**CNN Based Model for Object Recognition in CIFAR-10**

Dharshini Thangapushparaja, 67617845, MEng Electrical Engineering, UBC Okanagan

***Abstract*—** **The ability to recognize visual objects is crucial in everyday life. This skill is so powerful that we can recognize a face or an object with seemingly little effort, even if its position, scale, stance, or illumination differ. A great number of studies have been conducted in the field of computer vision to develop a human-like object recognition system. Deep neural networks have recently demonstrated tremendous progress in object classification performance, surpassing humans in some cases. In this project, a convolutional neural network (CNN)** **based model uses CIFAR-10, a well-known dataset of natural photographs, to conduct a large-scale experiment to assess human classification accuracy and make an accurate prediction. To achieve high accuracy on picture classification tasks, several function optimization methods such as Baseline, Dropout, Data augmentation, and batch normalization are applied. To avoid overfitting in the project, dropout is used to minimize model complexity and data augmentation to increase the data quality.**

***Keywords*—** Batch Normalization, CIFAR-10, CNN, Data Augmentation, Dropout

# I. INTRODUCTION

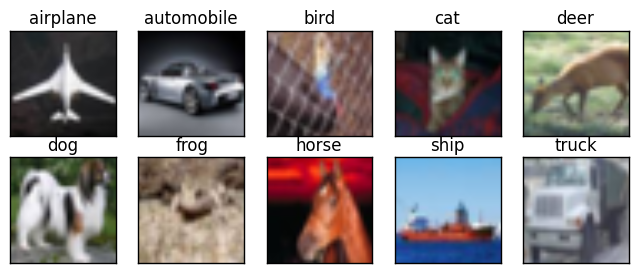
Artificial intelligence has a sub-category called Deep Learning. It is concerned with the development of neural network models capable of learning and making data-driven judgments without the need for human interaction [1]. One of the essential properties of deep learning is feature extraction, which involves extracting data or information from the model's input data. Every image is made up of pixels that are in relation to their neighbors. This phenomenon is used by deep learning approaches, which extract necessary features from a small portion of a larger dataset.

A Convolutional Neural Network (CNN) is made up of many convolutional layers that extract characteristics from input images. Kernels are applied to the image's neighborhood pixels as the input image pixels move through the convolutional layer. The kernel matrix is cross-correlated and then applied to the picture pixels, with the weights in the kernel being changed during training. Much of the data pre-processing duties are removed when CNN is used [2]. The core method is made up of convolutional layers, which are used to extract picture characteristics. Pooling- for dimensionality reduction without sacrificing crucial feature information, ReLU activation function, and classification of various types of pictures into multiple classes.

On the CIFAR-10 dataset, a deep neural network was used to perform picture categorization. To get the best performance out of the network, many hyperparameters must be tweaked. To minimize overfitting, we used dropout to reduce model complexity and data augmentation to enhance sample size to avoid overfitting. The choice of function optimizer is critical since it has a direct impact on convergence speed. In the network, the Adam optimizer was utilized. Except at the output layer, where SoftMax activation is utilized, the ReLU activation function is used throughout the network. The output labels are integers that must be converted to categories using a single hot encoding. Each forward pass during training produces an output value, which is compared to the actual output and a loss function is produced. To optimize the weights, a cross-entropy loss function is utilized with SoftMax activation at the output, and the gradient of the loss function is transmitted back into the network. To achieve the best possible accuracy, this process is repeated for a set number of epochs.

# II. The Data Set

The CIFAR-10 dataset (Canadian Institute For Advanced Research) is a set of photographs used to train machine learning and computer vision algorithms. It’s one of the most used datasets for machine learning studies. The CIFAR-10 dataset includes 60,000 32x32 color images divided into ten categories [3]. Airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks are among the ten diverse classes. The images are separated into two sets: training and test, with each set containing 50000 and 10000 images, respectively. Each class contains 6,000 photos. When it comes to detecting items in photographs, computer algorithms frequently learn by doing. CIFAR-10 is a series of images for teaching a computer to recognize items. Because the images in CIFAR-10 are low-resolution (32x32), researchers can quickly test different algorithms to discover which one works [6]. Keras is a deep learning API written in Python that works on top of the TensorFlow machine learning framework. It was made to facilitate speedy experimentation. The cifar10.load\_data() function in Keras allows you to automatically download standard datasets like CIFAR-10 and save them in the /.keras/datasets directory.



