Technical Report on Artificial Vision and Deep Learning

By

Chukwudi Emmanuel Agbaraji

Abstract

This work covers two different tasks, section 1 and section 2. In section 1, the study focuses on droplet recognition from a video using a computer vision algorithm was successfully carried out using a YOLO algorithm which is based on CNN layers. From the results of the study, inner and outer wraps of the droplets in the video were recognized by fixing square frames on each droplet. The successfully formed droplets were recognized and counted, and 11 droplets were counted. The centre of mass/blob of the droplets were detected and marked by adding a dot at the centre of each droplet. In section 2, the study aims to classify the images in a dataset called CIFAR-10 dataset through the experimentation of different convolutional layers using Convolutional Neural Networks (CNNs). Many experiments were carried out using various CNN model enhancement techniques. The CNN model achieved above 70% accuracy at the last experiment which is better than that in one in practical 9.

1 Introduction

This report covers two practical sections of the course work, sections one and two. Section one is on recognition of the droplets from a video using computer vision algorithms that is taken by a high-speed camera during the droplet 3D printing process. Section two is on designing a Convolutional Neural Network (CNN) architecture trained by the CIFAR-10 dataset, to classify the CIFAR-10 test images with improved accuracy.

Background

For section one:

several processes are involved in applying computer vision algorithms to identify droplets from a video, they include preprocessing, segmentation, feature extraction and classification. To increase the quality of the images, the video frames may go through preprocessing techniques like contrast enhancement, noise reduction, or image stabilization. In order to isolate droplets from the background, segmentation techniques are used, and to find regions of interest, this may require using thresholding, edge detection, or clustering techniques. Features like size, shape, and colour can be taken from each droplet region to provide a description of the droplets once they have been segmented. The droplets can then be sorted according to their characteristics using deep learning or pattern recognition algorithms. Using characteristics like size, shape, or activity, droplets can be categorized in this stage to separate them from other items.

Droplet recognition in computer vision tasks can be found in many recent world applications such as biomedical Imaging, Manufacturing processes, Environmental monitoring.

The aim of this work is to recognize droplets from a video using computer vision algorithms. The main objectives of this work are as follows:

i. To recognize the inner droplet.
ii. To recognize the outer wrap.
iii. To count for all the successfully formed droplets.
iv. To do centre of mass/blob detection.

For section two:

The second section of the coursework is a task which is based on the NN (Neural Network) for the design and performance improvement of a Convolutional Neural Network (CNN) architecture-based image classification model trained with a dataset which contains different sets of images.

The aim of this study is to classify the images in a dataset called CIFAR-10 dataset through the experimentation of different convolutional layers using Convolutional Neural Networks (CNNs). This work intends to improve the effectiveness of automated image recognition systems and classification by improving the CNN image recognition and classification model.

Image recognition and classification has potential applications in areas such as autonomous vehicles, security, medical imaging, robotics and surveillance.

In order to improve the performance of the network, the following objectives were defined:

To examine the model performance using different optimizers, different learning rates and a different number of epochs.

To use different designs of convolution layers and activation functions, and to add different layers and change the hyperparameters to modify the architecture and to improve the performance of the model.

To explore the classes which are defined as wrong classes according to the confusion matrix and to find a method to build the model to have a better prediction of these classes.

To apply data augmentation techniques to examine the model improvement.

To check the performance with and without batch normalisation or drop-out.

2. Design

2.1 For Section 1 (Droplet Detection using Computer Vision Algorithm)

2.1.1. Design Method

The computer vision algorithm applied in this is the You Only Look Once (YOLO) [1]. It is a cutting-edge computer vision object recognition system based on Convolutional Neural Network (CNN) layers, that frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. This allows for high accuracy and real-time performance.

Yolo splits the input image into a grid of cells and concurrently forecasts class probabilities and bounding boxes for each cell. The center coordinates and box dimensions are represented by the four coordinates (x, y, w, h) that make up each bounding box prediction. Additionally, confidence ratings that indicate the likelihood that the box contains an object and class probabilities for several object categories are included.

2.1.2 Data Description

The data used in this work is a video file which contains the video taken by a high-speed camera during droplet 3D printing procedure as shown in figure 1.

