VISUAL QUERY ANSWERING SYSTEM

A PROJECT REPORT

Submitted by

S. ARUN BAALAAJI [Reg No: RA1511004010223]
ABHIJIT SRIDHAR [Reg No: RA1511004010229]
R. RAVINDRA VIKRAM [Reg No: RA1511004010244]
PRATEEK RALHAN [Reg No: RA1511004010342]

Under the guidance of

Dr. A. RUHAN BEVI
(Associate Professor, Department of Electronics and Communication Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION

ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

S.R.M. Nagar, Kattankulathur, Kancheepuram District

APRIL 2019

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University u/s 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled “**VISUAL QUERY ANSWERING SYSTEM**” is the bonafide work of “**S. ARUN BAALAAJI [Reg No: RA1511004010223], ABHIJIT SRIDHAR [Reg No: RA1511004010229], R. RAVINDRA VIKRAM [Reg No: RA1511004010244], PRATEEK RALHAN [Reg No: RA1511004010342]**” who carried out the project work under my supervision along with his batch mates. Certified further, that to the best of my knowledge the work reported herein does not form any other project report on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

Date: Project Supervisor Head of the Department

Submitted for University Examination held on .... in the Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Kattankulathur.

Date: Internal Examiner External Examiner

DECLARATION

We hereby declare that the Major Project entitled “**VISUAL QUERY ANSWERING SYSTEM**” to be submitted for the Degree of Bachelor of Technology is our original work as a team and the dissertation has not formed the basis of any degree, diploma, associateship or fellowship of similar other titles. It has not been submitted to any other University or institution for the award of any degree or diploma.

Place:

Date:

S. Arun Baalaaji
[RA1511004010223]

Abhijit Sridhar
[RA1511004010229]

R. Ravindra Vikram
[RA1511004010244]

Prateek Ralhan
[RA1511004010342]

ACKNOWLEDGEMENTS

We warmly thank the people who helped and supported us in the preparation of this project. We would like to thank our Director, Dr. C. Muthamizhcelvan, Faculty of Engineering and Technology, SRM Institute of Science and Technology, Chennai.

We wish to express our deepest gratitude to Dr. T. Rama Rao, Head of the Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Chennai, for giving us the opportunity to take part in this project. We would also like to thank our Project Coordinator, Dr. K. Kalimuthu, Associate Professor in the Department of Electronics and Communication Engineering at the SRM Institute of Science and Technology, Chennai, for his valuable contributions to the project reviews. We take great pleasure in expressing our thanks to our guide, Dr.A.Ruhan Bevi, Associate Professor, Department of Electronics and Communication Engineering, SRM Institute of Science and Technology, Chennai for her valuable advice, constant encouragement, personal attention, quick help and providing us with an excellent atmosphere for doing research. Throughout her work, despite her busy schedule, she has provided us with warm and cordial support to carry out this research work.

Authors

ABSTRACT

Problems related to the intersection of vision and language are of considerable importance, both as difficult research questions and for the richness of the applications they allow. However, the inherent structure in our world and prejudices in our language tend to be a simpler learning signal than visual modalities, resulting in models that ignore visual information, giving an exaggerated sense of their abilities . The project mainly focuses on the research technique called 'Hierarchical Co-Attention' model.
The hierarchical co-attention model has been found to work in an efficient manner relating image systems. The project involves analysing the weak points of the hierarchical co-attention model and to develop a new model based on it to provide better performance metrics. The project involves building spatial maps of interest on images to improve the understanding of the image. The metric to be analysed in the project is accuracy and it is evaluated as per the Visual Question Answering challenge. The accuracy here reflects on the inter-human variability. To tackle the problem of understanding the human language we plan to build a custom Long Short Term Memory (LSTM) network. The project also involves analysing various datasets such as MS-COCO, Visual Question Answering (VQA) dataset and finding out the best combination of them to get better accuracy. The project will be trained and tested on a cloud computing service which provides Graphical Processing Unit (GPU) acceleration. The project will be visualized by means of a web application.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

VQA	Visual Question Answering
LSTM	Long Short Term Memory
CNN	Convolutional Neural Networks
PS-LSTM	Parallel Stack Long Short Term Memory
COCO	Common Objects in Context
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
GRU	Gated Recurrent Unit
GPU	Graphical Processing Unit

CHAPTER 1

INTRODUCTION

Problems related to the areas of language and vision are of considerable importance, both as difficult experimentation questions and for the richness of the tasks they allow. However, the deep-rooted structure in our world and prejudices in the language pose to be an easier learning signal than visual techniques, leading to frameworks that overlook perceptible information, giving an exaggerated sense of their abilities .

One of the most overwhelming and exhilarating accomplishments in the domain of Intelligent frameworks and architectures has been the introduction of Visual Question Answering (VQA)[1], a new and exciting problem that makes use of Natural Language Processing and Computer Vision for measuring the ability of a system for image understanding and generating inferences beyond object recognition, segmentation and image captioning. One of the primary goals of research in visual based Artificial Intelligence is to design systems that can understand and reply to questions pertaining to visual data. Visual Question Answering is a research field that is emerging at an enormous pace in the past few years and has been getting widespread attention from the entire Machine Learning community throughout the globe. It mainly comprises of showcasing an image to a computer system and asking a question about that image which the computer must answer, which can be in any format: open ended (yes / no), a single word, a short phrase, fill in the blank type or a multiple choice based answer. It is an inter-disciplinarily appealing task that works on the uniquely identified fields of Computer Vision and Natural Language Processing. Image classification, segmentation, sectoring [7,8] and other computer vision techniques are employed to understand the image under inspection while NLP techniques are used to understand and extract meaningful information out of the question, which are then effectively combined to generate answers to the questions in context with the image.

Thus, our main objective is to develop a unique framework that will overcome all the drawbacks of the existing Visual Question Answering based Systems, with higher degree of accuracy coupled with better image captioning techniques.

CHAPTER 2

LITERATURE SURVEY

2.1 Image Captioning with Triple-Attention and Stack Parallel LSTM

Xinxin Zhu, Lixiang Li, Jing liu, Ziyi Li, Haipeng Peng, Xinxin Niu; Journal of Neurocomputing (Elsevier), Volume 319, 30 November, Pages 55-65

Image captioning is used to describe the contents of an image. [1] reviews two proposed modifications for performance improvement of image understanding. The first concerns a unique method called Triple Attention, that uses the context information of the image at each step of LSTM, and proposes a structural overhaul of the basic schema LSTM the parallel arrangements of LSTM are adopted, they are called Parallel Stack Long Short Term Memory (PS-LSTM). These models achieved a significant increase in performance over the normal structure. The attention model is not only put into usage at the entry step of the **LSTM's!** (**LSTM's!**) hidden unit previously, but also at the exit step of each hidden unit at the present time. At every input step LSTM, conditional text embedding is used with the incorporation of words and images. Thus, the entry now contains textual and graphical information throughout the sentence generation process. Parallel LSTM are used to enter the input context information. Each of the parallel cells LSTM is initialized with different parameters, thus giving rise to different sets of characteristics extracted from the input. Then, a middle grouping layer is used to concatenate all outputs with the LSTM to get similar dimensions as that of the initial LSTM. Many experiments have been performed on the Common Objects in Context (COCO) dataset, and state-of-the-art results have been obtained, showing better performance than existing methods.

2.2 An Analysis of Convolutional Neural Networks For Image Classification

Neha Sharma, Vibhor Jain, Anju Mishra Journal of Procedia Computer Science (Elsevier), Volume 132, Pages 377-384, DOI: https://doi.org/10.1016/j.procs.2018.05.198

This article is about performance analysis of the popular Convolutional Neural Networks (CNN) to identify objects in real-time video streams. Alexnet, GoogleNet and ResNet50 are the most popular CNN for detecting and classifying image categories from images, while the reference datasets commonly used to evaluate their performance are ImageNet files , CIFAR10, CIFAR100 and MNIST. The authors analyze the performance of these three networks on the three most popular datasets. ImageNet, CIFAR10 and CIFAR100, because the true capacity of any model can only be revealed by its evaluation on several datasets. The trained models have even been exposed to videos in the form of a test datset, which is another measure to evaluate their performance.

The analysis, implementation, and testing of these models led them to conclude that GoogleNet and ResNet50 performed better than AlexNet, and even to recognize objects with greater accuracy. GoogleNet appeared to perform significantly better than the other two for the CIFAR100 and CIFAR10 datasets with 70% accuracy, while the other models had 13% and 53% accuracy respectively. The end results suggest that learning transfer networks work better than existing networks and seem more accurate.

2.3 Rethinking the Inception Architecture for Computer Vision

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens; CVPR-2015, arXiv:1512.00567 [cs.CV]

In this article, the authors discuss different ways to scale up existing models through appropriate convolutions while trying to reduce inefficiencies. Validation is performed on the validation set of the ILSVRC 2012 classification challenge which describes the gains of the conventional validation: 5.6% top-5 error and 21.2% top-1 in single frame validation with a model using less than thirty million parameters and whose computing costs five billion multiply-add by inference. By combining four models and using multiple crops, the results revealed an error of 17.3% top-1 and an error of 3.5% top-5.
In feedback networks, the flow of information across layers is clearly defined. The bottlenecks present for reducing dimensionality should not have drastic compression across the layers. For reduction of the high compression, the dimensions of the representation must gradually decrease through the network layers.
The reduction of the parameters is therefore a characteristic which gives a significant advantage for the efficiency of the calculation, because it accelerates the train network. Factoring is a technique that can be used to reduce parameters. Large kernel sizes such as 5x5/7x7 impose a large workload on computational tasks. Such a kernel/filter is replaced by a multilayer network (in this way the network gains depth). For example, a fully connected 5x5 layer can be substituted by adding one 3x3 convolution layer, with a fully connected layer over it. With same input size, weights between the layers are shared, which reduces the parameters. In addition, many activations are reused between layers, which greatly reduces the computational burden.
The model has been tested on low resolution images with different sizes of receiver fields. Maximum precision 1 increased from 75.2% for a field size of 79x79 to 76.6% for a field size of 299x299.

2.4 Deep Residual Learning for Image Recognition

Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun CVPR-2015, arXiv:1512.03385 [cs.CV]

This article addresses the problem of training networks that are deeper than conventional models. The authors propose a framework to speed up the process and improve performance. The technique of residual learning was brought out by the ResNet-152 model, which offers better accuracy even as network depth increases. This also leads to better optimization of identity mapping. A residual function is defined for better optimization of the identity mapping. A function is better optimized if it is pushed to zero rather than to a particular value. It can enhance the enforcement of model training, especially when it possesses a deep network with 20 layers or more. When deep networks begin to converge, a problem appears. As the network's complexity and size rises, the accuracy saturates after a point, after which it decreases rapidly. The authors aim to rectify this problem by using a deep residual learning framework.

These networks were then tested on the ImageNet dataset and the performance of these residual networks at 18 and 34 layers or ResNets proved to be much better than the usual networks used. Following this, the ResNet depth was increased to 50, 101 and 152 layers. It was found that while the ResNet 152 model had the greatest depth, it also showed the best performance among all. It has been found that for models with more than a thousand layers, such as the 1202 layer model, the over-adjustment problem began to appear.

2.5 Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks

Nils Reimers, Iryna Gurevych,2017, arXiv:1707.06799v2 [cs.CL]

Choosing the optimal settings in a LSTM can help achieve peak performance in the tasks. This article discusses the various methods that can be used to optimize hyperparameters and then system performance. You perform these assessments for five voice sequence markup tasks: voice parts, block splitting, NER, entity detection, and even detection. More than 50,000 different configurations were evaluated and it was concluded that factors such as the use of word embedding or the concluding layer tuning had a much greater outcome on system enforcement than succeeding factors such as the number of layers or storage units.
Authors used BiDirectional LSTM in the various sequence tagging tasks and evaluated them using a large amount of data. For each architecture, they determined the architectures and parameter values that performed best.
Based on the results obtained, they found that the use of character embedding was optimal for such tasks, and tested various nestings of this nature. They found that using an Adam optimizer with the Nesterov pulse enabled the fastest convergence and that the gradients Normalization led to a significant improvement. In terms of network architecture, they finally concluded that the best performance was achieved with two or more recurring layers, and about 100 recurring units per network (LSTM) gave good results. The results of this article have adjusted our LSTM model to lessen the overall model loss.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Objectives

- To define an improvised CNN architecture for image understanding using the existing models with enhanced Image Captioning.
- To design a custom LSTM network for better human language understanding.
- To concatenate the two frameworks into a single unique and optimized architecture and achieve better accuracy and optimized results than the existing VQA systems.