**Fig. 1** CIFAR 10 Dataset sample images

# III. Related Work

Cecilia Xi Yang [4] proposed an Object Recognition in the CIFAR-10 model where a simple CNN wad programmed with 2 convolutional layers to get 70.99% and a larger CNN with 3 convolutional layers was programmed to get 77.68%, proving that the accuracy of larger CNN was 10 times better than the simple CNN model. The difference that was made between them was increasing batch size and reducing the dropout layer from 50% to 20%.

Kruithof, et al. [5] have experimented with a Caffe-based network on 1000 ImageNet clad and performed additional trails with the Keras on the CIFAR10 dataset to validate the applicability and showed that if more target data is used, the accuracy increased. They have also studied the effect of copying all the layers and fine-tuning them and the size of the target dataset. They have concluded that the learning time can be reduced by freezing many layers in a network.

R. Doon, et al [7] have created deep learning models such as image classification and object detections using convolutional neural network . In this study, the CIFAR-10 dataset is employed, and several function optimization strategies such as Adam and RMS, as well as other regularization techniques, are applied to achieve good accuracy on the image classification job. The proposed model has reduced overfitting by reducing the model into 2: validation and training set.

Wenqing Yang and Harvey Han [8] have described the architecture of different models such as SoftMax Regression, SVM, CNN and Spatially-sparse CNN and tabulated the accuracy of the training set and test set of the difference.

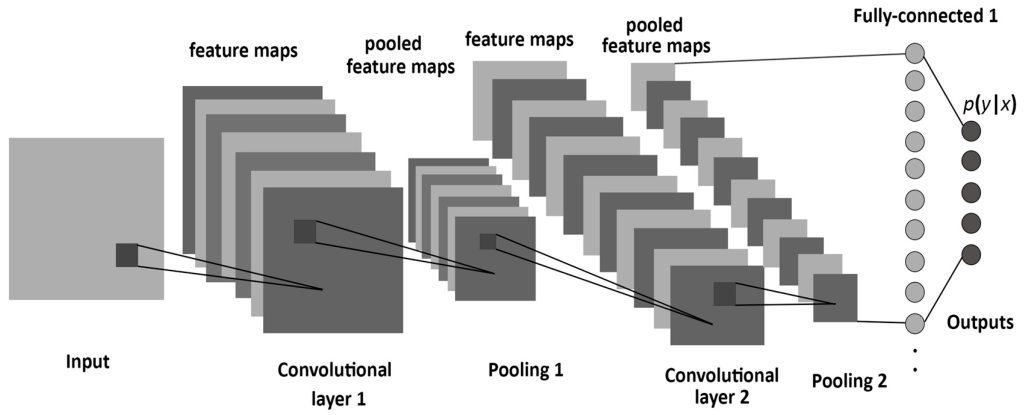
Shakti Punj, et al [1077] have proposed a system resolving problems like overfitting and reducing time complexity by increasing inputs, hidden layers, and altering epochs in a SimpleNet model. They have also drastically reduced memory consumption using deep compression.

The main goal of the project is to create a model and train it to identify and classify the 32x32 resolution images into the 10-class segregated by the CIFAR-10 datasets. The focus is mainly to increase the accuracy metric as much as possible. This includes high optimization techniques. The project is developed from a basic VGG model baseline to applying Data augmentation and Regularization techniques.

# IV. Architecture Of The Model

Figure 1 depicts the basic architecture and block diagram of a convolutional neural network. Convolution layers, max-pooling layers, and fully connected dense layers along with batch normalization layers and dropout layers and a flatten layer make up the final network.





