Object Recognition

Shayeree Sarkar, Baishali Mullick,*
*Department of Electrical and Computer Engineering, Carnegie Mellon University

18752 - Estimation, Detection and Learning
Course Project, Spring 2020

1 Problem Statement

Object recognition is breaking into a wide range of industries, with use cases ranging from personal security to productivity in the workplace including image retrieval, security, surveillance, automated vehicle systems and machine inspection .Significant challenges stay on the field of object recognition including, how to train models that generalize well to real-world settings that have not been seen in training and how to better exploit small-scale training data. While deep learning has shown great success in various tasks with a large amount of labeled data, current techniques generally break down if few labeled examples are available.

2 Train and Test Data Collection and Data Visualisation

We have used the Cifar-10 dataset for the purpose of object recognition and collected the preprocessed images for training from https://www.cs.toronto.edu/~kriz/cifar.html We collect random test data images from google and preprocess it using normalization and data augmentation with the Keras data framework. [1]

3 Methods

3.1 Logistic Regression

Binomial and Multinomial Logistic Regression are popular classifiers used especially on Discriminative Models

We perform the following steps :

1. We load the data set and split it into train, validation and test dataset using masking

2. We preprocess the images for the three split datasets by normalizing the images, and then reshaping them from the original form of **(32,32,3)** to **(3073,1)** for each image after adding a bias unit

*Email: shayeres@andrew.cmu.edu, bmullick@andrew.cmu.edu

[1] The data visulasation using PCA is depicted in the appendix section 4.1

3. For binary logistic regression we form a subset of two classes out of the ten classes, namely, cats and dogs by randomly sampling images for these classes from the train dataset and their corresponding labels

4. We train the weight vector using gradient descent algorithm for **200 epochs** and **learning rate** of **0.0001** and achieve a **training accuracy** of **56 percent** using cost function in 3.2, and update parameters using equation 3.4

5. We use the sigmoid function for the prediction of the labels such that ones **greater than 0.5** are classified as either cats or dogs and we achieve a **test accuracy** of **51 percent**

6. For Multinomial Logistic Regression we train the weight vector using **stochastic gradient descent** algorithm this time, for a subset of 20000 images containing equal samples for each class and train it for **150 epochs** only to achieve achieve a **training accuracy of 80 percent** and **test accuracy of 68 percent** with the **learning rate** of **0.0001** using cost function in 3.2, and update parameters using equation 3.4 [2]

3.2 Convolutional Neural Network

3.2.1 VGG-3 Baseline Model

We have explored CNN design with a **VGG-3 architecture**. The results of the model on the test dataset showed an improvement in classification accuracy with each increase in the depth of the model. Overfitting starts at around 15-to-20 epochs. We have used **Dropouts,L2 Regularization factor**, and **Data Augmentation** of the training images to achieve this accuracy with **learning rate** of **0.001** after **125 epochs** with a Batch size of 64 which has helped in achieving an accuracy of **89 percent** on the training dataset and **88 percent** on test dataset [3]

3.2.2 VGG-3 Baseline Model with SVM

Conventionally, the Softmax function is the classifier used at the last layer of the CNN network. Usage of linear support vector machine (SVM) in the last layer of the CNN instead, often has been proven to give high test accuracy. We have used **Dropouts, regularization factor, and data augmentation** of the training images to achieve this accuracy with **learning rate** of **0.001** after **125 epochs** with a Batch size of 64 and the loss as **Hinge Loss** helping in achieving an accuracy of **85.1 percent** on the **training dataset** and **83.4 percent** on **test dataset**. [4]

3.3 Support Vector Machines using Linear Kernel

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N —the number of features) that distinctly classifies the data points by margin maximization. The plot of accuracy against C for the linear kernel is shown above. The performance of the SVM on both the test set and training set improve substantially until **C 0.1**, after which the performance diminishes. The best **test accuracy** of **28.6 percent** was obtained for **C = 0.1**. We conclude SVM alone is a **poor classifier** for image data. [5]