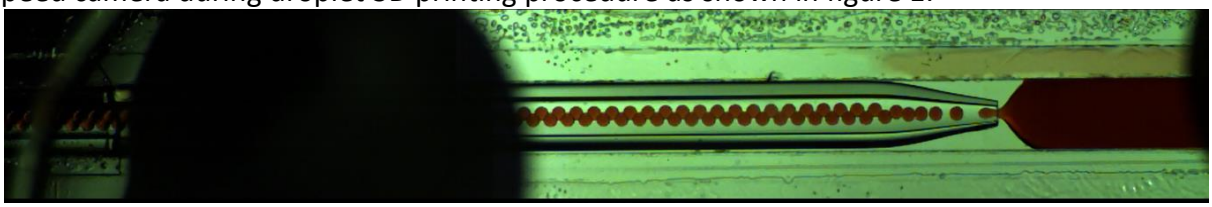

Figure 1: The video from a high-speed camera showing the droplets

2.1.3 Design Experiments

In these experiments, the video file was processed using the Python code 1 in the appendix, which makes use of the OpenCV package (cv).

2.1.3.1 Inner Wrap Detection

In order to achieve this objective, the video was loaded using cv.VideoCapture() function, all the frames in the video were processed and a deep learning model was applied to detect inner wrap droplets in each frame. The detected inner wrap of the droplets on the frame were annotated and the annotated frames are saved to an output video file.

2.1.3.2 Outer Wrap Detection

The cv.VideoCapture() function was used to load the video, each frame was processed, and then a deep learning model was used to identify outer wrap droplets in each frame in order to accomplish the goal. An output video file was generated to contain the annotated frames together with the detected outer wrap of the droplets on the frame.

2.1.3.3 Count Successfully Formed Droplets

This section of the Python code receives a video stream, processes it, detect and counts objects in each the frames, and outputs a video with object count data. The following vital steps were involved in this segment:

Importing Libraries: This step involves importing libraries that enable object counting in the video. Import object_counter module from the Ultralytics library, which provides functionalities related to object detection and counting. b64encode function was imported for encoding binary data as base64.

Video Input Setup and Object Counter: The code at this stage opens the video file and the object_counter.ObjectCounter() function initializes the object counter using the ObjectCounter class from the module imported.

Object Detection and Counting: This involves, processing every frame of the loaded video, detecting and tracking the objects in the frame using deep learning model. The code counts objects in the frame by calling the start_counting method of the counter object and updates the frame with count information.

2.1.3.4 Centre of Mass Detection

This section of code adds centre markers to the objects that are detected in each frame of the loaded video and uses the YOLO model for object detection. The recognized items and center markers are then included in an annotated output video that is generated and displayed.

2.2 SECTION 2 (Design and Improvement of a CNN model for Image Classification using CIFAR-10 Dataset)

2.2.1 Data Description

The CIFAR-10 dataset, which is typically used for image classification tasks in computer vision and machine learning research, consists of 60,000 32x32 color images divided into 10 classes, with 6,000 images in each class. Using the tensorflow software tool, the dataset for this study will be imported. The list of such images are: Airplanes, Automobiles, birds, cats, deer, dogs, frogs, horses, ships, trucks.

2.2.2 Design Experiments

Package Importation: Many packages required to execute achieve the aim of this work were imported such as the OpenCV (cv) for image processing, tensorflow for deep learning operations, seaborn for ploting of graphs, etc.

ii. Loading dataset: The CIFAR-10 dataset was loaded using the function: cifar10.load_data(). The function was used to load both the train and test datasets.

Data Analysis: The data was analysed to explore the containt of the CIFAR-10 dataset before processing it. This was carried out using the function: tf.unique(cifar_train_labels[:, 0])[0].

Class Imbalance Check

The class imbalance check was done by counting the occurrence of each class and plotting a bar chart to demonstrate the imbalance.

Image Data Processing: The images in the dataset were processed based on three methods: normalization, augmentation and label encoding.

This was done by dividing the pixel values of the training and testing images by 255, so that the pixel values remain constant between 0 and 1.

Model Development:
A few CNN-based deep learning models were examined in this study in order to classify the cifar-10 images into their respective classes. This process includes experimentation with different optimizers, different learning rates and a different number of epochs. In order to achieve this, feature extraction using convolutional layers to extract intricate features from the train images was carried out.

For the Custom CNN Model
The model was compiled initially using the Adam optimizer by setting the learning rate to 0.001.
Batch size of 32 was used to update the weights of the model after processing each batch of 32 training samples while the data will be trained for 20 epochs.
The custom model was evaluated based on the accuracy and loss validations.

Experiment 1
In this experiment, basic CNN without Data Augmentation was used and the images were normalized without Augmentation. Adam optimization was applied with 20 epochs, 32 batch sizes, learning rate of 0.001, ReLU activation method and two convolutional blocks.

