence length (from training)
max_length = 34
# load the model
model = load_model('/content/drive/MyDrive/Image Captioning Project(II)/model_16.h5')
# load and prepare the photograph
photo = extract_features('/content/drive/MyDrive/Image Captioning Project(II)/Flicker8k_Dataset/1009434119_febe49276a.jpg')
# generate description
description = generate_desc(model, tokenizer, photo, max_length)

query = description
stopwords = ['startseq','endseq']
querywords = query.split()

resultwords  = [word for word in querywords if word.lower() not in stopwords]
result = ' '.join(resultwords)
filename = "/content/drive/MyDrive/Image Captioning Project(II)/Flicker8k_Dataset/1009434119_febe49276a.jpg"
display(Image(filename))
print(result)
```

Figure 9. Providing image to model for captioning

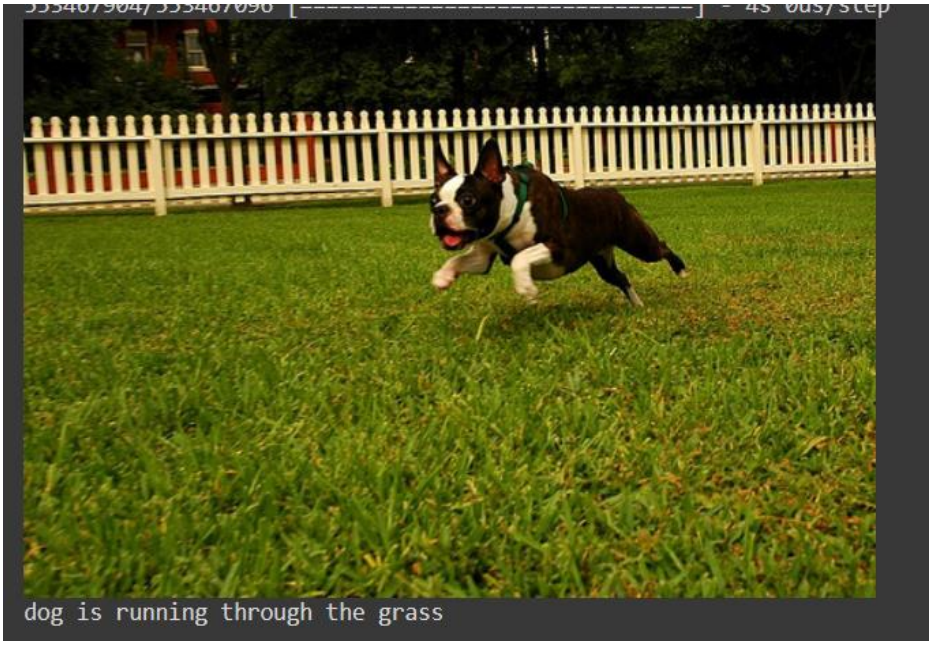
```
filename = "/content/drive/MyDrive/Image Captioning Project(II)/Flicker8k_Dataset/1015118661_980735411b.jpg"
display(Image(filename))
model = load_model('/content/drive/MyDrive/Image Captioning Project(II)/model_16.h5')
photo = extract_features('/content/drive/MyDrive/Image Captioning Project(II)/Flicker8k_Dataset/1015118661_980735411b.jpg')
max_length = 34
description = generate_desc(model, tokenizer, photo, max_length)
query = description
stopwords = ['startseq','endseq']
querywords = query.split()

resultwords  = [word for word in querywords if word.lower() not in stopwords]
result = ' '.join(resultwords)
print(result)
```

Figure 10. Providing image to model for captioning

Chapter 6

RESULTS AND DISCUSSION

6.1 Test Reports



output 1

Output 2

Chapter 7

CONCLUSIONS

7.1 Conclusion

- Through this project, we learned about the deep learning techniques used for image captioning problem.
- We learned that the result of generated captions is influenced by the training dataset.
- The Flickr8k dataset contains many outdoor images of humans and dogs and our model gives better results on outdoor images with people and dogs and confuses many different things in other images to dogs and people.
- We have implemented pre-trained CNN and LSTM model and used BLUE evaluation metrics.
- The Bilingual Evaluation Understudy Score, or BLEU for short, is a metric for evaluating a generated sentence to a reference sentence. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.
- We have achieved an Effective BLUE score of 0.5179 for our model.

7.2 Future Scope of the Project

Future scope of the project might be to improve on segmentation technique. To store newly annotated image into a separate database. As segmentation is a core part of system a better technique is required to be developed in future. Using a image database which has high quality images will improve the overall systems performance.

Bibliography

[1] Claudio Cusano, Gianluigi Ciocca. Image annotation using SVM.

[2] Chih-Fong Tsai, Chihli Hung. Automatically Annotating Images with Keywords: A Review of Image Annotation Systems.Recent Patents on Computer Science 2008, 1, 55-68, 1874- 4796 /08 c 2008 Bentham Science Publishers Ltd.(2007)

[3] Junhua Mao Deep Captioning with Multimodel Recurrent Neural Networks (M-RNN). Published as a conference paper at ICLR 2015.(2015).

[4] Aanchan K Mohan, Marwan A. Tokri Automatic Image Annotation using Neural Networks

[5] Ivor Wai-Hung Tsang, Qi Mao. Objective-Guided Image Annotation. IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 22, NO. 4, 1057–7149/ c. 2012 IEEE.(2013)

[6] Quanzeng You, Hailin Jin. Image Captioning with Semantic Attention.

[7] Andrez Karpathy, Li Fei-Fei. DeepVisual-Semantic Alignments for Generating Image Descriptions. (2015).

[8] YunchaoGong, ThomasK.Leung Deep Convolutional Ranking for Multilabel Image Annotation. (2014).

[9] Yin-Fu Huang, Hsin-Yun Lu Automatic Image Annotation Using Multiobject Identification. Fourth Pacific-Rim Symposium on Image and Video Technology,978-0-7695-4285-0/10 c 2010 IEEE DOI 10.1109/PSIVT.2010.71. (2010).

[10] Nalin Kanungo, Gyayak Sanghi Automatic Multi-Label Image Annotation for Smart Cities. IEEE Region 10 Symposium (TENSYMP), 978-1-5090-6255-3/17/c2017 IEEE. (2017).

[11] Gabriel Mihai, Liana Stanescu. Automated Annotation of Natural Images Using an Extended Annotation Model. International Journal of Computer Science and Applications Technomathematics Research Foundation Vol. 9, No. 3, pp. 1 – 19. (2012)

[12] Mahdia Bakalem, Nadjia Benblidia. A Comparative Image Annotation. 978-1-4799-4796-6/13/ c 2013 IEEE. (2012).

Websites:

https://keras.io/

https://keras.io/getting-started-30-seconds-to-keras https://keras.io/layers/convolutional/

https://keras.io/layers/pooling/

https://keras.io/layers/advanced-activations/

https://keras.io/preprocessing/image/

https://keras.io/losses/

https://keras.io/datasets/

https://keras.io/examples/cifar10cnn/