3.2 Problem statement

Existing CNN architectures are defined for image understanding that are limited only to popular datasets. There is a need to improvise the existing architectures for image understanding outside the popular datasets and develop a more generic framework that can be implemented with enhanced capabilities. VQA is an upcoming research area that works in collaboration with Computer Vision and Natural Language Processing, thus posing many challenges to the researchers worldwide.

3.3 Dataset Analysis

Recently, a plethora of datasets have been developed and released to analyze this domain that contain challenging questions about images based on NLP.Accurate interpretable abilities such as object recognition attribute recognition, spatial relationship determination as well as higher order skills dealing with logical inferential statistics is employed for making comparisons etc. and answering these questions with proper precision and correction. Here, we describe some of the prominent datasets and models/methodologies that have been used to tackle the task of VQA, as well as how effectively they perform when deployed in different environments.

3.3.1 MS COCO: Common Objects in Context

It is mainly composed of gathered images of common context and they provide a better reflection of day to day occurrences and providing image captioning and segmentation [6]. Precise object localization is achieved using pre-instance segmentations for labelling of the objects. The dataset comprises of 91 object categories with 82 categories of more than 5000 labelled instances in 328000 images. As compared to popular datasets like ImageNet [10], PASCAL VOC [11] and SUN [12], it tends to possess fewer categories with more instances per image (7.7) that can aid in training object models with the ability of localizing precise 2D objects. They also have 5 captions per image and 250,000 people with key points, thus making it as the best suited dataset for the preparation of our VQA Architecture.

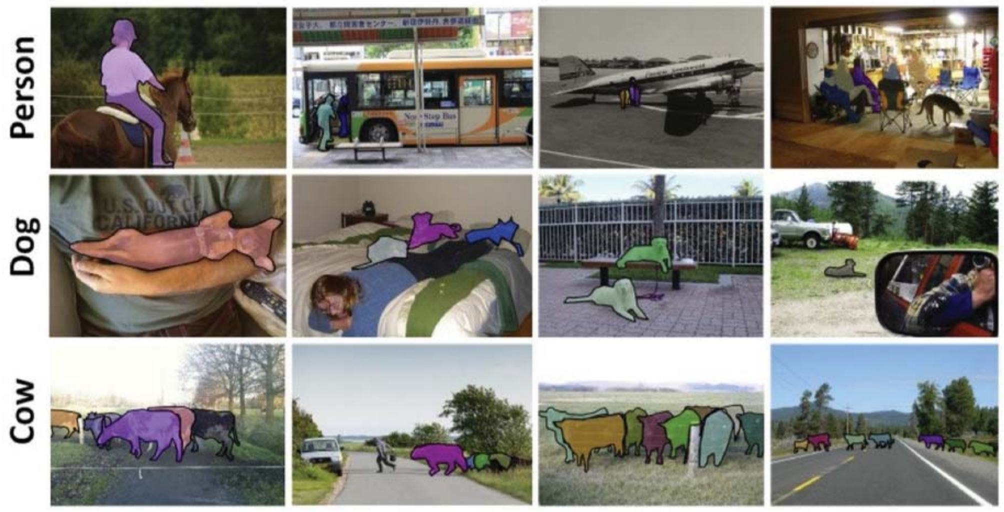


Figure 3.1: MSCOCO Dataset

3.3.2 VizWiz VQA Dataset

VizWiz [15] consists of more than 31,000 questions asked by visually challenged humans, who have taken a photo with a mobile and talked about it and recorded a question about it. Compared to the previous datasets, pictures are taken by visually impaired photographers and are therefore often of bad aspect, questions are spoken and more conversations are given, and most of the visual questions can not be deciphered, making them a sophisticated dataset. In contrast to the existing image data sets, the image quality is poor because of inadequate lighting, improper focus and the frame of interest. Questions tend to be entertaining or suffer from imperfections in audio recording, and other audio content.

3.3.3 CLEVR Dataset

The Compositional Language and Elementary Visual Reasoning (CLEVR) diagnostics dataset [14] , comprises of 100k rendered images with around a million questions that are generated automatically, with 853,000 unique questions. It consists of simple Three-dimensional shapes; simplifying recognition and providing other with the necessary focus on reasoning skills. It also incorporates diverse collection of synonyms for each material, shape, and colour to incorporate Language Diversity. Their experiment demonstrations showcased that the dataset facilitates in-depth analysis as compared to the other VQA datasets[13] due to shortcomings pertaining to Short-Term Memory, Long reasoning chains, spatial relationships and disentangled representations.

3.3.4 The FM-IQA Dataset

The Freestyle Multilingual Image Question Answering (FM-IQA) dataset was constructed by Haoyuan Gao with the aim of training and evaluating their mQA model. With a collection of 150,000 pictures and 310,000 pairs of Chinese questions and their answers supported by their English translations. It initially comprised of 158,000 images from the MS-COCO and has some simple image understanding questions of common knowledge, e.g., the actions of objects (e.g., What is the boy on the road doing?),

Table 3.1: Attribute Analysis of Different Datasets in VQA

Dataset	Images	Total Objects	Object Categories	Attribute per Images	Question Answers
MS-COCO	328000	27472	80	-	-
Image Net	14197122	14167122	21841	-	-
PASCAL detection	11530	27450	20	-	-
SUN attributes	14000	-	-	700	-
VQA	204721	-	-	-	614163
FM-IQA	158392	-	-	-	316193
Visual Genome	108077	3843636	33877	26	1773258
VizWiz	31173	6	7	-	48169
CLEVR	10000	-	-	15	999968

the object class (e.g., Is there any dog in the image?) etc. Additionally, the dataset also contains some questions that need exemplary reasoning skills with clues from vision and language supported by diversified answers.

3.3.5 Visual Genome

Containing over 108,000 images with a mean of 35 objects, 21 object based pair-wise relationships for each image and 26 attributes, the authors [15] canonicalized these constraints and different phases (nouns) in region descriptions and pairs of Question-Answers to Word-Net synsets that is mainly a large scale database of English grammar interlinked by using semantic relations. With multiple region image descriptions and dense object diversity, it is a large scale intersection of MS-COCOs [6] three hundred thousand images and YFCC100Ms (Thomee et al.2016) 100 million images, this one has around [10]8000 images of creative common domain, with image dimensions ranging from 72 pixels to 1280 pixels, accompanied with a mean width of 500 pixels. On an average, it contains 26 attributes per image and provides a hierarchical understanding of pictures that allows an image study from various perspectives; from minute-level information (pixels) like subjects, to interactions involved with inferential statistics, and to deeper cognition based tasks like query replying.

3.3.6 VQA Dataset

With around 123,000 training and validation images and around 81,000 test images from the well-known MSCOCO dataset, this dataset [1] contains 20 human models that belong to numerous castes, creeds, races, etc., with 8 different expressions adjustable limbs for continuous pose variations. The Clipart Set includes more than 100 objects and a variety of poses with 31 animals. It can be used for both indoor and outdoor shooting. Both for the real pictures as In addition to abstract scenes, the same user interface was used. Overall, the VQA dataset has more than 760000 questions. To answer the questions, two different modalities have been suggested: (i) open and (ii) multiple-choice. For sure Special accuracy metrics were used to evaluate the responses generated for the open task, whereas each question for the multiple choice task was designed to specifically generate 18 candidate responses.

3.4 Inception V3

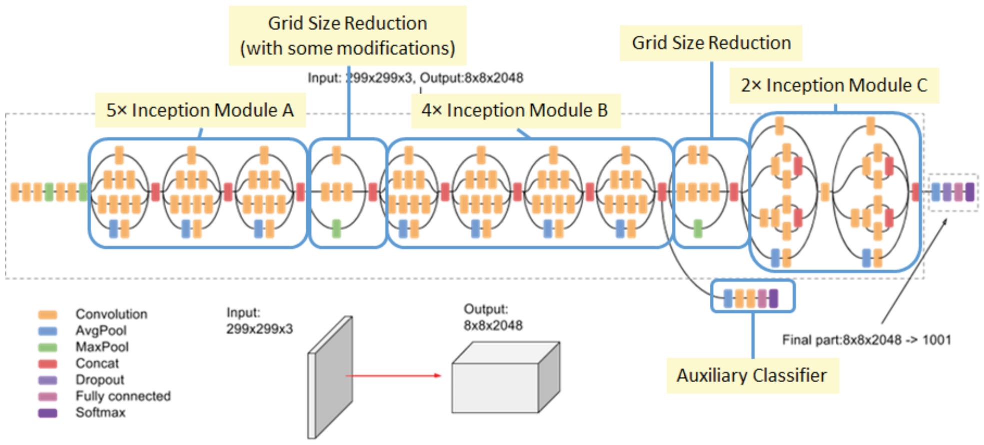


Figure 3.2: Inception v3 model

The Inception v3 model [3] is trained with the ImageNet dataset [10]. It has a top 5 error rate of 3.5% and a top 1 error rate of 17.3%. The last shift can be retrained for new categories. It consists of 1x1, 3x3, 5x5 folding modules that are stacked on top of each other. The Max Pooling layers have Step 2 to halve the resolution of the grid. Dimensional reduction reduces the computational complexity. It processes visual information at different scales and is aggregated to the next level to simultaneously abstract features from previous scales. An auxiliary classifier provides additional regularization and reduces fading of the gradient in the reverse direction. Reduced gradient fading allows subsequent features to learn from previously scaled features. An average pooling layer with a filter size of 5x5 and step 3 is used. Other structural features of this model include a 1x1 convolution with 128 filters for dimension reduction and Rectified Linear Unit (ReLU) activation, a fully connected 1024 unit layer and rectified linear activation, a dropout layer with a 70% to falling output ratio, and a linear layer with Softmax loss as a classifier.

3.5 ResNet 152

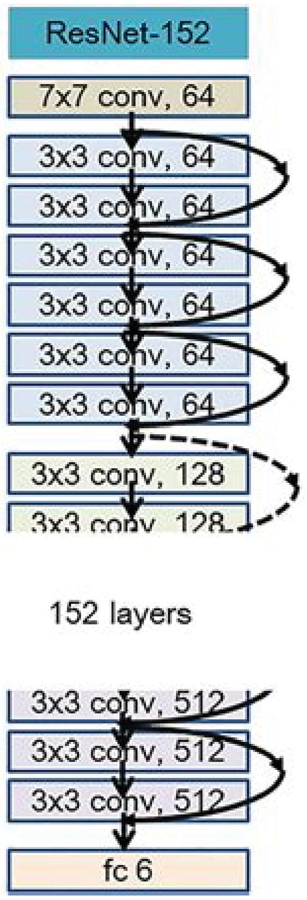


Figure 3.3: Resnet 152 model

The ResNet 152[4] has lower complexity than VGG 16/19 nets. It was originally designed using a residual deep learning framework, which was implemented to enhance the working of deeper networks. One of the advantages of this framework is that the bottle neck design can diminish the complexity of connections while keeping the same performance as a network with a similar number of layers. This allows developing deep layers with comparatively less complexity than a fully connected network. Based on the initial tests with 18 and 34 layers, it was found that the 34-layer network exhibited better performance. The use of skip connections helps alleviate the vanishing gradient problem. By increasing the network depth to 152 layers, 5.71% top-5 error rate is obtained which is much better than VGG-16, GoogLeNet (Inception-v1), and PReLU-Net.

The typical architecture involves resizing the image to a 224x224 form. A batch normalisation is applied in between each convolution layer and the subsequent activation function. Usually, no dropout is used. However, as the network size grows over a 1000 layers, dropout and regularisation are recommended.