Experiment 2
In the second experiment, basic CNN with Data Augmentation was used and the images were normalized with augmentation. Adam optimization was applied with 15 epochs, 64 batch sizes, learning rate of 0.001, ReLU activation method and two convolutional blocks.

Experiment 3

In the third experiment, custom CNN with tuned hyperparameter, without Data Augmentation was used and the images were normalized without augmentation. SGF optimization was applied with 15 epochs, 64 batch sizes, learning rate of 0.0001, leaky_ReLU activation method and three convolutional blocks were used for the experiment.

Experiment 4

In this experiment, Custom CNN model was used and images were normalized without augmentation. SGF optimization was applied with dropout. 20 epochs, 32 batch sizes, learning rate of 0.001, ReLU activation method and two convolutional blocks were used for the experiment.

Experiment 5

In this experiment, transfer learning methods was applied by leveraging on pre-trianed weights from previous models and adapting them to this experiment with the objective of improving the performance of the model. To achieve this, Densenet-201 was applied.

Experiment 6

In this experiment, the CNN model was enhanced by using hyperparameter tuning technique. This was done by generating the best hyperparameter combination using RandomSearch and using the weights of the first experiment.

3. Results

3.1 Section 1 Results

This subsection presents the results from the execution of section 1 code for the recognition of droplets from a video using computer vision algorithms.

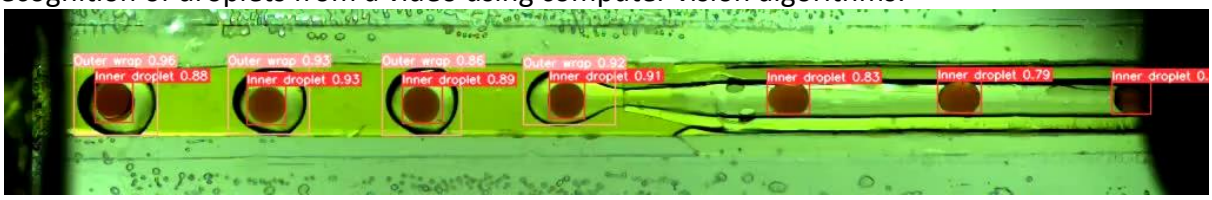

Figure 2: Inner droplet detection

The result in figure 2 show the detection of the inner and outer wraps of the droplets using the YOLO computer vision algorithm. The inner and outer wraps can be seen and identified with the square frames tracking every droplet as they move in the video.

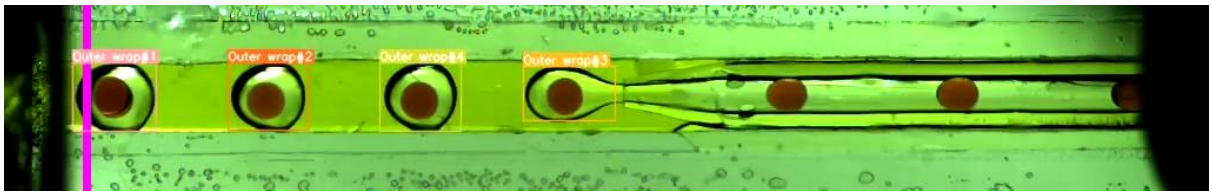

Figure 3: Droplet detection and counting from 1 to 3

The results in figure 3 demonstrates the object detection and counting for the detection and counting of the droplets in the 3D printing process. In this result, the counts start from 1.

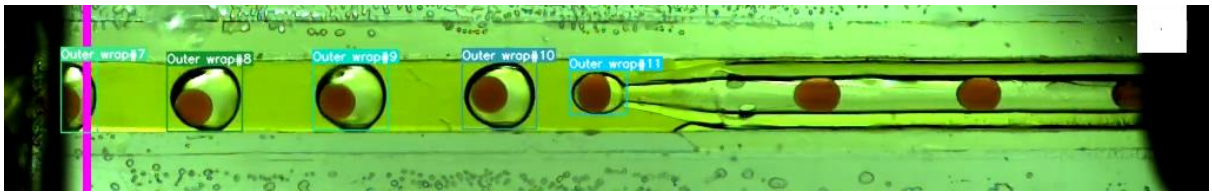

Figure 4: Droplet detection and counting from 7 to 11

The results in figure 4 shows the continuation of the droplet detection and counting process which counted to 11 droplets. Hence, the algorithm was able to detect and count the droplets from 1 to 11, making it 11 droplets counted in the video.