**Fig. 2** Typical Block Diagram of 2-layer CNN

1. Convolutional Layers: The input image and a stack of filters/features make up the convolution layer. The image array and the filter are convoluted in 2D, resulting in a stack of filtered image arrays at the output. Numbers reflect how closely the filter fitted that section of the image in the filtered photos. This becomes the location map for the feature. The convolution layer refers to the process of convolving an image with several filters to create a stack of filtered images. The network's initial convolution layer takes a single image as input and outputs a stack of 32 filtered images.
2. Batch Normalization Layers: This layer particularly allows every layer to do learning more independently. It also reduces the internal covariance shift and affects training speed by increasing it.
3. Max Pooling Layers: After every two convolution layers, the network architecture employs max-pooling layers [7]. There are three pooling levels total, each measuring 2x2. By maintaining the maximum value, these pooling layers reduce the image stack size. To produce a lower stack of filtered photos, Max Pooling is performed to the full stack of filtered images.
4. Dropout and Flatten Layers: Dropout is a technique where a selected neuron is ignored during training to prevent the neural network from overfitting. Flatten layer converts the multi-dimensional array to a 1-D array.
5. Dense Layer: The layer collects every single neuron from the preceding layer and creates a fully connected layer in which every output depends on every input

The main goal of the project is to create a model and train it to identify and classify the 32x32 resolution images into the 10-class segregated by the CIFAR-10 datasets. The focus is mainly to increase the accuracy metric as much as possible. This includes high optimization techniques. The project is developed from a basic VGG model baseline to applying Data augmentation and Regularization techniques.

# V. Proposed Work

The approach is to building the project Object Recognition model in CIFAR-10 using CNN can be discussed in the following steps:

* Explore the CIFAR-10 dataset
* Build a Baseline VGG16 model
* Apply Data augmentation and Regularization techniques
* Train and validate the model accuracy
* Test and predict the model with the test dataset

The method followed for building this model involves- Importing and training the data set, creating a test harness to conduct a thorough evaluation of a model and define a performance baseline for a classification task, boosting learning and model capacity by extending a baseline model, creating a finished model, test its performance, and apply it to new photos to make predictions.

The approach in the source code [4] uses 2D convolution, pooling and ReLU activation in the Simple CNN and Larger CNN which is improvised for obtaining a higher accuracy. The accuracy obtained is about 77.68%, which is not sufficient enough for an artificially intelligent application to compete with the human brain. Prediction involves the identification of a poor-quality image of size 32 x 32 pixels and categorization of it into the CIFAR-10 subsets with the least possible error, i.e., highest possible accuracy.

The chosen metric for evaluating the model performance through simulation is the accuracy, which is appropriate for the object recognition CIFAR-10 dataset. An SDG optimizer was used initially to compile the accuracy metric of the model.

The model accuracy has increased or improvised by the following the below ideas

* Training more Epochs
* Using Data Augmentation
* Designing a larger network topology
* Tuning the learning rate

# VI. Methodology

The project initially involves processing the 32x32 pixel image by loading the image and seeding it to get a repeatable result every time. The loaded data is normalized from 0-255 to 0, 1 and the categorical data variable is converted to a Machine learning algorithm using one-hot encoding. This improves prediction and classification accuracy. An alternative method that can be used is the Dummy Coding Scheme. Processed data or image was initially tested with the VGG baseline model, then using transfer learning the whole VGG16 model is used.

The whole project was worked out with 5 stages:

1. Baseline model
2. Baseline model + 0.2 Dropout
3. Baseline model + 0.2 Dropout + Batch Normalization
4. Baseline model + 0.2 Dropout + Batch Normalization + 100 epoch
5. VGG 16 + 0.2 Dropout + Batch Normalization + 100 epoch

The learning rate (lrate) parameter is tunable by step size to move towards the minimum loss function. The ideal value of lrate is between 1 and . Optimization technique: SGD, Stochastic Gradient Descent is replaced by ADAM, which is a combination of Adaptive Moment Optimization and RMS Propagation. The reason for the replacement is that SGD is comparatively more unstable than ADAM and ADAM helps in fast convergence. The metrics that are commonly reviewed to learn the performance in the classification are Accuracy, Precision, Recall, and F-1 score.