3.6 LSTM

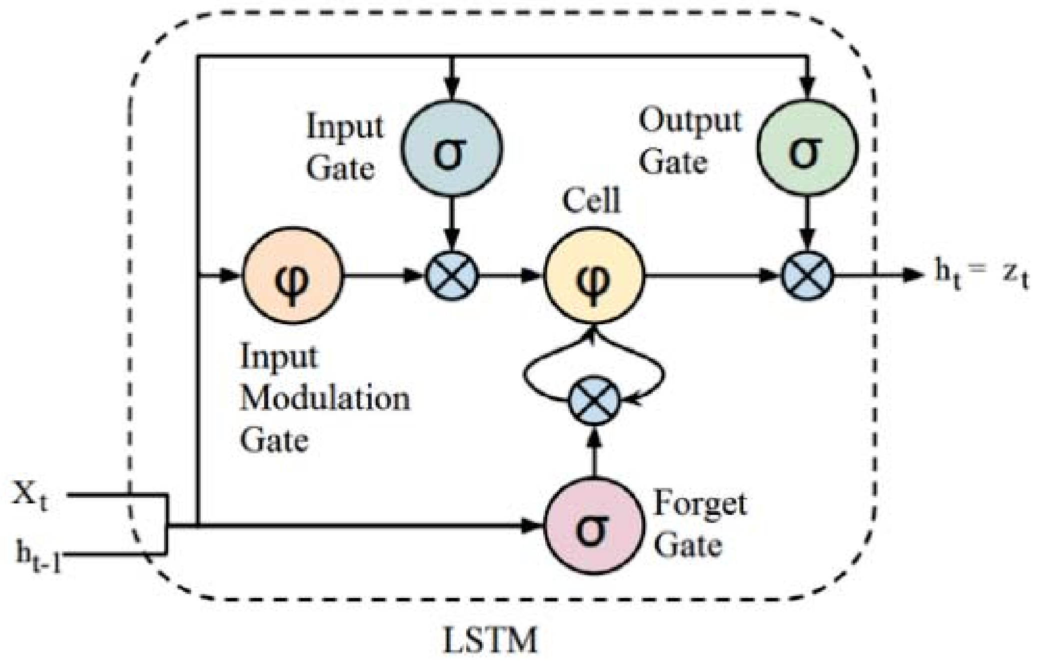


Figure 3.4: LSTM cell

For a lot of memory based applications, the go to networks are Recurrent Neural

Networks, or Recurrent Neural Network (RNN). In a regular RNN cell, an input is fed and an output is determined. However, it also contains an internal hidden state, where the calculations are performed on the input to yield the output. In the next step, the outputs of the current cell as well as the current hidden state are taken as inputs to compute the new outputs. This leads to it possessing a type of short term 'memory', where it can remember the relation between different elements of the input.

However, RNNs are bad at remembering long term data. For example, in the case of text data, it cannot remember relationships between elements if the elements are too far apart. However, this is a common occurrence when dealing with text data, and a solution is required. This problem is called gradient vanishing. RNN suffer from the gradient vanishing and gradient explosion problems. Gradient vanishing occurs when the learning rate becomes increasingly smaller till there is no change in the loss. Gradient explosion occurs when the learning rate is too large, and the network cannot arrive at a minimum loss i.e, it cannot optimise the cost function.

The solution to this comes in the form of the LSTM cell. It has most of the functionalities of the RNN cell, but can handle long term data and is more robust. This provides the solution we required when facing the problem of generating captions for the images. The LSTM[1] cell used three gates, the input, output, and forget gates, which each have their own weight matrices. The input gate is used to modulate the information coming in. It is used to determine what parts of the data to be used in generation of the output. For example, information learnt in the previous time step is useful in making the current prediction, but the same cannot be said for information learnt ten or maybe twenty time steps ago. The forget gate is used to remove any unnecessary parts of the data. For all the information learnt, there might be a lot of redundant or useless information. The forget gate is used to filter out this information. Finally, the output gate acts as an attention mechanism, which decides which part of the output to focus on and make the prediction.

An alternative to the LSTM[10] is the GRU. It offers similar functionality to the LSTM[1] but with a simplified architecture. The gated feedback structure is faster and better in performance than a standard stacked RNN. Hence, it also requires less computing power and executes faster. While the GRU is used primarily for machine translation and speech recognition tasks, the LSTM unit has more diverse applications.

Also, while the GRU performs better on small datasets, the LSTM generally outperforms it in most other cases.

In a GRU, there is a single hidden state. Further, the input and forget gate are merged

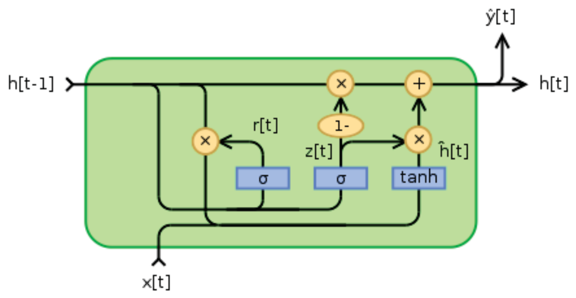


Figure 3.5: GRU cell

into one update gate. This gate does the same job as the input and forget gates in the LSTM. The output gate is removed, and a reset gate is added that breaks the information flow from the previous state.

CHAPTER 4

PROPOSED METHODOLOGY

The approach for Visual Query Answering involves a typical model consisting of Convolutional Neural Networks (CNN)[2] and Long Short Term Memory (LSTM)[1]. The image and language features are combined as shown for a multiple-layered perceptron structure.

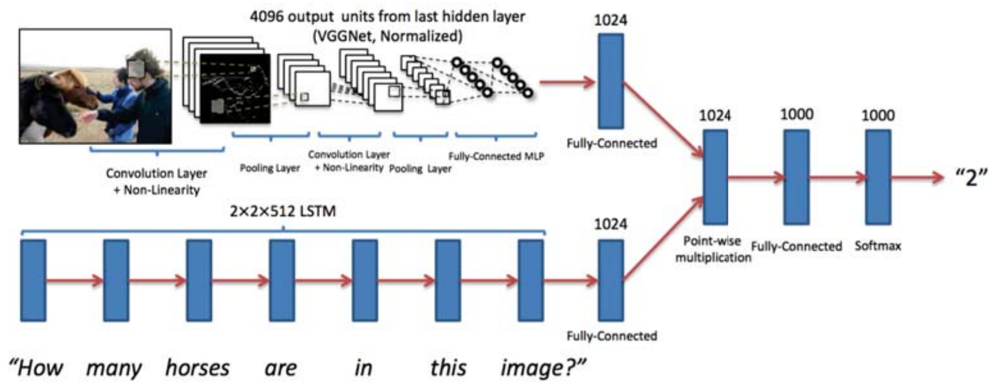
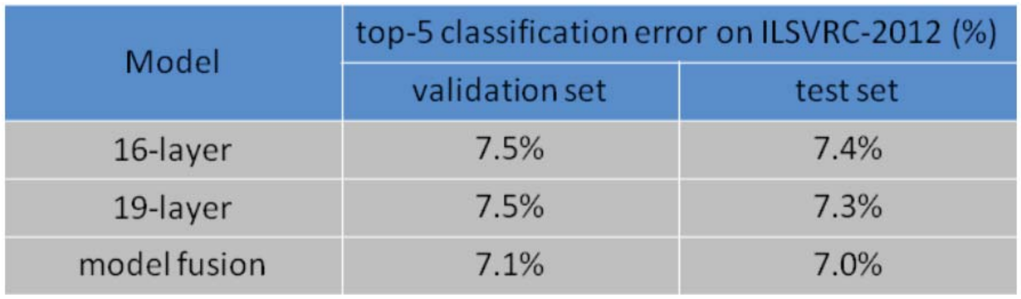


Figure 4.1: Proposed model

4.1 Image Features

Image features are obtained from CNNs. The architecture considered for obtaining the image features is VGGNet. The architectures like GoogleLeNet and ResNet have shown better results than VGGNet however we consider VGGNet because it is simple, portable, versatile, relatively small to use. The simplicity comes from utilizing 3x3 convolution layers on top of each layer, Max pooling decreases the volume size. It comprises of two fully connected layers in connection with the Softmax layer. Each of the fully connected layers have 4096 nodes.The drawbacks of VGGNet include that there exists a pre-training process before the network can actually be trained. This makes the process of training slow-paced (138 million parameters to be trained). VGG16 and VGG19 contain 16 and 19 convolution layers respectively.

For extracting image features a random image is chosen and the model is run upon that



Model	top-5 classification error on ILSVRC-2012 (%)	
	validation set	test set
16-layer	7.5%	7.4%
19-layer	7.5%	7.3%
model fusion	7.1%	7.0%

Figure 4.2: VGGNet Performance

until the last layer is reached. By virtue of the last layers we extract features with 4096 dimensionality.

4.2 Word Embeddings

For extracting question features we need to generate word embedding where each word can be represented in a vector form so that the words can have a relationship between each other. The GloVe model results in a 300 dimensional vector representation of words in the vocabulary. The training methodology of the GloVe model is to obtain optimal values of vectors (words) such that the scalar product of vectors (words) is equal to the logarithm of the words'co-occurrence probability.

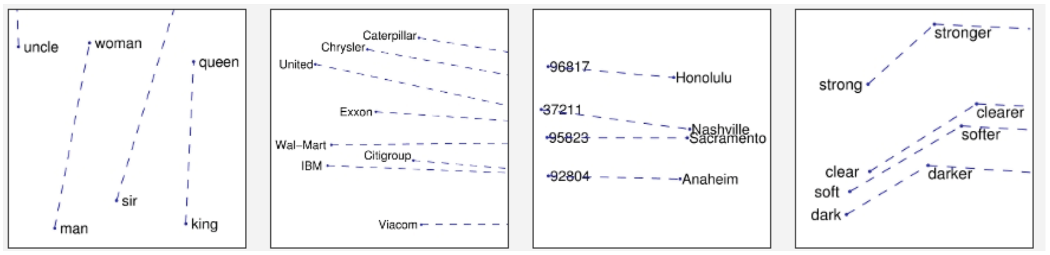


Figure 4.3: Relation between words represented by a distance between their vectors

4.3 VQA Model

The model we consider for a VQA is that which concatenates the Image feature extraction and Word embeddings and runs the multiple layered structure of perceptron model earlier mentioned.
The convolutional models[2] we are using are the ResNet-152, VGG16 and Inception V3 models, in addition to the classifier layers at the end. The classifier layers mainly are composed of batch normalization layer and linear layer. The second one is just a fully connected network and the batch normalization is done to accelerate the pace at which the learning rate is tuned. The ResNet 152 is first sequentialized and then the classifier is added to its last layer. This configuration is compiled to make a final CNN model. The ResNet 152 has the fully connected layer of 2048 input features 1000 neurons at the output. On the other hand, the Inception V3 has multiple layers of convolution stacked on top of each other. Additionally, it utilises an auxiliary classifier, which reduces gradient fading and provides regularisation. This makes the network useful in dealing with sequential data. But VGG16 has been preferred due to its simplistic design and versatility.
Word2Vec is an embedding method that further enhances the performance of the one hot representation. Instead of predicting a word based on a sequence, it looks at the words both preceding and succeeding a target word, i.e, it learns the context in which a word appears. Word2Vec can be implemented in two main ways: Continuous Bag of Words (BOW), where it predicts a word based on the surrounding words, or skip-gram, where it uses a word to predict the context it appears in.Generally, skip-gram is observed to have better performance. While word2vec is a predictive model - a feed-forward neural network that learns vectors to improve the predictive ability, GloVe is a count-based model. Compared to word2vec, GloVe allows for paralle implementaion, which means that its easier to train over more data. It is even said to combine benefits of the word2vev skip-gram model in the word analogy tasks, with those of matrix factorization methods exploiting global statistical information. For long sentences the model proposed has a 3 layered LSTM network for minimizing vanishing gradient problem. In order to obtain decent results the word embeddings are sort of concatenated with extracted image features. As per the performance of the current model the fully connected

layers can be experimented with in order to obtain better results.

Network training involves the use of a stochastic gradient for the distributed machine learning system of Google Cloud. It uses 50 replicas, each running on a Tesla P4 GPU with a batch size of 32 for 100 eras. it is degraded every two epochs with an exponential rate of 0.94. In addition, gradient clipping with Threshold 2.0 stabilizes the workout. Model evaluations are performed using a running average of the parameters calculated over time.