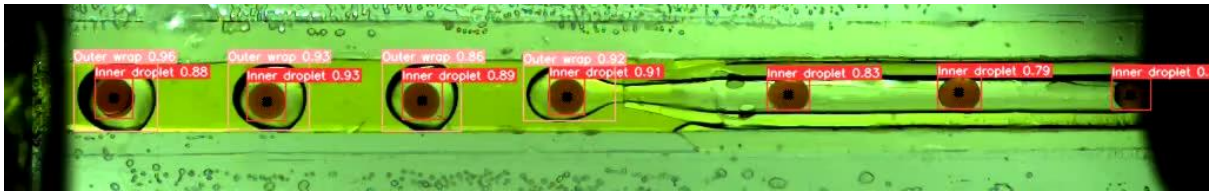

Figure 5: Droplet centre of mass/blob detection

The results in figure 5 demonstrate the centre of mass/blob detection using the computer vision algorithm. In order to demonstrate the detection of the centre of mass/blob, the code added a mark at the centre of each droplet.

3.2 Section 2 Results

Data Analysis

Class 6: frog
Class 9: truck
Class 4: deer
Class 1: automobile
Class 2: bird
Class 7: horse
Class 8: ship
Class 3: cat
Class 5: dog
Class 0: airplane

Checking for Class Imbalance

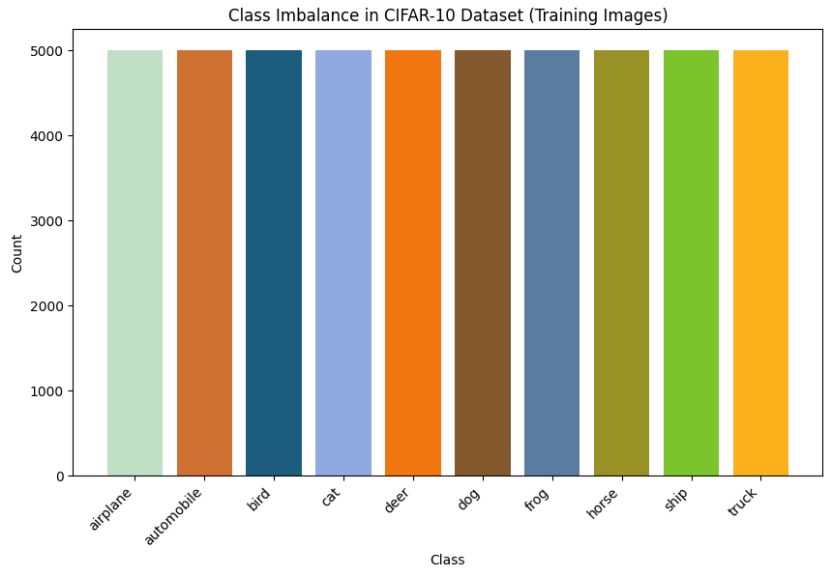

Figure 6: Class imbalance check

The results of the imbalance of the classes in the CIFAR-10 dataset was recorded in bar chart a demonstrated in figure 6.