Data augmentation is an approach for artificially boosting the amount of data by adding slightly modified copies to current training data without actually gathering new data. Either data warping or oversampling artificially increases the amount of the training dataset or aids the model in preventing overfitting at the source of the problem. To avoid over-fitting, we used several augmentation factors such as rotation, flipping, and so on to enhance our data. The parameters that are utilized to enhance data are listed below.

1. Rotation range (15) – Input data generates with rotation from -15 to 15
2. Horizontal flip – Flip the image horizontal direction randomly
3. Width shift range (0.1) – Shifted, either towards the left or right randomly
4. Height shift range (0.1) – Shifted either towards the top or bottom randomly

The data is separated into two parts, validation data and training data, to check for overfitting. As the model is trained, the error is plotted on the training and validation statistics. Overfitting was identified by a substantial discrepancy in the error graph between the training and validation datasets. Reduce model complexity using regularization or enhance dataset using approaches such as data augmentation to reduce overfitting. Adding a component to our cost function that penalizes too complex models is one method to apply regularization.

**Table 1.** Classification of loss curves [4]

| **Training Loss** | **Validation Loss** | **Loss curves** |
| --- | --- | --- |
| Low | Low | Good fit |
| Low | High | Overfitting |
| High | Low | Unlikely |
| High | High | Underfitting |

The variance of underfitting data is low, but the bias is large, whereas the variance of an overfitted model is low, but the bias is high. Our model had a low training loss but a large validation loss at first, indicating that it was overfitting. Overfitting can be avoided by reducing capacity through regularization and Dropout while increasing data through data augmentation.

# VI1. Training

Several parameters are used to be tuned up in our proposed CNN model. However, the principal once that we have used here is given below.

* Batch size (128): the number of processed images in every iteration
* Learning rate (0.001): It is the amount that the weights are updated during the training process.
* Optimizer: Adam is the optimizer that we utilized. It is a stochastic gradient descent substitution optimization approach for minimizing the loss function for training DL models. It is computationally efficient, uses little memory, and is well suited to issues with a lot of data, parameters.
* Loss Function: ReLU is used as a loss function. It measures the performance of a classification model whose

output is a probability value between 0 and 1.

* The number of epochs (100): The number of times the using dataset is passed to the created model.

During training, Dropout drops out neurons at random with a certain probability. Every iteration of the training algorithm employs a random subset of the network. This method enables neurons to learn useful traits without the assistance of other neurons. The complete network is used for inference once it has been trained. To avoid overfitting, we can either limit the network's effective capacity by employing methods like dropout, or we can increase the amount of training data by utilizing data augmentation to generate changed versions of the samples we currently have.

# VIII. Comparison

The baseline model includes 6 convolutional layers, 3 max-pooling layers, a flatten and 3 dense layers and was trained with SGD optimization, categorical cross entropy and ReLU activation. Using a dropout layer of 20% after every block increased the accuracy. Next step of the project is to introduce a regularization method, batch normalization after every convolutional layer of the basic model, which indeed increased the accuracy. Data augmentation is an important part of the project as it trains the model to classify the image into the 10 classes even when the orientation of the image is different. Finally, epochs are increased as the dataset is large, to get more accurate results.

**Table 2**. Accuracy Comparison in percentage

| **Method** | **Acc** | **Val\_Acc** | **Loss** | **Val\_Loss** |
| --- | --- | --- | --- | --- |
| BL | 78 | 76 | 43 | 85 |
| BL+DO | 88 | 82 | 32 | 53 |
| BL+DO+BN | 86 | 85 | 38 | 46 |
| BL+DO+BN+100e | 97 | 88 | 16 | 42 |

Where BL, DO, BN, 100e, Acc and Val, indicates Baseline, Dropout, Batch Normalization, 100 epochs, Accuracy and Validation respectively.

From table 1, our model has developed from overfitting to good fit by making significant changes and from table 2, it is clearly visible that every method used in the model has increased the training and validation accuracy. It has also reduced the losses to a minimum significant level.