CHAPTER 5

CONCLUSION

By tuning numerous hyper parameters and through extensive training and testing mechanisms, we were able to obtain some decent results using our current improvised architecture.

5.1 Results

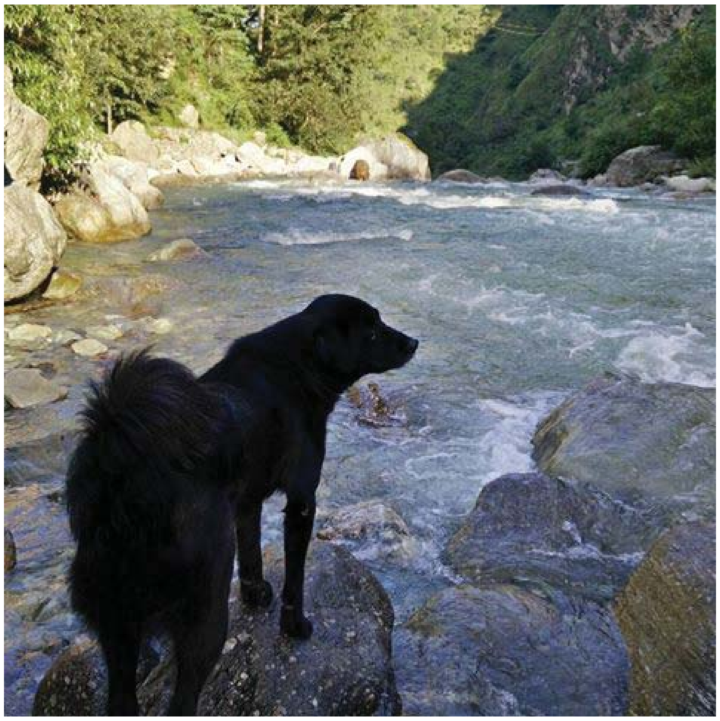
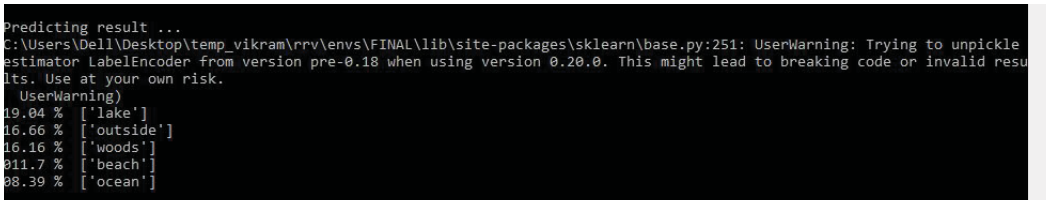


Figure 5.1: Question: Where is this?



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
19.04 %  ['lake']
16.66 %  ['outside']
16.16 %  ['woods']
011.7 %  ['beach']
08.39 %  ['ocean']
```

Figure 5.2: Predicted Answer: Lake

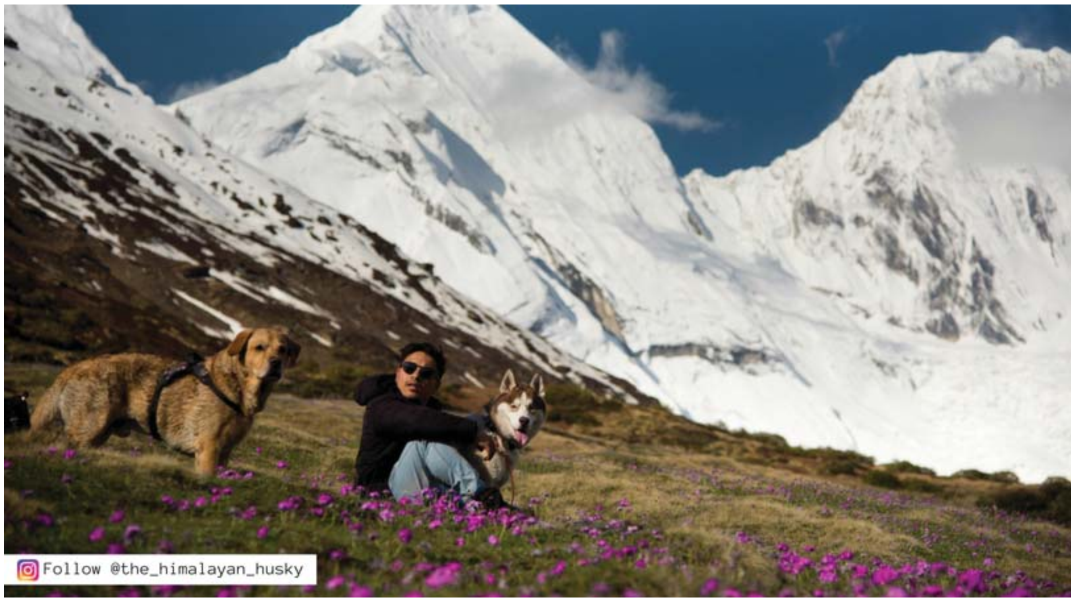
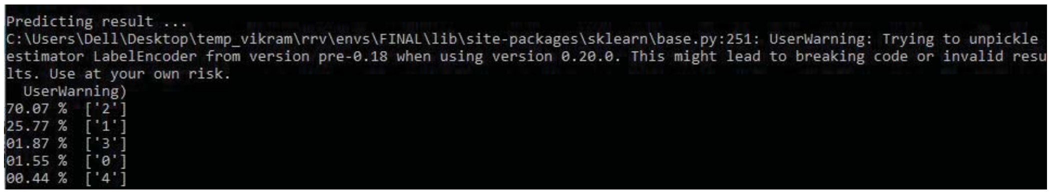
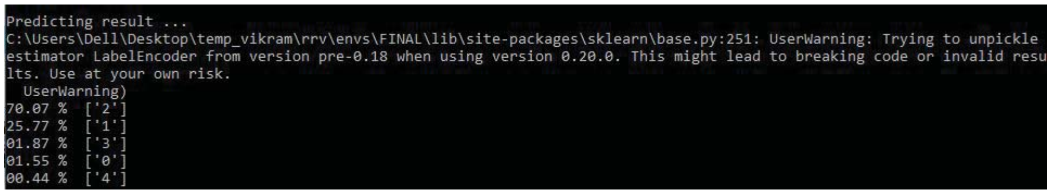


Figure 5.3: Question: How many dogs?



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
70.07 %  ['2']
25.77 %  ['1']
01.87 %  ['3']
01.55 %  ['0']
00.44 %  ['4']
```

Figure 5.4: Predicted Answer: 2



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
70.07 %  ['2']
25.77 %  ['1']
01.87 %  ['3']
01.55 %  ['0']
00.44 %  ['4']
```

Figure 5.5: Question: What is behind them? Predicted Answer: snow

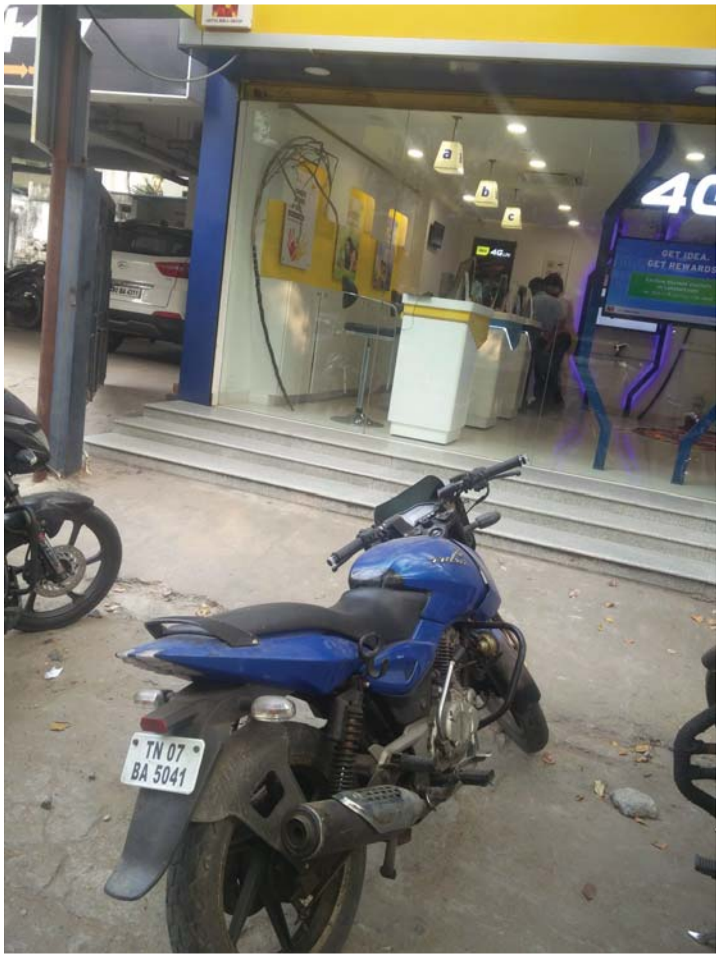
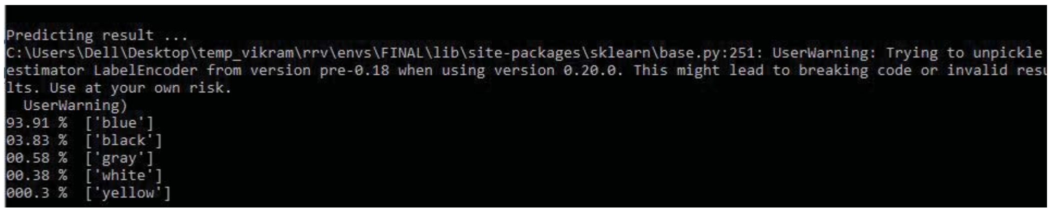
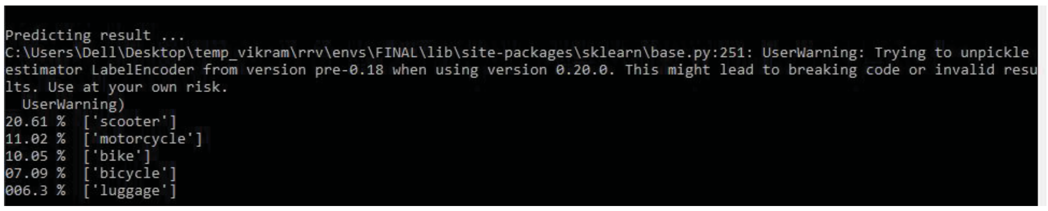


Figure 5.6: Question: What colour is the motorcycle?



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
93.91 %  ['blue']
03.83 %  ['black']
00.58 %  ['gray']
00.38 %  ['white']
000.3 %  ['yellow']
```