[2] All graphs and output image data is plotted in Appendix Section 4.2.1
[3] All graphs and output image data is plotted in Appendix Section 4.2.2
[4] All graphs and output image data is plotted in Appendix Section 4.2.3
[5] All graphs and output image data is plotted in Appendix Section 4.2.4

4 Appendix

4.1 Train and Test Data Collection and Data Visualisation

The data visulasation using PCA for CIfar-10 dataset is depicted below:

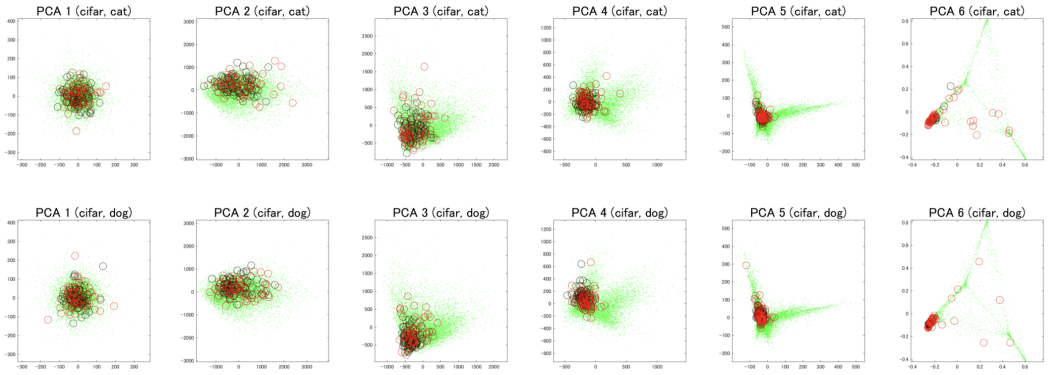

Figure 1: Visualization results for CIFAR-10 data ("cat" and "dog" classes) by usual PCA

4.2 Methods

4.2.1 Logistic Regression

Binomial and Multinomial Logistic Regression are popular classifiers used especially on Discriminative Models They use the following activation function :

$$p\left(y \mid \ x=1, \Theta\right)=\frac{\exp\left(\theta_y x\right)}{\sum_{j=1}^{K} \exp\left(\theta_j x\right)}=\operatorname{softmax}\left(\left(\Theta x\right)_y\right) \tag{4.1}$$

We perform the following steps :

1. We load the data set and split it into train, validation and test dataset using masking

2. We preprocess the images for the three split datasets by normalizing the images, and then reshaping them from the original form of **(32,32,3)** to **(3073,1)** for each image after adding a bias unit

3. For binary logistic regression we form a subset of two classes out of the ten classes, namely, cats and dogs by randomly sampling images for these classes from the train dataset and their corresponding labels

4. We train the weight vector using gradient descent algorithm for **200 epochs** and **learning rate** of **0.0001** and achieve a **training accuracy** of **56 percent** using cost function in 3.2, and update parameters using equation 3.4

$$J(\theta)=-\log p\left(|\mathbf{X}, \theta\right)=\sum_{i=1}^{N}-y^{(i)}\left(\theta^{T} x^{(i)}\right)+\log\left(1+e^{\theta^{T} x^{(i)}}\right) \tag{4.2}$$

5. We use the sigmoid function for the prediction of the labels such that ones **greater than 0.5** are classified as either cats or dogs and we achieve a **test accuracy** of **51 percent**

6. For Multinomial Logistic Regression we train the weight vector using **stochastic gradient descent** algorithm this time, for a subset of 20000 images containing equal samples for each class and train it for **150 epochs** only to achieve achieve a **training accuracy of 80 percent** and **test accuracy of 68 percent** with the **learning rate** of **0.0001** using cost function in 3.2, and update parameters using equation 3.4

$$\frac{\partial J(\theta)}{\partial \theta_j} = -\sum_{i=1}^{N}{}_{j}^{(i)} \left[y^{(i)} - \frac{e^{\theta^{T(i)}}}{1 + e^{\theta^{T(i)}}} \right] \tag{4.3}$$

$$\theta_j \leftarrow \theta_j - \eta \frac{\partial J(\theta)}{\partial \theta_j}, (\eta = learningrate) \tag{4.4}$$