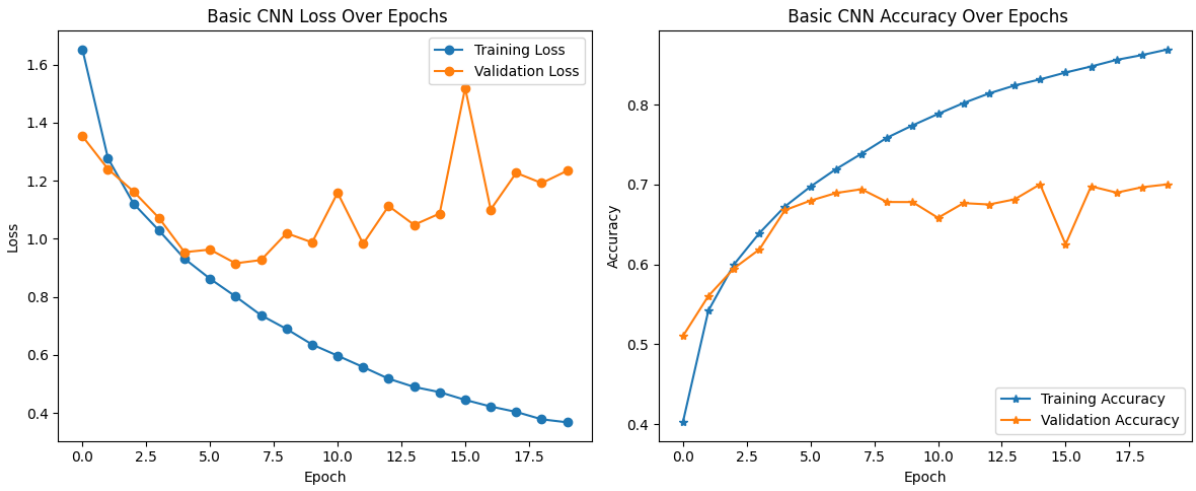

Figure 7: Loss and accuracy graphs for the custom CNN model

The results in figure 7 shows that the model learnt complex characteristics from the Cifar-10 dataset, as evidenced by the fact that the accuracy grew steadily from the first to the last epoch and the model loss decreased throughout each epoch. But compared to the validation performance, the training performance is better. Thus, there is a slight overfitting of the model.

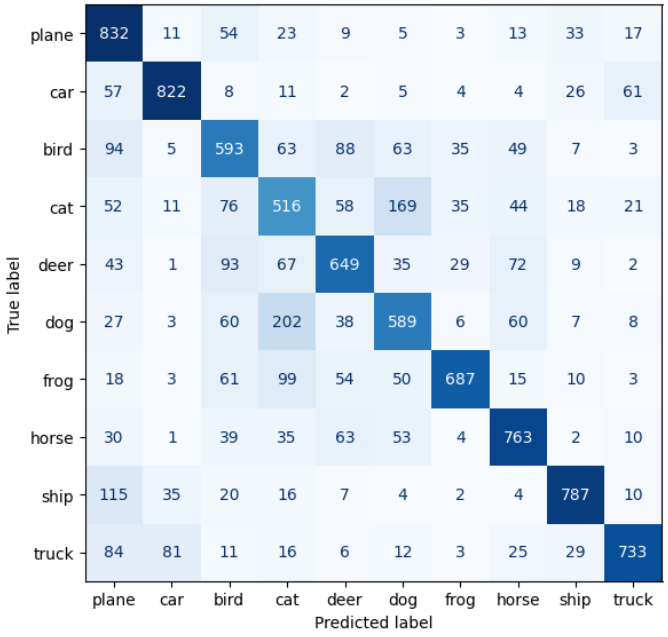

Figure 8: Confusion matrix for the custom CNN model

The results in figure 8 shows that most of the test images from CIFAR were correctly identified, as indicated by the diagonal of the confusion matrix. 832 images of plane were correctly identified as such with accuracy. 822 automobile images had accurate classifications.

On the other hand, the model presents a small number of false positives and false negatives. For example, 81 images of trucks were incorrectly identified as vehicles, while 202 images of dogs were incorrectly classified as cats.

To solve this misclassification, further techniques were tested, such as augmentation, batch normalization removal, experimentation with various optimizers, and transfer learning, etc.

A performance of roughly 70% is shown by the first model across the key classification metrics. This performance is better than the experiment in practical 9 which gave about 60% and can be further improved by experimenting with other techniques like optimizer tweaking, hyperparameter tuning, augmentation, and so on.

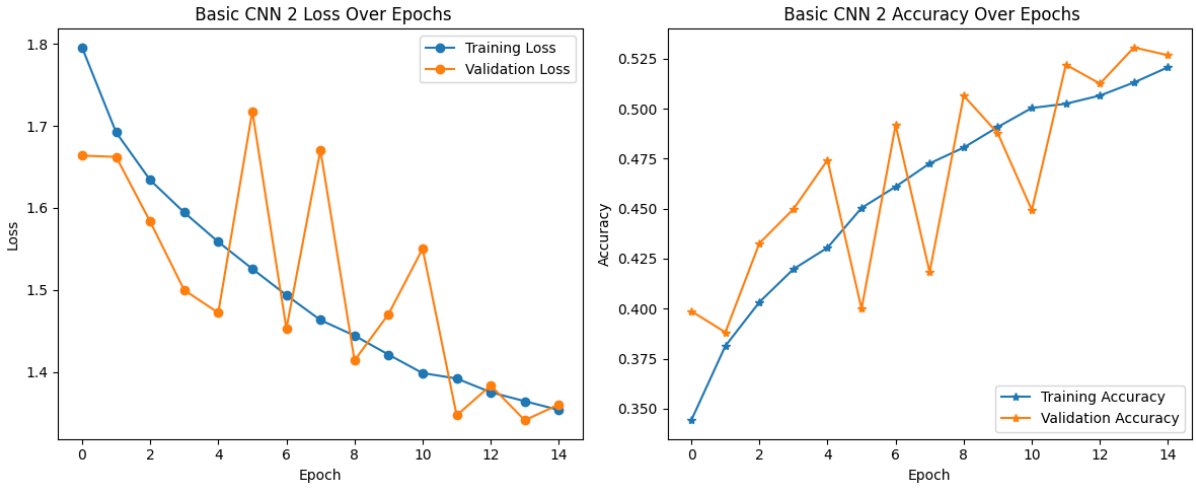

Figure 9: Loss and accuracy graphs of experiment 2

The results in figure 9 shows that the model in second experiment shows slightly overfitting characteristics.