Figure 5.7: Predicted Answer: Blue



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
20.61 %  ['scooter']
11.02 %  ['motorcycle']
10.05 %  ['bike']
07.09 %  ['bicycle']
006.3 %  ['luggage']
```

Figure 5.8: Question: What is this? Predicted Answer: Scooter

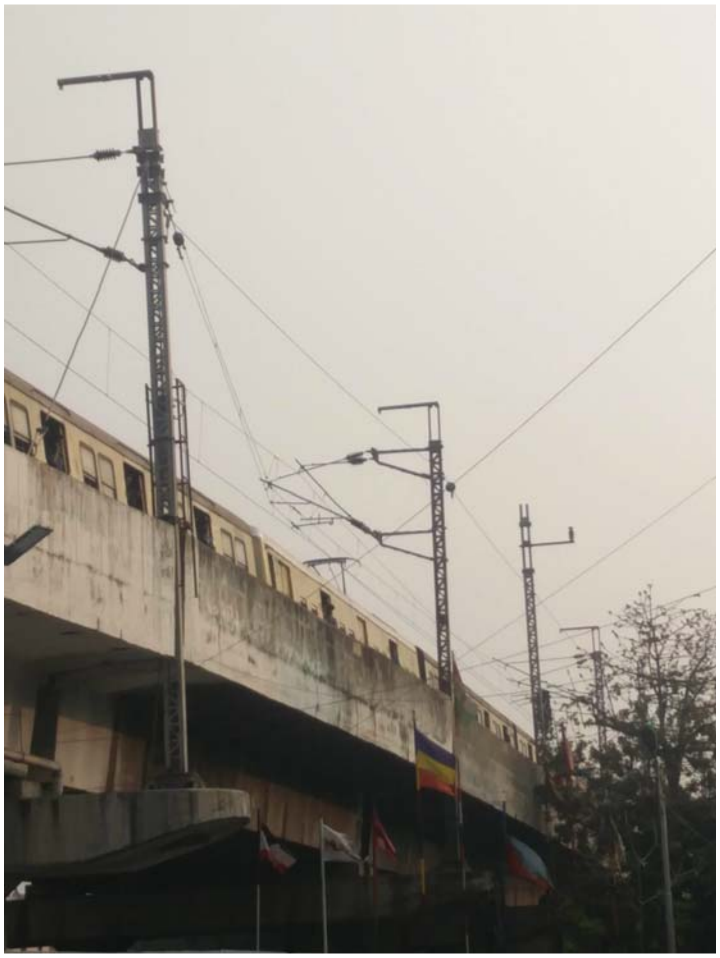
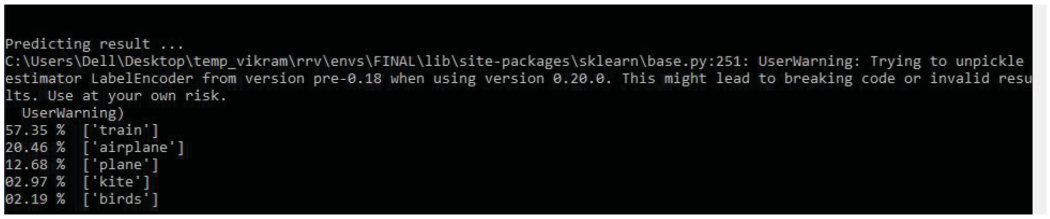
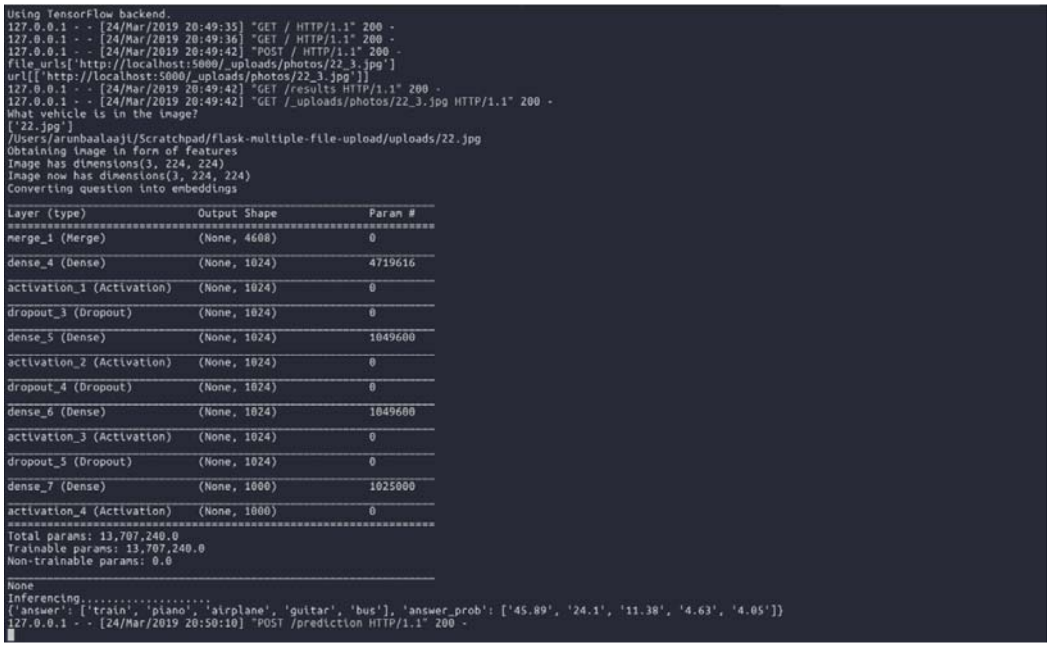


Figure 5.9: Question: What is this?



```
Predicting result ...
C:\Users\Dell\Desktop\temp_vikram\rrv\envs\FINAL\lib\site-packages\sklearn\base.py:251: UserWarning: Trying to unpickle
estimator LabelEncoder from version pre-0.18 when using version 0.20.0. This might lead to breaking code or invalid resu
lts. Use at your own risk.
  UserWarning)
57.35 %  ['train']
20.46 %  ['airplane']
12.68 %  ['plane']
02.97 %  ['kite']
02.19 %  ['birds']
```

Figure 5.10: Predicted Answer: train



```
Using TensorFlow backend.
127.0.0.1 - - [24/Mar/2019 20:49:35] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Mar/2019 20:49:36] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Mar/2019 20:49:42] "POST / HTTP/1.1" 200 -
file_urls['http://localhost:5000/_uploads/photos/22_3.jpg']
url[['http://localhost:5000/_uploads/photos/22_3.jpg']]
127.0.0.1 - - [24/Mar/2019 20:49:42] "GET /results HTTP/1.1" 200 -
127.0.0.1 - - [24/Mar/2019 20:49:42] "GET /_uploads/photos/22_3.jpg HTTP/1.1" 200 -
What vehicle is in the image?
['22.jpg']
/Users/arunbaalaaji/Scratchpad/flask-multiple-file-upload/uploads/22.jpg
Obtaining image in form of features
Image has dimensions(3, 224, 224)
Image now has dimensions(3, 224, 224)
Converting question into embeddings
____________________________________________________________________
Layer (type)                 Output Shape              Param #
====================================================================
merge_1 (Merge)              (None, 4608)              0
____________________________________________________________________
dense_4 (Dense)              (None, 1024)              4719616
____________________________________________________________________
activation_1 (Activation)    (None, 1024)              0
____________________________________________________________________
dropout_3 (Dropout)          (None, 1024)              0
____________________________________________________________________
dense_5 (Dense)              (None, 1024)              1049600
____________________________________________________________________
activation_2 (Activation)    (None, 1024)              0
____________________________________________________________________
dropout_4 (Dropout)          (None, 1024)              0
____________________________________________________________________
dense_6 (Dense)              (None, 1024)              1049600
____________________________________________________________________
activation_3 (Activation)    (None, 1024)              0
____________________________________________________________________
dropout_5 (Dropout)          (None, 1024)              0
____________________________________________________________________
dense_7 (Dense)              (None, 1000)              1025000
____________________________________________________________________
activation_4 (Activation)    (None, 1000)              0
====================================================================
Total params: 13,707,240.0
Trainable params: 13,707,240.0
Non-trainable params: 0.0
____________________________________________________________________
None
Inferencing....................
{'answer': ['train', 'piano', 'airplane', 'guitar', 'bus'], 'answer_prob': ['45.89', '24.1', '11.38', '4.63', '4.05']}
127.0.0.1 - - [24/Mar/2019 20:50:10] "POST /prediction HTTP/1.1" 200 -
```

Figure 5.11: Model Architecture

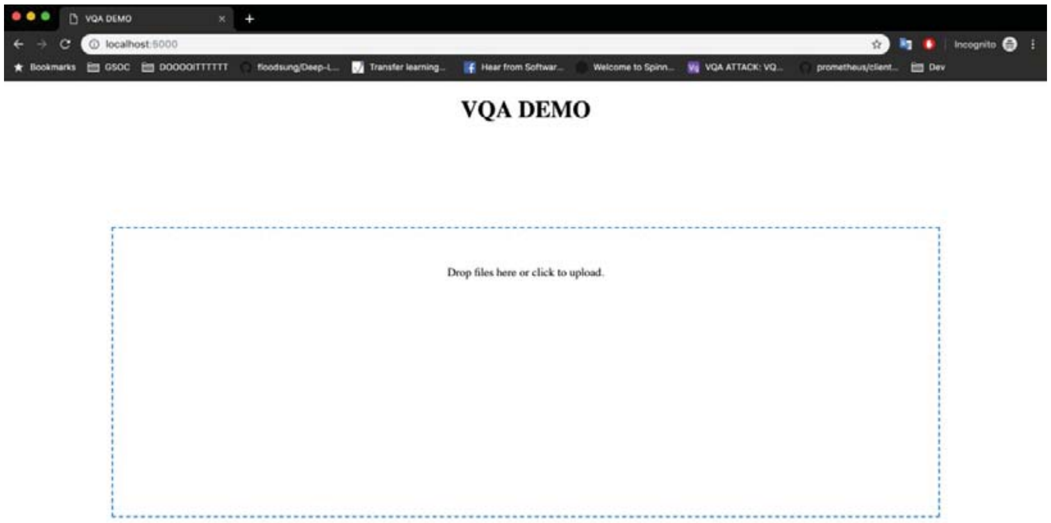


Figure 5.12: Flask based framework for VQA system

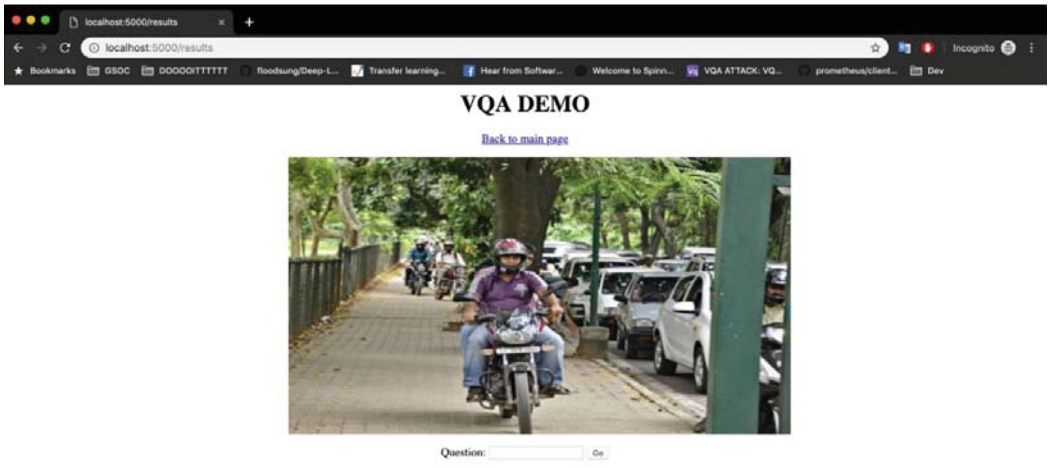
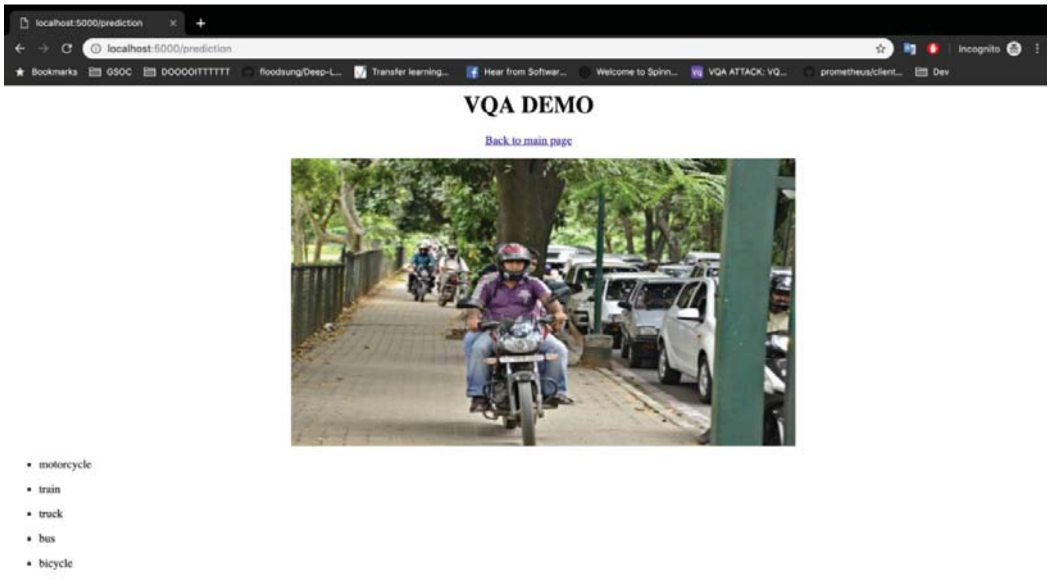


Figure 5.13: Taking Questions as input from the User



- motorcycle
- train
- truck
- bus
- bicycle

Figure 5.14: Final Output for the VQA System

5.2 Suggestions and Recommendations

By further fine-tuning the hyper-parameters like the learning rate, number of epochs for training, optimizer type, batch size etc., the architecture can be polished even further to generate even more accurate and precise results than the existing architectures.

Extensive Computational power is needed to run, train and evaluate the scripts due to the heavy architecture design. Thus, it is strongly recommended to facilitate these programs using powerful Graphical processing Units or cloud platforms like Google Cloud Platform, Amazon Web Services, Microsoft Azure etc.

5.3 Conclusion

The field of VQA has grown by leaps and bounds and is still growing at a fantabulous rate, despite being a new domain of research and development in Machine learning and Computer Vision. We analyzed some of the prominent datasets and existing architectures and significant improvements in performance continue to be seen that portrays that there is an enormous amount of work still going on and there are still various chances for further innovation in this research domain. We were able to successfully design and construct a VQA Architecture by concatenation of the fine-tuned VGG16 arrchitecture for performing the image feature extraction while using GloVe word embeddings for the language understanding part to provide accurate and precise results with suitable accuracy and precision.

5.4 Further Research

The numerous VQA based Datasets, Algorithms and models continue to keep on receiving widespread acclaim and showcase optimized results. Upon, in-depth qualitative analysis of the above mentioned datasets and some of the advanced scale VQA algorithms and approaches, certain specialities as well as their draw backs related to structured data formation, preprocessing and augmentation technicalities were mentioned and need further improvement in this sphere in order to maximize the accuracy and generate better and efficient VQA based systems. More research is needed to counter the adverse effects of Improper Question-Answer biasing, improper instance segmentation etc. and more optimized evaluation metrics and benchmarks need to be developed for better supervision of such systems.

REFERENCES

[1] Xinxin Zhu, Lixiang Li, Jing liu, Ziyi Li, Haipeng Peng, Xinxin Niu, "Image Captioning with Triple-Attention and Stack Parallel LSTM", Journal of Neurocomputing (Elsevier), Volume 319, 30 November, Pages 55-65.

[2] Neha Sharma, Vibhor Jain, Anju Mishra, "An Analysis of Convolutional Neural Networks For Image Classification", Journal of Procedia Computer Science (Elsevier), Volume 132, Pages 377-384,DOI: https://doi.org/10.1016/j.procs.2018.05.198.

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, "Rethinking the Inception Architecture for Computer Vision",CVPR-2015, arXiv:1512.00567 [cs.CV].

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition", CVPR-2015, arXiv:1512.03385 [cs.CV].

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, "VQA: Visual Question Answering", The IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2425-2433.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona et. al. , "Microsoft COCO: Common Objects in Context",Computer Vision ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham.

[7] Peng Zhang et. al., "Yin and Yang: Balancing and Answering Binary Visual Questions", 2016, arXiv:1511.05099.

[8] Li, Yi Yang, Jianyu Wang, Wei Xu, "Zero-Shot Transfer VQA Dataset", arXiv:1811.00692 [cs.AI].

[9] Nils Reimers, Iryna Gurevych,"Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks",2017, arXiv:1707.06799v2 [cs.CL].

[10] Wei Dong., Richard Socher., Li-Jia Li., Kai Li., Li Fei-Fei., "ImageNet: A large-scale hierarchical image database" , IEEE Conference on Computer Vision and Pattern recognition 2009, DOI: 10.1109/CVPR.2009.5206848.

[11] Everingham, M., Van Gool, L., Williams, C.K.I. et al., "PASCAL Visual Object Classes (VOC) Challenge", International Journal of Computer Vision (2010) 88: 303,DOI: https://doi.org/10.1007/s11263-009-0275-4.

[12] Jianxiong Xiao, James Hays, Krista A. Ehinger et. al., "SUN database: Large-scale scene recognition from abbey to zoo", IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2010, DOI:10.1109/CVPR.2010.5539970.

[13] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, D. Parikh, "Making the V in VQA matter: Elevating the role of image understanding in visual question answering.", arXiv preprint arXiv:1612.00837, 2016.

[14] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, R. Girshick., "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning", arXiv preprint arXiv:1612.06890, 2016.

[15] Krishna R., Zhu. Y., Groth, O. et al. ,"Visual genome: connecting language and vision using crowd sourced dense image annotations", International Journal of Computer Vision (2017) 123:32., DOI: https://doi.org/10.1007/s11263-016-0981-7.

APPENDIX A

SOURCE CODES