Results for Logistic Regression

The graphs we obtain are shown below:

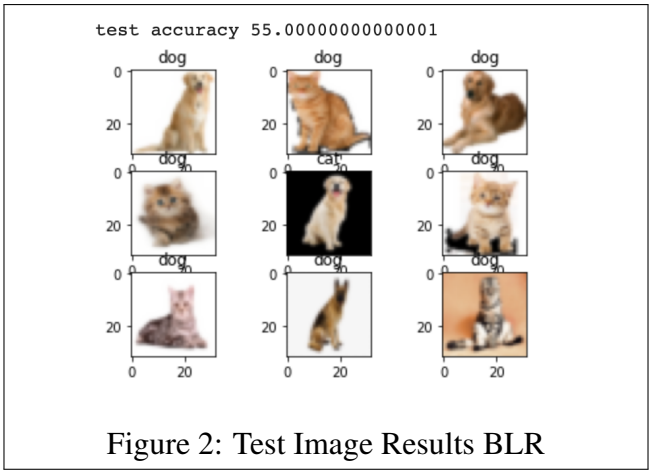

Figure 2: Test Image Results BLR

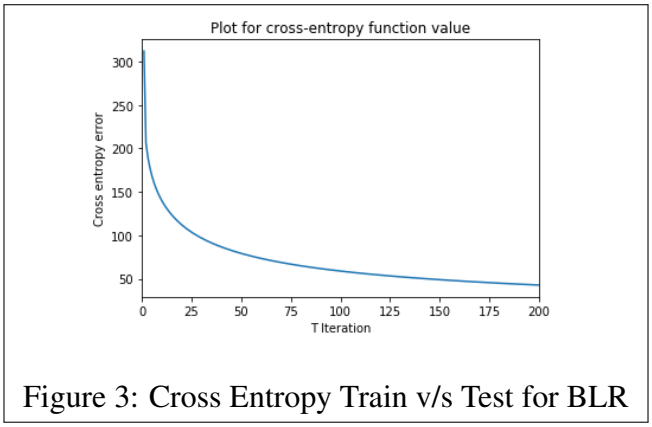

Figure 3: Cross Entropy Train v/s Test for BLR

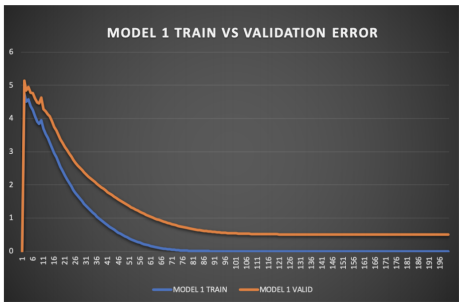

Figure 4: Cross Entropy Train v/s Test for MLR

4.2.2 Convolutional Neural Network

VGG-3 Baseline Model We have explored CNN design with a **VGG-3 architecture**. The results of the model on the test dataset showed an improvement in classification accuracy with each increase in the depth of the model. Overfitting starts at around 15-to-20 epochs.

We have used **Dropouts,L2 Regularization factor**, and **Data Augmentation** of the training images to achieve this accuracy with **learning rate** of **0.001** after **125 epochs** with a Batch size of 64 which has helped in achieving an accuracy of **89 percent** on the training dataset and **88 percent** on test dataset **Results for CNN:**