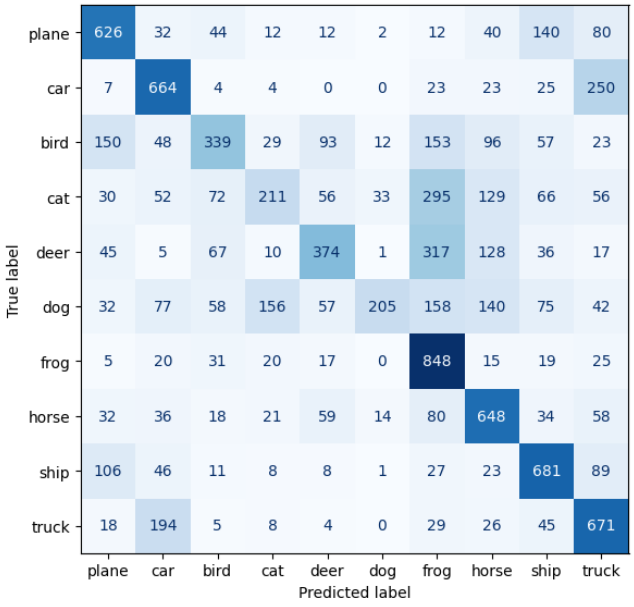

Figure 10: Confusion matrix of the second experiment

Figure 10 shows that the model built using the augmented data committed more misclassification than that of the original dataset.

The second experiment recorded accuracy of approximately 53% which is less than that in practical 9.