```
//VQA.py - VQA ARCHITECTURE

import numpy as np
import spacy
import keras.backend as backend
from keras.optimizers import SGD
from sklearn.externals import joblib
from keras.layers.convolutional import Conv2D as c2d
from keras.layers.convolutional import MaxPooling2D as m2d,
   ZeroPadding2D as z2d

from keras.models import Sequential
from keras.layers.core import Flatten, Dense, Dropout,
   Reshape, Activation, Dropout
from keras.layers import LSTM, Merge, Dense




def remove_layer(model):
   model.layers.pop()
   if not model.layers:
      model.inbound_nodes = []
      model.outputs = []
      model.outbound_nodes = []
   else:
      model.layers[-1].outbound_nodes = []
      model.outputs = [model.layers[-1].output]
   model.built = False
```

```
    return model


def VGG_16(weights):
    weight_dict = h5py.File(weights, 'r')
    model = Sequential()
    model.add(z2d((1,1),input_shape=(3,224,224)))
    model.add(c2d(64,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(64,( 3, 3), activation='relu'))
    model.add(m2d((2,2), strides=(2,2)))

    model.add(z2d((1,1)))
    model.add(c2d(128,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(128,( 3, 3), activation='relu'))
    model.add(m2d((2,2), strides=(2,2)))

    model.add(z2d((1,1)))
    model.add(c2d(256,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(256,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(256,( 3, 3), activation='relu'))
    model.add(m2d((2,2), strides=(2,2)))

    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(m2d((2,2), strides=(2,2)))
    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(z2d((1,1)))
    model.add(c2d(512,( 3, 3), activation='relu'))
    model.add(m2d((2,2), strides=(2,2)))

    model.add(Flatten())
    model.add(Dense(4096, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(4096, activation='relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1000, activation='softmax'))

    flattened = model.layers

    naive_bayes_layer = weight_dict.attrs['nb_layers']

    for layer in range(naive_bayes_layer):
       val = weight_dict['layer_{}'.format(layer)]
       w = [val['param_{}'.format(param)] for param in
          range(val.attrs['nb_params'])]
       if not w : continue
       if len(w[0].shape) >2:
          w[0] = np.swapaxes(w[0],0,3)
          w[0] = np.swapaxes(w[0],0,2)
          w[0] = np.swapaxes(w[0],1,2)
       flattened[layer].set_weights(w)
    # remove last tow layers to match dimensions
    model = remove_layer(model)
    model = remove_layer(model)
    weight_dict.close()
    return model
```

```
def set_params():
   backend.set_image_data_format('channels_first')
# sets image to be read as (depth, input_depth, rows, cols)
   backend.set_image_dim_ordering('th')


def VQA():
   image_size = 4096
   word_size = 300
   n_lstm= 3
   n_dense = 3
   n_hidden = 1024
   n_hidden_lstm= 512
   activation = 'tanh'
   dropout = 0.5
   max_word = 30 # max words in question




   #Image layer

   image_model= Sequential()
   image_model.add(Reshape((image_size,),
      input_shape=(image_size,)))
   #LSTM layer
   language_model = Sequential()
   language_model.add(LSTM(n_hidden_lstm,
      return_sequences=True, input_shape=(max_word,
      word_size)))
   language_model.add(LSTM(n_hidden_lstm,
      return_sequences=True))
```

```
    language_model.add(LSTM(n_hidden_lstm,
        return_sequences=False))

    #combine model
    model = Sequential()
    model.add(Merge([language_model, image_model],
        mode='concat', concat_axis=1))

    for _ in range(n_dense):
        model.add(Dense(n_hidden, kernel_initializer='uniform'))
        model.add(Activation(activation))
        model.add(Dropout(dropout))
    model.add(Dense(1000))
    #Final layer with Top 1000 answers
    model.add(Activation('softmax'))
    print (model.summary())
    return model


def main(image_path=None, ques=None):
    set_params()
    warnings.filterwarnings("ignore",
        category=DeprecationWarning)
    print("Obtaining image in form of features")
    image_resized = cv2.resize(cv2.imread(image_path), (224,
        224))
    image_float = image_resized.astype(np.float32)
    normalized_vector = [103.939, 116.779, 123.68]
    for third_dimension in range(3):
        image_float[:, :, third_dimension] = image_float[:, :,
            third_dimension] - normalized_vector[third_dimension]
    # convert from width,height,channel to channel,width,height
    image_float = image_float.transpose((2,0,1))
    print("Image has dimensions"+ str(image_float.shape))
```

```
image = np.expand_dims(image_float, axis=0)
# Adding extra dimension for maintaining model size
print("Image now has dimensions"+ str(image_float.shape))
im_features = np.zeros((1, 4096))
model_image = VGG_16('/Users/arunbaalaaji/Scratchpad/
flask-multiple-file-upload/VQA_Attack/Weights/vgg16_dict.h5')
stochastic_gd= SGD(lr=0.1, decay=1e-6, momentum=0.9,
   nesterov=True)
model_image.compile(optimizer=stochastic_gd,
   loss='categorical_crossentropy')
im_features[0,:] = model_image.predict(image)[0]
print("Converting question into embeddings")
embedding = spacy.load('en_vectors_web_lg')
word_as_tokens = embedding(str(ques))
tensor_q = np.zeros((1, 30, 300))
# 30 because max is 30 words in a question
for x in range(len(word_as_tokens)):
   tensor_q[0,x,:] = word_as_tokens[x].vector
vqa = VQA()
vqa.load_weights('/Users/arunbaalaaji/Scratchpad/
flask-multiple-file-upload/VQA_Attack/Weights/VQA_weights.hdf5')
vqa.compile(loss='categorical_crossentropy',
   optimizer='rmsprop')
print("Inferencing....................")
output = vqa.predict([tensor_q, im_features])
# Answer will be a top 1000 answer vector
# Answers are encoded as labels to minimize compute as
   words increase compute
encoder = joblib.load('/Users/arunbaalaaji/Scratchpad/
flask-multiple-file-upload/VQA_Attack/Weights/labelencoder.pkl')
answer_vector = {}
answer_vector['answer']= []
answer_vector['answer_prob']= []
temp_list = []
```

```
    for label in reversed(np.argsort(output)[0,-5:]):
        temp_list.append(label)
        answer_vector['answer'].append(str
        (encoder.inverse_transform(temp_list)[0]))
        answer_vector['answer_prob'].append(str
        (round(output[0,label]*100, 2)))
        temp_list.pop()
    print (answer_vector)
    return answer_vector


if _name_ == "_main_":
    main('test.jpg', "What vehicle is in the picture?")
\
```

```
//Flask code:

from flask import Flask
from flask import redirect
from flask_uploads import configure_uploads
from flask_dropzone import Dropzone
from flask_uploads import UploadSet
from flask_uploads import patch_request_class
from wtforms import StringField
from flask_uploads import IMAGES
from VQA_Attack.demo import main
from flask import render_template
from flask import request
from flask import session, url_for
from flask_wtf import FlaskForm

url = []
file_path = []
```

```
import os

class Question(FlaskForm):
    question = StringField('Question')



app = Flask(_name_)
dropzone = Dropzone(app)
app.config['DROPZONE_ALLOWED_FILE_TYPE'] = 'image/*'

app.config['DROPZONE_UPLOAD_MULTIPLE'] = True

app.config['DROPZONE_ALLOWED_FILE_CUSTOM'] = True

app.config['SECRET_KEY'] = 'vqademo'

app.config['DROPZONE_REDIRECT_VIEW'] = 'results'

app.config['UPLOADED_PHOTOS_DEST'] = os.getcwd() + '/uploads'

photos = UploadSet('photos', IMAGES)
configure_uploads(app, photos)
patch_request_class(app)