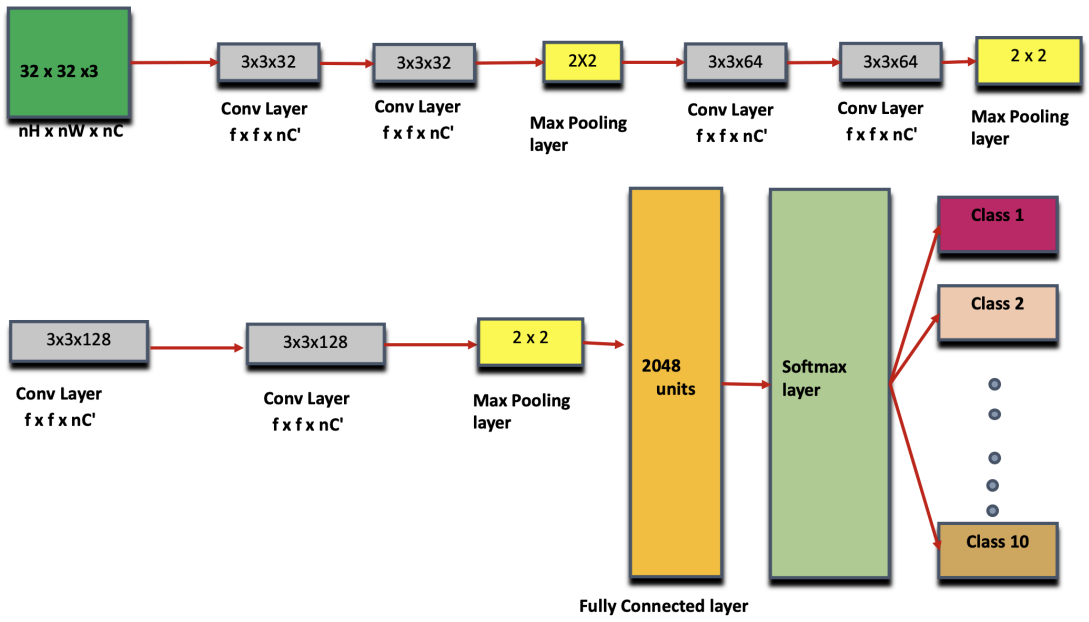

Figure 5: VGG-3 Baseline Model Architecture

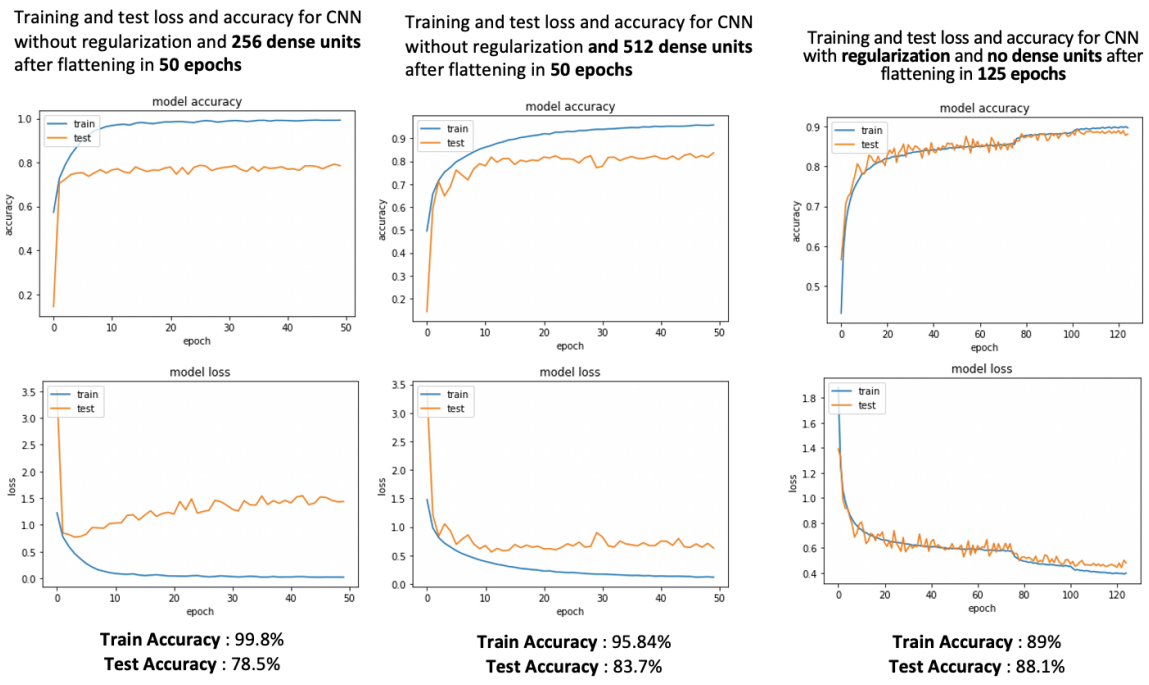

Figure 6: VGG-3 Baseline Model Architecture Model Comparisons

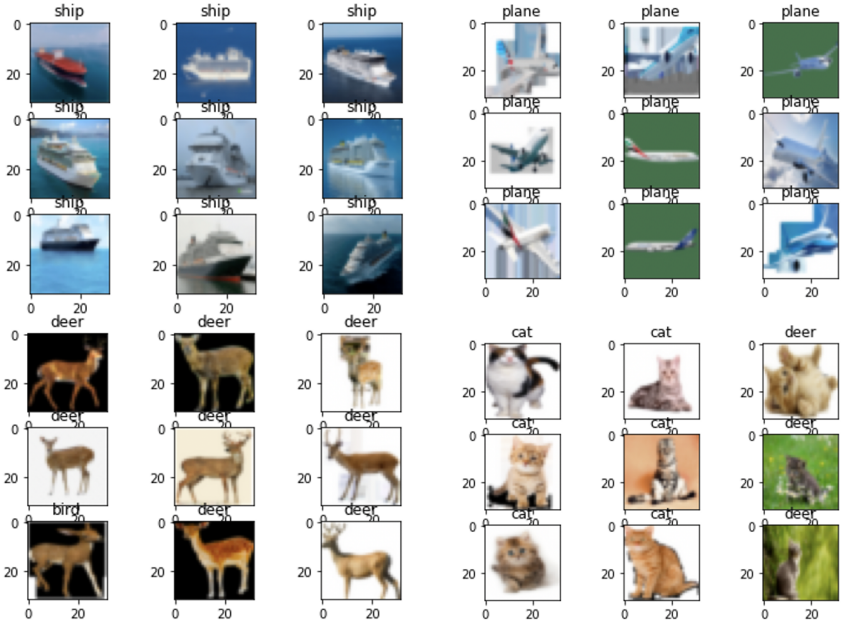

Figure 7: CNN Output with VGG-3 Baseline Model on Test Images

4.2.3 CNN VGG-3 Baseline Model with SVM

Conventionally, the Softmax function is the classifier used at the last layer of the CNN network. Usage of linear support vector machine (SVM) in the last layer of the CNN instead, often has been proven to give high test accuracy.

We have used **Dropouts, regularization factor, and data augmentation** of the training images to achieve this accuracy with **learning rate** of **0.001** after **125 epochs** with a Batch size of 64 and the loss as **Hinge Loss** helping in achieving an accuracy of **85.1 percent** on the **training dataset** and **83.4 percent** on **test dataset**. We can see the Architecture in Fig 8, the loss and accuracy plots in Fig 9 and results on test images in Fig 10.