# IX. Results

The CIFAR-10 data set was used to train the final model.

The dataset is split into two parts: training data (50000 images) and testing data (10,000 images). For 100 epochs, the network is trained, 128 batches were created from the training data. As a result, one epoch is completed when the network has been trained on all 128 batches, each batch containing 256 images. The model is updated to a VGG 16 model, and there are two methods for doing so: coding the model or importing a previously trained model via Transfer learning.

**Table 3**. Model Summary

| **Block** | **Layer** | **Shape** | **Parameter** |
| --- | --- | --- | --- |
| **1** | Conv2D | 32,32,32 | 896 |
|  | BatchNorm | 32,32,32 | 128 |
|  | Conv2D | 32,32,32 | 9248 |
|  | BatchNorm | 32,32,32 | 128 |
|  | MaxPooling, Dropout | 16,16,32 | 0 |
| **2** | Conv2D | 16,16,64 | 18496 |
|  | BatchNorm | 16,16,64 | 256 |
|  | Conv2D | 16,16,64 | 36928 |
|  | BatchNorm | 16,16,64 | 256 |
|  | MaxPooling, Dropout | 8,8,64 | 0 |
| **3** | Conv2D | 8,8,128 | 73856 |
|  | BatchNorm | 8,8,128 | 512 |
|  | Conv2D | 8,8,128 | 147584 |
|  | BatchNorm | 8,8,128 | 512 |
|  | Conv2D | 8,8,128 | 147584 |
|  | BatchNorm | 8,8,128 | 512 |
|  | Conv2D | 8,8,128 | 147584 |
|  | BatchNorm | 8,8,128 | 512 |
|  | MaxPooling, Dropout | 4,4,128 | 0 |
| **4** | Conv2D | 4,4,256 | 295168 |
|  | BatchNorm | 4,4,256 | 1024 |
|  | Conv2D | 4,4,256 | 590080 |
|  | BatchNorm | 4,4,256 | 1024 |
|  | Conv2D | 4,4,256 | 567030 |
|  | BatchNorm | 4,4,256 | 984 |
|  | Conv2D | 4,4,256 | 567040 |
|  | BatchNorm | 4,4,256 | 1024 |
|  | MaxPooling, Dropout | 2,2,256 | 0 |
| **5** | Conv2D | 2,2,512 | 1180160 |
|  | BatchNorm | 2,2,512 | 2048 |
|  | Conv2D | 2,2,512 | 2359808 |
|  | BatchNorm | 2,2,512 | 2048 |
|  | Conv2D | 2,2,512 | 2359808 |
|  | BatchNorm | 2,2,512 | 2048 |
|  | Conv2D | 2,2,512 | 2359808 |
|  | BatchNorm | 2,2,512 | 2048 |
|  | MaxPooling, Dropout | 1,1,512 | 0 |
| **6** | Flatten | 512 | 0 |
|  | Dense | 1024 | 525312 |
|  | BatchNorm | 1024 | 4096 |
|  | Dropout | 1024 | 0 |
|  | Dense | 512 | 524800 |
|  | BatchNorm | 512 | 2048 |
|  | Dropout | 512 | 0 |
|  | Dense | 256 | 131328 |
|  | BatchNorm | 256 | 1024 |
|  | Dropout | 256 | 0 |
|  | Dense | 128 | 32896 |
|  | BatchNorm | 128 | 512 |
|  | Dropout | 128 | 0 |
|  | Dense | 10 | 1290 |
|  | **Total Parameters** |  | **12,099,448** |

The results of training for 100 epochs are shown below in table 4 about the accuracy metric and classification report. Additionally, the confusion matrix is also used to summarize the results.