@app.route('/', methods=['GET', 'POST'])
def index():

    if "file_urls" not in session:
        session['file_urls'] = []
    file_urls = session['file_urls']

    if request.method == 'POST':
        file_obj = request.files
        for f in file_obj:
            file = request.files.get(f)

            filename = photos.save(
                file,
                name=file.filename
            )

            file_path.append(filename)
            file_urls.append(photos.url(filename))

        session['file_urls'] = file_urls
        return "uploading..."
    return render_template('index.html')


@app.route('/results', methods=['POST', 'GET'])
def results():

    if "file_urls" not in session or session['file_urls'] == []:
        return redirect(url_for('index'))

    file_urls = session['file_urls']
    url.append(session['file_urls'])
    print('file_urls'+ str(file_urls))
    print('url'+ str(url))
    session.pop('file_urls', None)
    return render_template('results.html', file_urls=file_urls)

@app.route('/prediction', methods=['POST'])
def question():
   if request.method == 'POST':
        question = request.form.get('question')
        print(question)
        if question is not None:
            urls = url.pop()
            print (urls)
            print (file_path)
            print (os.getcwd()+'/uploads/'+file_path[0])
            fileloc = (os.getcwd()+'/uploads/'+file_path[0])
            output=main(fileloc, question)
            file_path.pop()
            return render_template('prediction.html',
                file_urls=urls,accuracies=output['answer'],
                lables=output['answer_prob'])
```

Sample

ORIGINALITY REPORT

6%	2%	4%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, Devi Parikh. "Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering", 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017 Publication	1%
2	Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions", 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. Publication	<1%
3	Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, Zbigniew Wojna. "Rethinking the Inception Architecture for Computer Vision", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016 Publication	<1%

4	Submitted to Cranfield University Student Paper	<1%
5	www.csauthors.net Internet Source	<1%
6	Submitted to Rochester Institute of Technology Student Paper	<1%
7	Submitted to Hong Kong Baptist University Student Paper	<1%
8	mdpi.com Internet Source	<1%
9	Submitted to National Institute of Technology, Kurukshetra Student Paper	<1%
10	research.googleblog.com Internet Source	<1%
11	Ting Xiao, Lei Liu, Kai Li, Wenjian Qin, Shaode Yu, Zhicheng Li. "Comparison of Transferred Deep Neural Networks in Ultrasonic Breast Masses Discrimination", BioMed Research International, 2018 Publication	<1%
12	Neha Sharma, Vibhor Jain, Anju Mishra. "An Analysis Of Convolutional Neural Networks For Image Classification", Procedia Computer Science, 2018	<1%

Publication

13	arxiv.org Internet Source	<1%
14	"Computer Vision – ECCV 2018 Workshops", Springer Nature, 2019 Publication	<1%
15	Kushal Kafle, Christopher Kanan. "Visual question answering: Datasets, algorithms, and future challenges", Computer Vision and Image Understanding, 2017 Publication	<1%
16	Submitted to UC, Boulder Student Paper	<1%
17	stackoverflow.com Internet Source	<1%
18	Xinxin Zhu, Lixiang Li, Jing Liu, Ziyi Li, Haipeng Peng, Xinxin Niu. "Image captioning with Triple-Attention and Stack Parallel LSTM", Neurocomputing, 2018 Publication	<1%
19	Yash Goyal, Tejas Khot, Aishwarya Agrawal, Douglas Summers-Stay, Dhruv Batra, Devi Parikh. "Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering", International Journal of Computer Vision, 2018	<1%

Publication

20	pybit.es Internet Source	<1%
21	"Advances in Multimedia Information Processing – PCM 2018", Springer Nature America, Inc, 2018 Publication	<1%
22	Submitted to De Montfort University Student Paper	<1%
23	rezzyekocaraka.com Internet Source	<1%
24	Spandana Gella, Frank Keller, Mirella Lapata. "Disambiguating Visual Verbs", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018 Publication	<1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		

Analysis of Deep Convolutional Neural Networks for Image Classification

Abhijit Sridhar
Student
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India
abhijitsridhar_sri@srmuniv.edu.in

S. Arun Baalaaji
Student
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India
arunbaalaaji_msa@srmuniv.edu.in

Prateek Ralhan
Student
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India
prateekralhan_ash@ srmuniv.edu.in

R. Ravindra Vikram
Student
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India
ravindravikram_rav@ srmuniv.edu.in

A. Ruhan Bevi
Associate Professor
Department of Electronics and Communication Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India
ruhan.b@ktr.srmuniv.ac.in

Abstract— **Deep Neural Networks are an important factor in developing viable Deep Learning and Computer Vision algorithms. Due to their architecture, they excel in discriminating between various classes of data, and thus perform well in tasks such as image classification, or object detection. Various datasets have been created to evaluate the performances of these networks and their applications for example, ImageNet for classification, and MS-COCO for image captioning.**
This paper aims at comparing the performance of networks such as VGG16 and 19, ResNet, and InceptionV3 on the CIFAR10 dataset and determining the model better suited for classification.

Keywords—convolutional neural networks, max pooling, computer vision.

I. Introduction

Convolutional Neural Networks [1] are one of the fundamental blocks of deep learning and computer vision. Their architecture and design makes them extremely suited to handle image data [2].
It’s primitive skeleton consists of a convolutional layer, a max pooling layer, followed by a softmax or output layer [3]. The convolutional layer applies a function to a certain window of pixels defined by the kernel of the layer. The max pooling layer helps in down sampling, or reducing the dimensionality of the preceding convolutional layer. These two layers can be repeated multiple times, followed by a output layer which performs the classification [7]. Various models can be built by varying the parameters of each layer as well as the number of layers.
However, for image processing it was found that networks with a greater number of layers i.e, “deeper networks”, performed better [9]. Hence the research into deep convolutional neural networks was started [10]. The goal of this paper is to compare the performances of three such networks: VGG16, VGG19, ResNet, and InceptionV3. All the models were trained and evaluated on the CIFAR10 dataset. It was found that the ResNet model performed the best with a validation accuracy of around 93% while the VGG-16 showed poorest performance with an overall accuracy of around 91%.

II. DATASET

Out of the multiple existing datasets for image classification and object detection, we used the CIFAR 10 dataset for the purpose of our comparison.
Compiled by the Canadian Institute For Advanced Research (CIFAR), it comprises of pictures used to train computer vision algorithms and is a subset of ~ 80 million tiny images dataset by MIT. However, it differs in that all the images are labeled. It consists of 60,000 images of size 32x32 pixels, arranged in 10 classes: cars, airplanes, cats, birds, deer, frogs, ships, dogs, horses, and trucks. Each of these 10 classes has 6000 images attributed to it. 50,000 images make up the training set, and the testing set consists of the remaining 10,000 images.
The advantage of this dataset is the low resolution of the images (32x32). This allows researchers to rapidly iterate on various algorithms to obtain the best results possible. Due to its relatively small size when compared with ImageNet and MS-COCO, it allowed us to quickly compare the performances of various CNN architectures.

III. CONVOLUTIONAL NEURAL NETWORKS

A. VGG

The VGG network is a deep convolutional neural network for object recognition that was designed and trained by Oxford’s Visual Geometry Group. It was awarded the first and second places in ImageNet’s localization and classification challenge in 2014, and set the bar for state of the art performance. Following this, they made their model and weights available online. This allowed for other researches to utilize and build upon this model for their own object detection needs. We selected this model due to it's simple architecture and implementation, and tried to compare its performance with the current state of the art networks. The model came to be as part of an experiment to determine the effects of the depth of the network on its accuracy. It was discovered that a significant improvement could be obtained over previous models by increasing the number of layers to 16 or 19. Based on this hypothesis, they developed two variants of the VGG network; the VGG 16 and VGG 19 [5].

The input to both these network is a fixed 224*224*3 RGB image. The images are preprocess by subtracting the mean RGB value from each pixel, for all the images in the training set. Following this, the picture is subjected to pass through a multiple convolutional layers based stack. After this, five max pooling layers are used to implement spatial pooling. Each layer has a stride of 2, performed over a 2x2 pixel window. Next, three fully connected layers are used, the first two of which have 4096 units. The third layer has 1000 units, to perform the 1000-way ILSVRC classification. Thus, each class has 1000 channels. The last layer is the softmax layer used for the final decision making. All the hidden layers use the ReLu activation function.

These networks differed from the other existing networks in that they used a stack of comparatively small receptive fields,(3 layers of 3x3), instead of a single layer with a larger receptive field, such as 7x7 or 11x11 at the input layer. This receptive field is used throughout the network, and is convolved with the input at every pixel. The main advantages of the small receptive field are that it allows for three non-linear rectification layers, which help in decision making, and that it reduces the number of parameters we are required to deal with. This reduction in parameters acts as a form of regularization.