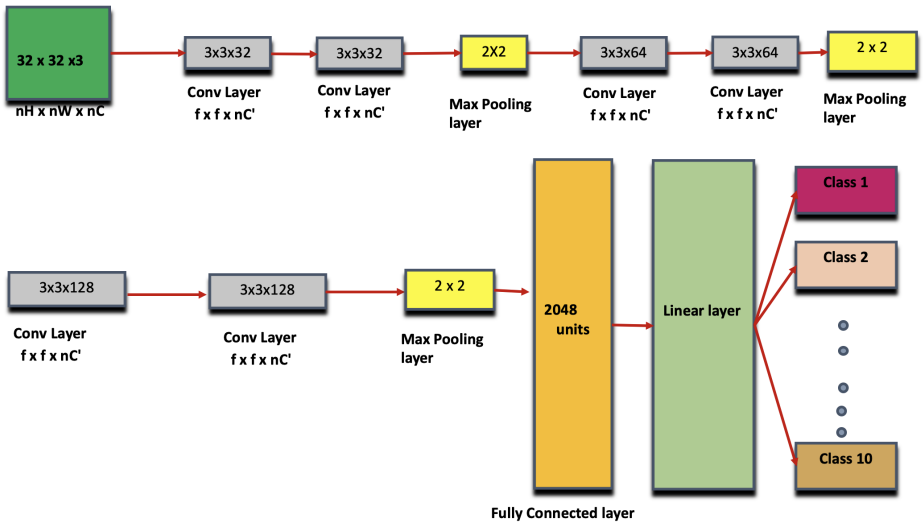

Figure 8: CNN with VGG 3 baseline Model with SVM

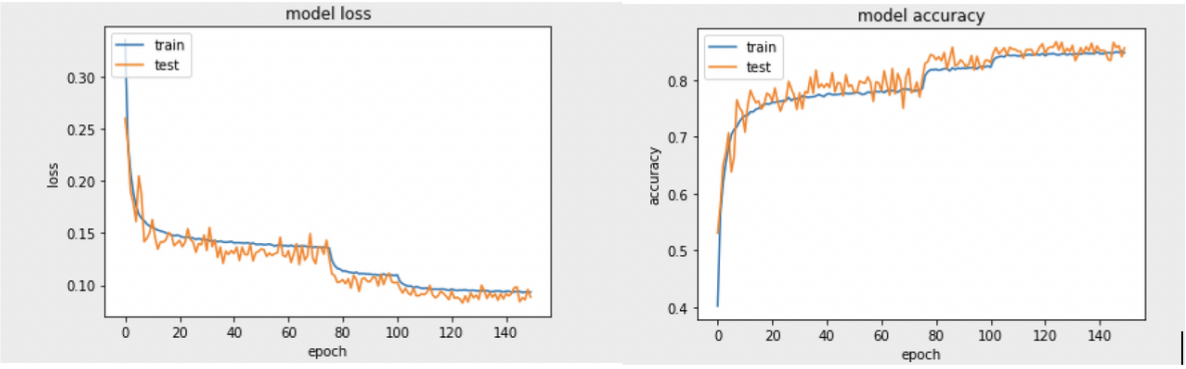

Figure 9: Model accuracy and Loss for Train v/s Test for CNN with SVM

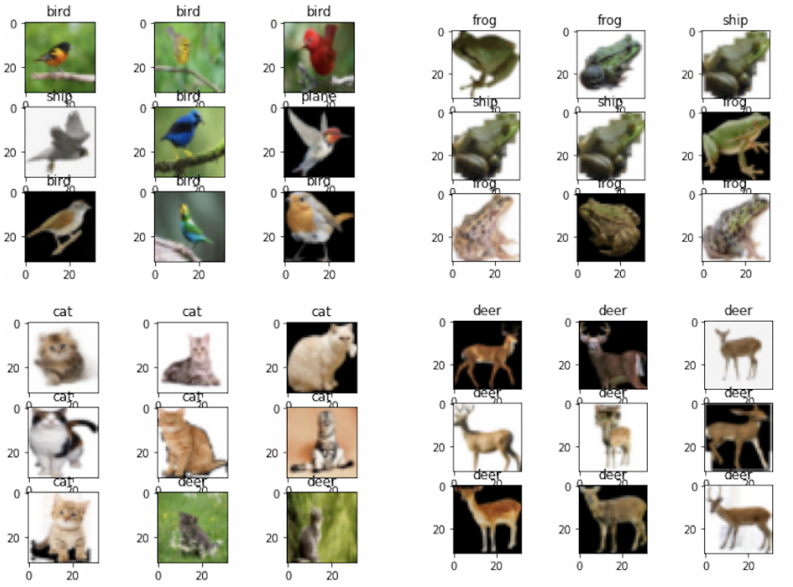

Figure 10: Ouput on Test Images of CNN with VGG 3 baseline Model with SVM

4.2.4 Support Vector Machines using Linear Kernel

The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N —the number of features) that distinctly classifies the data points by margin maximization.