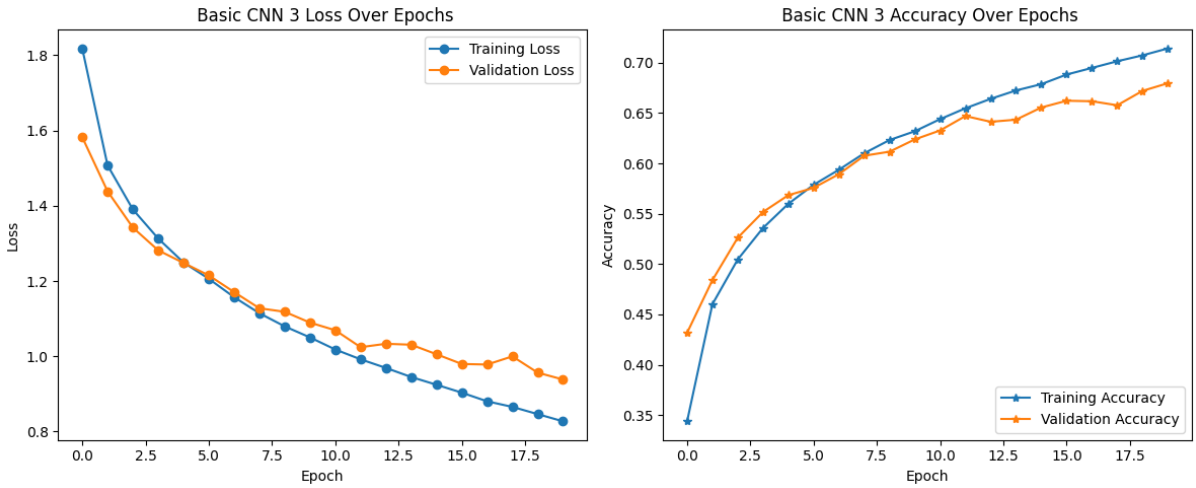

Figure 11: Loss and accuracy graphs of experiment 3

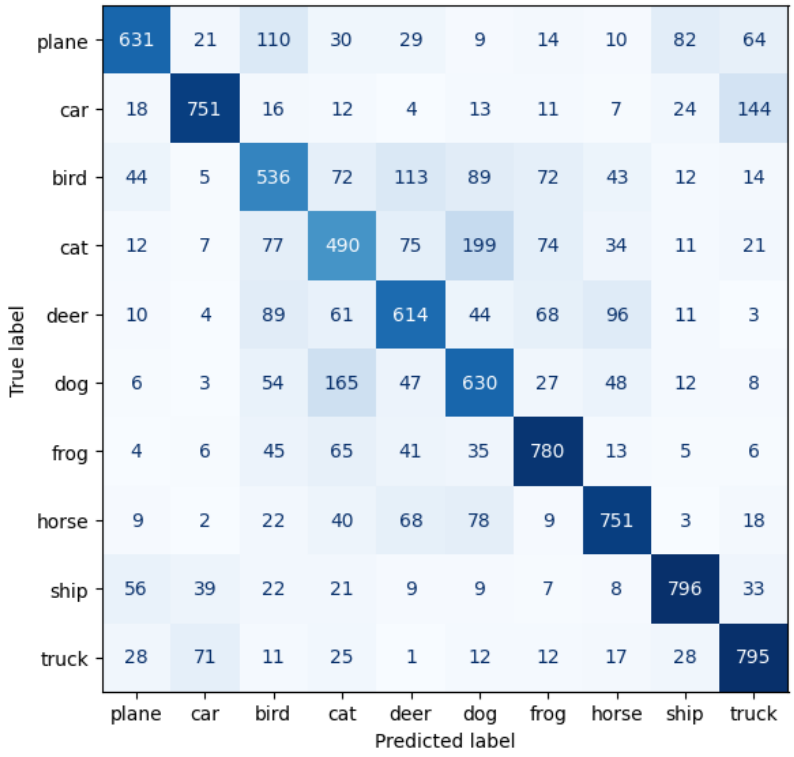

Figure 12: Confusion matrix for the third experiment

Comparing the first model with the results in figure 11 and figure 12, which has two convolutional blocks with batch normalization and dropout, to the third model experiment, which was developed without these features, dropout, lowered learning rate, and batch normalization, the first model performed better. The third experiment recorded accuracy of approximately 68% which is also better than that in practical 9.