The VGG16 network obtained top one and top five accuracies of 70.5% and 90% on the ImageNet dataset. We have evaluated the performance of both the VGG 16 and 19 variants on the CIFAR 10 dataset, and compared them to the performance of other similar networks.

B. *ResNet*

When deeper Neural Networks, in deep learning models like Convolutional Neural Networks, start converging, their performance is hindered. As we see in networks with increasing depth, the overall accuracy, at some point tends to stagnate and drop rapidly. The ResNet architecture gives insight on dealing with such issues [2].

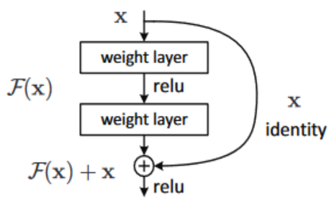


Fig. 1: Residual Learning, Kaiming He, 2015

Opposite to the conventional approach of directly mapping input to output with a function G(x), a residue function H(x) is defined as

$$H(x) = G(x) - x \tag{1}$$

Where H(x) stands for the layers stacked in the deep layered structure [6]. The idea revolves around the fact that G(x) is easier to optimize than the function which needs to be mapped to H(x). This is called *Mapping of Residue*. By optimizing this mapping, the residue is pushed towards a zero value which is easier than fitting H(x) to a particular output value. In other words G(x) = 0 is much simpler and easier to obtain than to perform H(x) = s. This simple concept not only helps improve accuracy, but also decreases the computational cost of the network [8].

Similar to the VGG model, various variants of the ResNet architecture exist, with 18, 50, 101, 152 and more layers. It was observed that the accuracy steadily increased with an increase in the number of layers. However, after a certain number of layers, the performance drops severely due to over fitting. The architecture uses filters of 3x3, with strides equal to 2. A global average pooling method is employed, and similar to the VGG network, the fully connected layer is terminated with a softmax layer for probability estimation. For the purpose of this comparison, we used the ResNet18 model, which still proved to perform better than the other models

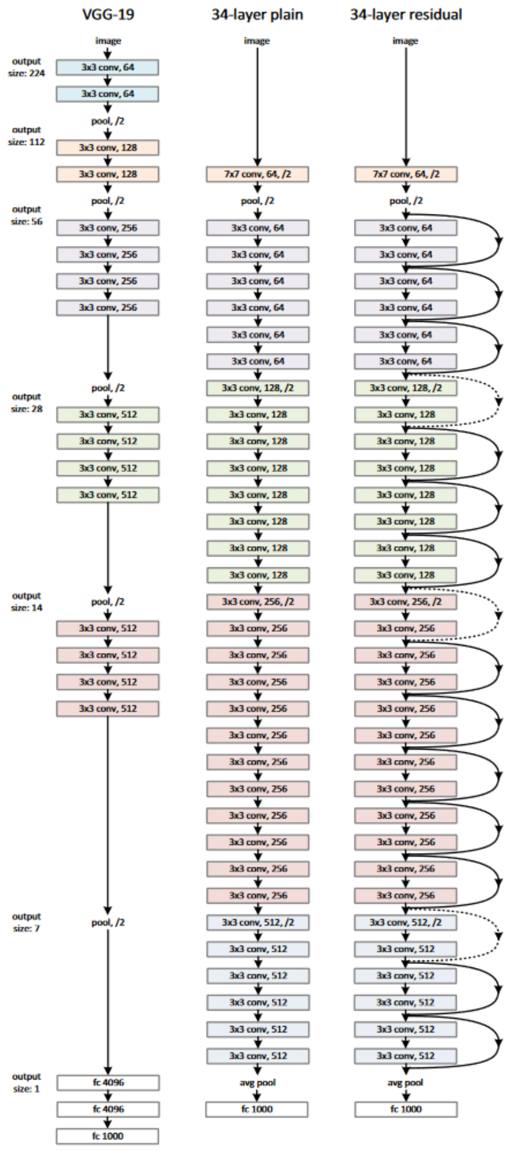


Fig. 2 : Comparison of VGG16 with Residual Networks, Kaiming He, 2015

The ResNet architecture attained a top 5 error rate of 5.71% on the ILSVRC competition, beating all existing architectures at the time. It also converged 28% faster than VGG-16. Fig. 2 shows the differences between the VGG and ResNet architectures.

C. *Inception V3*

The Inception architecture was considered as a significant turning point in the design and development of CNN classifiers. Prior to that, most popular CNNs mainly comprised of deeply stacked convolutional layers with the aim of achieving better performance. The Inception architecture uses a plethora of tricks to improve performance in terms of speed and accuracy, and its perpetual progression lead to the formation of various versions of the same [4].

Due to problems like extremely large size variations in images, selecting the right kernel size for the convolution operation becomes difficult. A bigger kernel size is chosen for data that is distributed more globally, while a smaller kernel is preferred for data that is distributed more locally. Deep networks are immune to overfitting, but it becomes tough to pass gradient updates through the complete network. In order to overcome this issue, a 'naive' inception architecture was proposed by Google that implements convolution on the input, with 3 distinguished filters of sizes 1x1, 3x3, 5x5. Apart from this, max pooling operations are executed. The outputs are concatenated and dispatched to the next inception module. To lower the computational needs, the number of input channels is restricted by appending a surplus 1x1 convolution before the 3x3 and 5x5 convolutions. The resultant network was popularly known as GoogLeNet (Inception v1). Inception V2 was proposed to diminish representational bottleneck, but reducing the dimensions too much may lead to data loss, known as a "representational bottleneck". It also aimed at using better factorization methodologies, like Two 3×3 convolutions replaces one 5×5 convolution as shown in Figure 3.

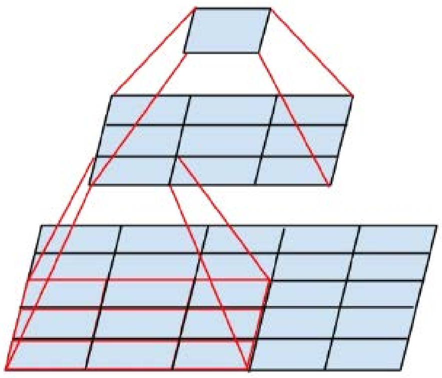


Fig. 3 : Replacing the 5x5 convolution, Christian Szegedy, 2015

An important feature of the InceptionV3 is the auxiliary classifier. They were mainly introduced to improve the convergence of deep networks. Approximately, 320 feature maps are implemented through convolution with stride 2 using efficient grid size reductions along with max pooling method. These 2 feature maps sets are combined together as 640 feature maps and proceed to the following level of the architecture, making it more efficient.

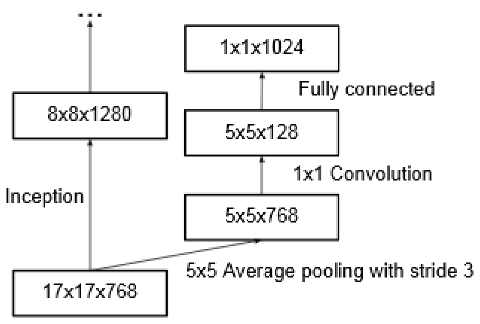


Fig. 4 : Auxiliary classifiers, Christian Szegedy, 2015

Inception Net v3 incorporated all of the above upgrades stated for Inception v2, and in addition used the following: RMSProp Optimizer, Label Smoothing-A type of regularizing component appended to the loss formula that protects the network from becoming too confident about a class), Factorized 7x7 convolutions, and BatchNorm in the Auxiliary Classifiers.

With a depth of 42 layers, the computation cost is only about 2.5 more than that of GoogLeNet, as well as much more profitable than that of VGGNet.

IV. RESULT

All the networks were modeled and tested on the CIFAR10 dataset. The images were first normalized in order to speed up the training process and increase the accuracy. All four networks were put through a similar training process. The images in the training dataset were first cropped to a fixed image size of 32x32 pixels to avoid errors in classification, and then grouped together in batches of 128 and loaded into the system. This batch size allowed for faster training of the network. Smaller batch sizes took too long to train, and larger batch sizes sacrificed accuracy for training speed.

An initial learning rate of 0.1 was chosen which was later changed to 0.01 when the training accuracy plateaued. This caused a jump in both training and validation accuracies, but also gave signs of overfitting i.e, training accuracy reaching 100%. Due to cross entropy loss being suitable to characterize classifiers, it was selected as the loss function. A stochastic gradient descent optimizer function was employed to help decrease the loss. Each model was trained for 200 epochs in a similar fashion and the results were tabulated.

TABLE I. COMPARISON OF ACCURACIES FOR DIFFERENT MODELS

Model	**Accuracy**
VGG 19	91.9
VGG 16	90.8
ResNet18	93.3
InceptionV3	92.3

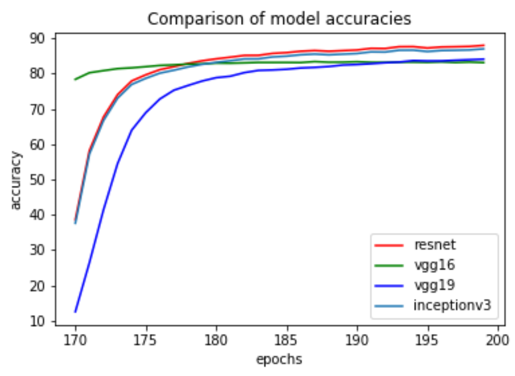
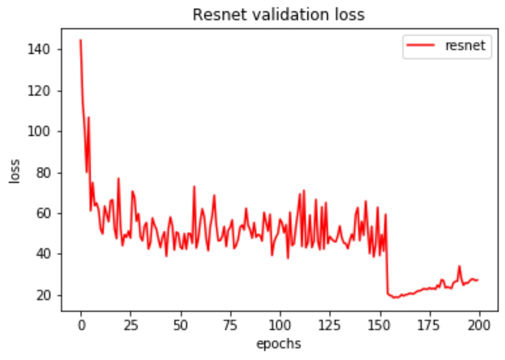
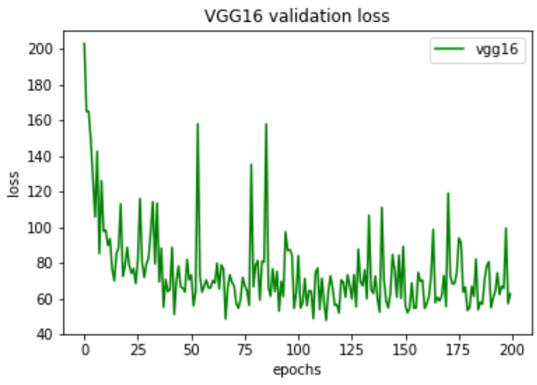
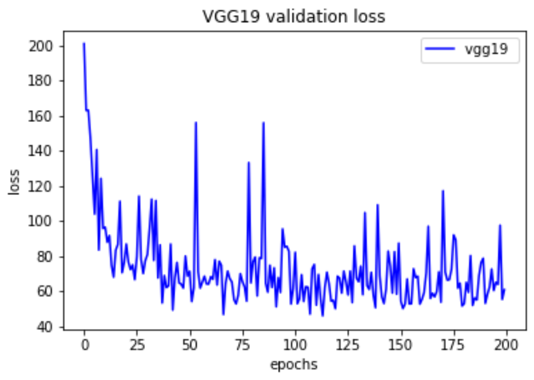


Fig. 5 Numerous Model Accuracies







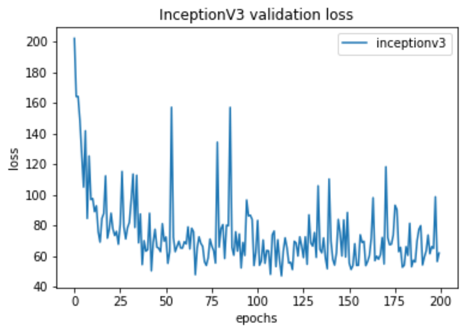


Fig. 6 Different validation losses for numerous models

V. CONCLUSION

The domains of Deep Learning and Computer Vision are relatively new and still have a long way to go. One of the key factors in this is to identify the networks which perform well, or those that are better suited for the task at hand than others. This can be achieved through an understanding of the characteristics of each model, and where it performs best. In our experimentation, we have determined that the ResNet architecture performs the best when compared to the other popular architectures, though not by a large margin. We can conclude that better performance is not achieved just through a deeper network, but also a better handling of the way the network processes the data as seen in the residual functions present in ResNet networks.

REFERENCES

[1] Saad Albawi, tareq Abed Mohammed, Saad-Al-Zawi ;"Understanding of a convolutional neural network"; The International Conference on Engineering and Technology (ICET), 2017, DOI: https://doi.org/10.1109/ICEngTechnol.2017.8308186

[2] Nadia Jmour, Sehla Zayen, Afef Abdelkrim; “Convolutional Neural Networks for image classification”; International Conference on Advanced Systems and Electric Technologies (IC_ASET), 2018, DOI: https://doi.org/10.1109/ASET.2018.8379889

[3] Hoo- Chang Shin et. Al; “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning.” IEEE Transactions on Medical Imaging (Volume: 35, Issue :5, May 2016), DOI: https://doi.org/10.1109/TMI.2016.2528162

[4] Christian Szegedy, Vincent vanhoucke, Sergey loffe, Jonathon Shlens, Zbigniew Wojna; “Rethinking the Inception Architecture for Computer Vision”, arXiv:1512.00567 [cs.CV]R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.

[5] Karen Simonyan, Andrew Zisserman; “Very Deep Convolutional Networks for Large-Scale Image Recognition”, arXiv:1409.1556 [cs.CV]

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; “Deep Residual Learning for Image Recognition”, arXiv:1512.03385 [cs:CV]

[7] Zahra Sadeghi; ”Conceptual Content in Deep Convolutional Neural Networks: An Analysis into multi-faceted properties of neurons”, arXiv:1811.00161 [cs.CV]

[8] Baoqi Li, Yuyao He; “An Improvised ResNet Based on the Adjustable Shortcut Connections”, IEEE Access, DOI: 10.1109/ACCESS.2018.2814605

[9] Hyeong-Ju Kang; “Accelerator-Aware Pruning for Convolutional Neural Networks”, arXiv: 1804.09862

[10] Sebastian Palacio, Joachim Folz et.al; “What do Deep Networks Like to See”, arXiv: 1803.08337

Prateek Ralhan <prateekralhan_ash@srmuniv.edu.in>

Notification for decision for your paper ICASISET19-TRACK8-1026-0316 submitted in ICASISET19

1 message

noreply@chroniclemanager.com <noreply@chroniclemanager.com> Mon, Apr 8, 2019 at 4:23 PM
To: prateekralhan_ash@srmuniv.edu.in

Dear Prateek Ralhan,

We are pleased to inform you that your paper, Analysis Of Deep Convolutional Neural Networks For Image Classification has been accepted for presentation in
INTERNATIONAL CONFERENCE ON ADVANCED SCIENTIFIC INNOVATION IN SCIENCE, ENGINEERING AND TECHNOLOGY 2019.
Please remember to quote the paper id, ICASISET19-TRACK8-1026-0316, whenever inquiring about your submission.

kindly pay your registration fee for Conference on or before 10th April 2019 by using the following links and send registration details to icasiset@gmail.com.

https://coredroidtech.com/payments.php (Link for Payment Gateway)

Comments:

Good Work. This work is accepted.

Icasiset Editor,
Editor
ICASISET19