The plot of accuracy against C for the linear kernel is shown above. The performance of the SVM on both the test set and training set improve substantially until **C 0.1**, after which the performance diminishes. The best **test accuracy** of **28.6 percent** was obtained for **C = 0.1**. We conclude SVM alone is a **poor classifier** for image data. Image Classification on the CIFAR-10 Dataset using Support Vector Machines is done using Linear Kernel. The results are shown in Fig 11 and Fig 12.

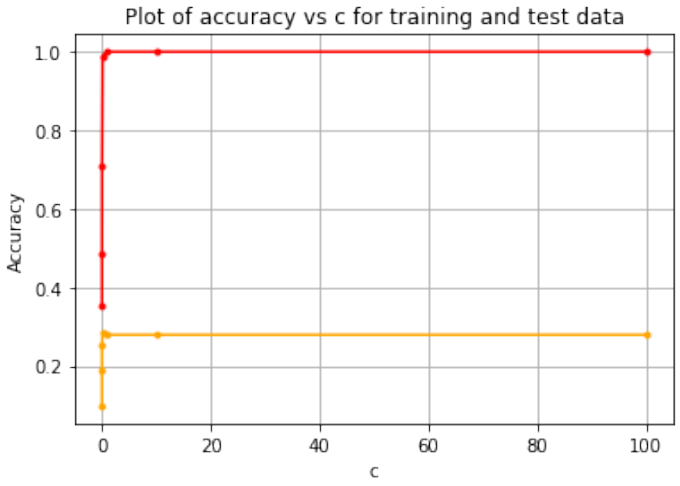

Figure 11: Train v/s Test Accuracy with SVM

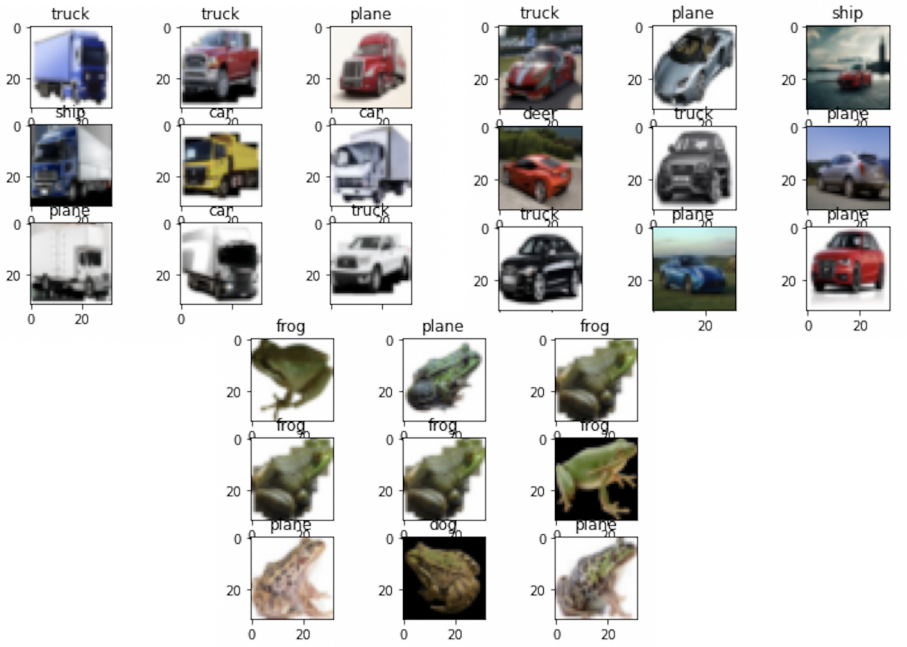

Figure 12: Output for Test Images with SVM

5 Acknowledgments

Shayeree Sarkar and Baishali Mullick acknowledges the help and support of Professor Rohit Negi and the Teaching Assistant of the course 18752 - Zinan Lin