**Table 4**. Accuracy

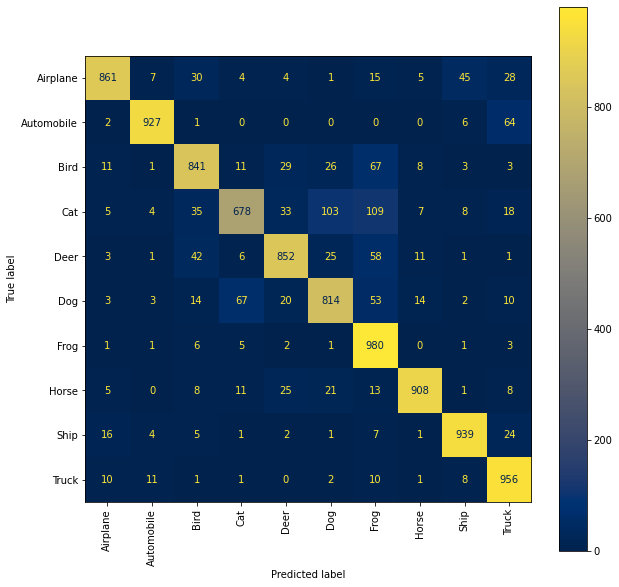
| **Validation Accuracy** | 87.56% |
| --- | --- |
| **Train Accuracy** | 96.12% |

A classification report of table 8 is a performance evaluation metric in machine learning. It is used to show the precision, recall, F1 Score, and support of your trained classification model.

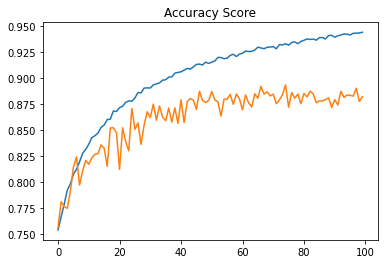
**Table 5**. Classification report

|  | Precision | Recall | F1 score | Support |
| --- | --- | --- | --- | --- |
| 0 | 0.94 | 0.86 | 0.90 | 1000 |
| 1 | 0.97 | 0.93 | 0.95 | 1000 |
| 2 | 0.86 | 0.84 | 0.85 | 1000 |
| 3 | 0.86 | 0.68 | 0.76 | 1000 |
| 4 | 0.88 | 0.85 | 0.87 | 1000 |
| 5 | 0.82 | 0.81 | 0.82 | 1000 |
| 6 | 0.75 | 0.98 | 0.85 | 1000 |
| 7 | 0.95 | 0.91 | 0.93 | 1000 |
| 8 | 0.93 | 0.94 | 0.93 | 1000 |
| 9 | 0.86 | 0.96 | 0.90 | 1000 |
| Accuracy |  |  | 0.88 | 10000 |
| Macro avg | 0.88 | 0.88 | 0.87 | 10000 |
| Weighted avg | 0.88 | 0.88 | 0.87 | 10000 |

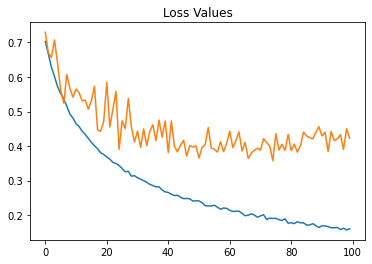
Figure 3 below shows the confusion matrix for the final model, which indicates the true positives, false negatives, false positives and true negatives. This is an additional factor to verify the classification report.

**Fig. 3** Confusion Matrix

The initial network is overfitted, as evidenced by the difference between the training and validation curves, as seen in figure 4 and 5 before performing data augmentation.

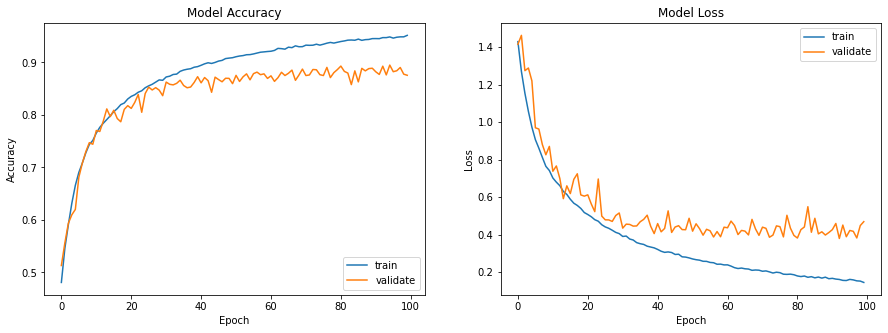


**Fig. 4** Accuracy vs Epochs curve for overfitted model

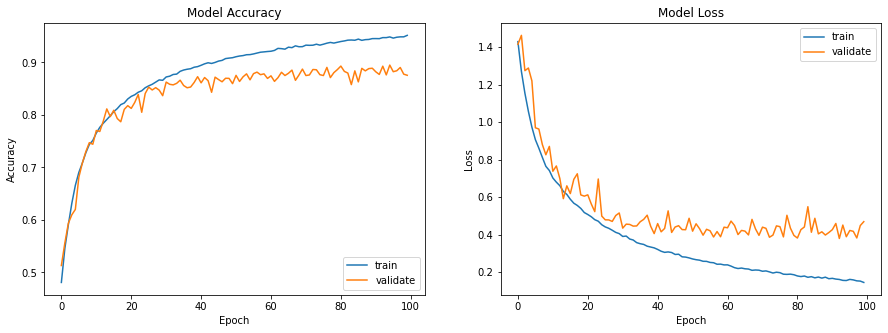


**Fig. 5** Loss vs Epochs curve for overfitted model

The figures below show the effect of applying data augmentation to the final network where the overfitting is reduced.



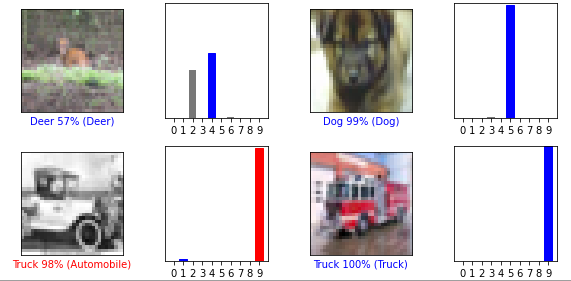
**Fig. 6** Accuracy vs Epochs curve for better fit model



**Fig. 7** Loss vs Epochs curve for better fit model

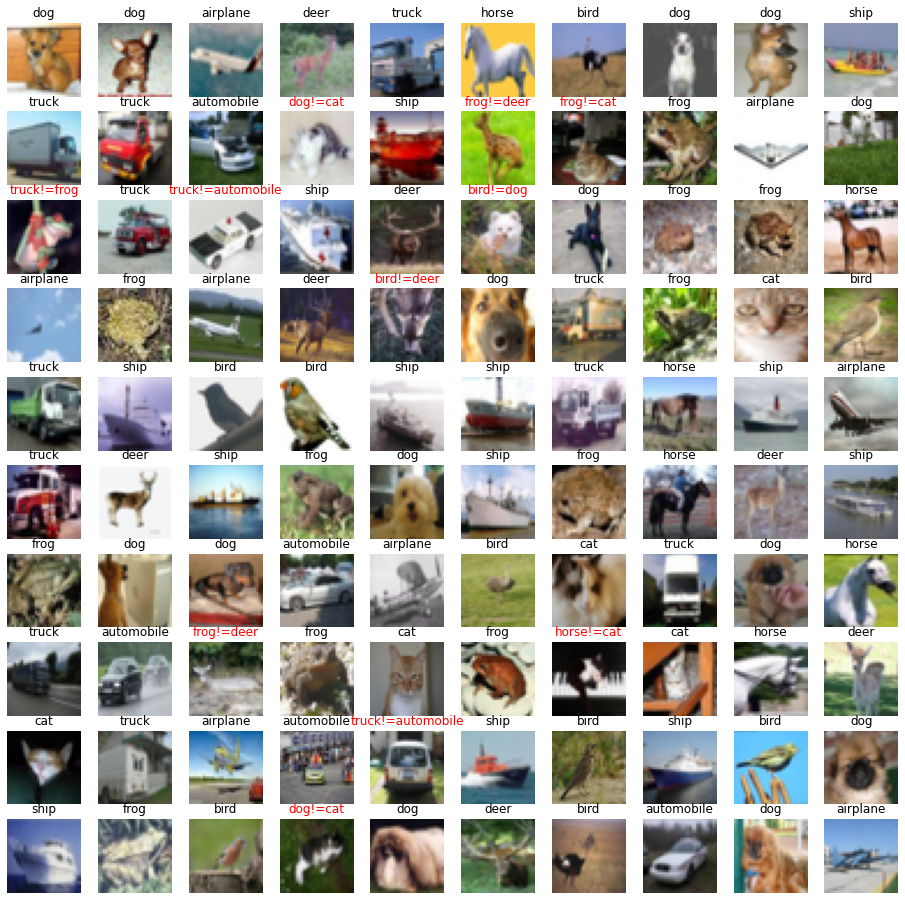
# X. Prediction

The project results are verified using 2 different prediction method. A bar graph representation in the figure 8, showing the percentage of the classification in the respective classes.



**Fig. 8** Bar graph

The other is an accuracy calculation by classifying 100 32x32 pixel images and verifying it with its actual class classification.



**Fig. 9** Percentage estimator

The above prediction gives the following results

* The number of correct answers: 89
* The number of mistake: 11
* A correct answer rate: 89.0 %

# XI. Future Work

Overfitting can be further reduced by introducing ensembling, input pip and deeper neural network. Different methods that can be implemented are [9]

* Pixel Scaling: Comparing the performance of several methods for scaling the pixels, such as centering and standardization.
* Learning Rates: Comparing the performance of different learning rates, adaptive learning rates, and learning rate schedules.
* Transfer Learning: On this dataset, we can try employing transfer learning, such as a pre-trained VGG-16 model.

# XII. Conclusion

Convolutional Neural Network is used to classify photos into ten categories in this project. The network was trained using the CIFAR-10 dataset as a baseline. In terms of picture categorization, the Adam optimization strategy for updating weights provides us with the best accuracy. Regularization and dropout approaches are used to reduce model overfitting. We were able to attain 96.12% accuracy on the training data and 87.56 percent accuracy on the test set using deep network architecture.

References

1. J. D. Kelleher, Deep learning. MIT press, 2019
2. U. Karn, “An intuitive explanation of convolutional neural networks”. Data science blog, 2016.
3. Hope, Tom; Resheff, Yehezkel S.; Lieder, Itay (2017-08-09). “[Learning TensorFlow: A Guide to Building Deep Learning Systems](https://books.google.com/books?id=sUowDwAAQBAJ&pg=PA64).” "O'Reilly Media, Inc.". pp. 64–.[ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [9781491978481](https://en.wikipedia.org/wiki/Special:BookSources/9781491978481). Retrieved 22 January 2018.
4. [https://github.com/CeciliaXiYang/neural-network-projects/blob/master/Object Recognition in CIFAR-10](https://github.com/CeciliaXiYang/neural-network-projects/blob/master/Object%20Recognition%20in%20CIFAR-10) with Convolutional Neural Networks/source.ipynb **(Source)**
5. Kruithof, Maarten & Bouma, Henri & Fischer, Noëlle & Schutte, Klamer. (2016). Object recognition using deep convolutional neural networks with complete transfer and partial frozen layers. 99950J. 10.1117/12.2241177.
6. <https://www.cs.toronto.edu/~kriz/cifar.html>
7. R. Doon, T. Kumar Rawat, and S. Gautam, "Cifar-10 Classification using Deep Convolutional Neural Network," 2018 IEEE Punecon, 2018, pp. 1-5, DOI: 10.1109/PUNECON.2018.8745428.
8. <http://cs229.stanford.edu/proj2014/Wenqing> Yang, Harvey Han, Object Recognition in Images.pdf
9. <https://machinelearningmastery.com/how-to-develop-a-cnn-from-scratch-for-cifar-10-photo-classification/>
10. <https://www.irjet.net/archives/V8/i6/IRJET-V8I660.pdf>