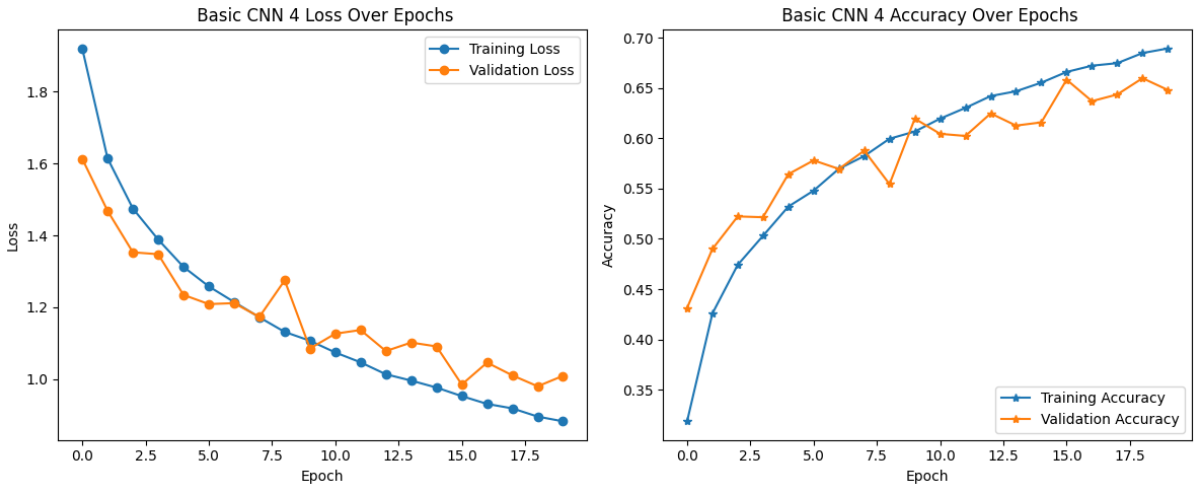

Figure 13: Figure 11: Loss and accuracy graphs of experiment 4

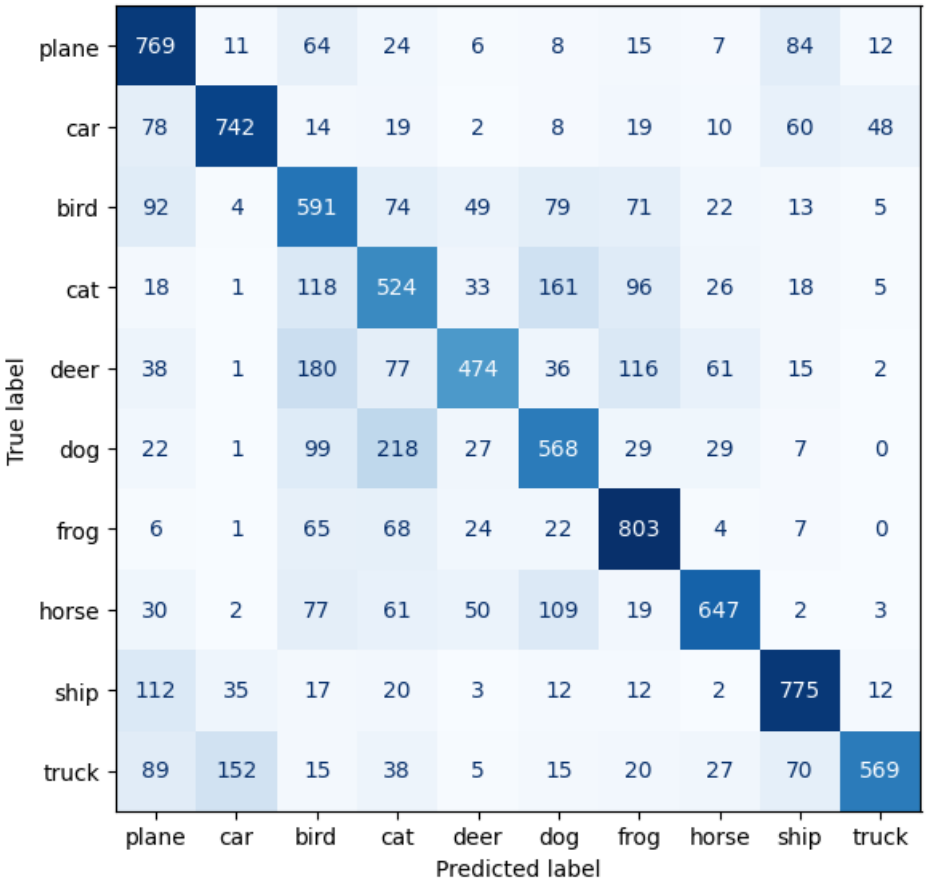

Figure 14: Confusion matrix for experiment 4

The results in figures 13 and 14, the model in experiment 4 performed with lesser performance characteristics than the first experiment and it recorded and accuracy of 65% which is also better than the one practical 9.

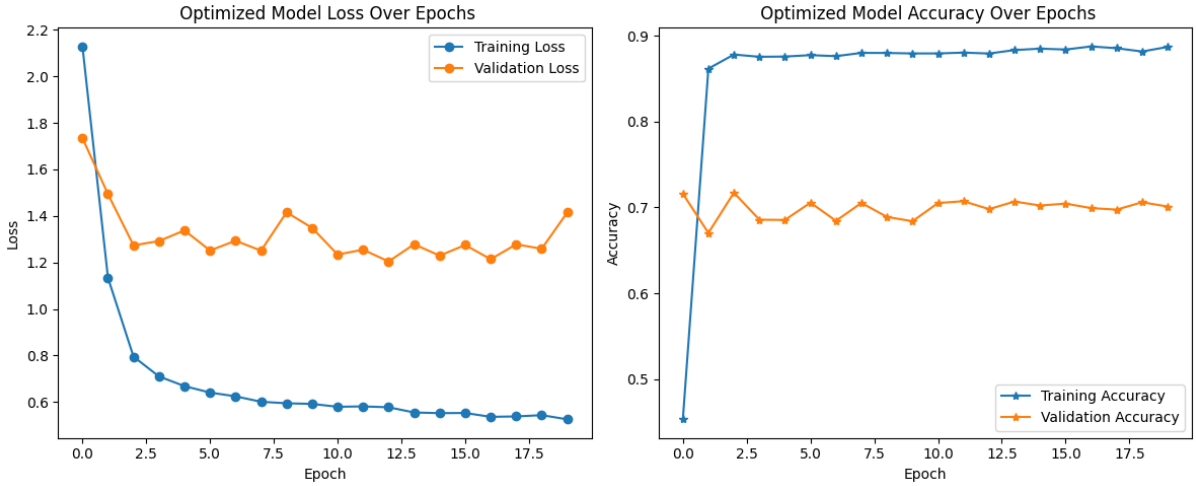

Figure 15: Loss and accuracy graphs of experiment 6

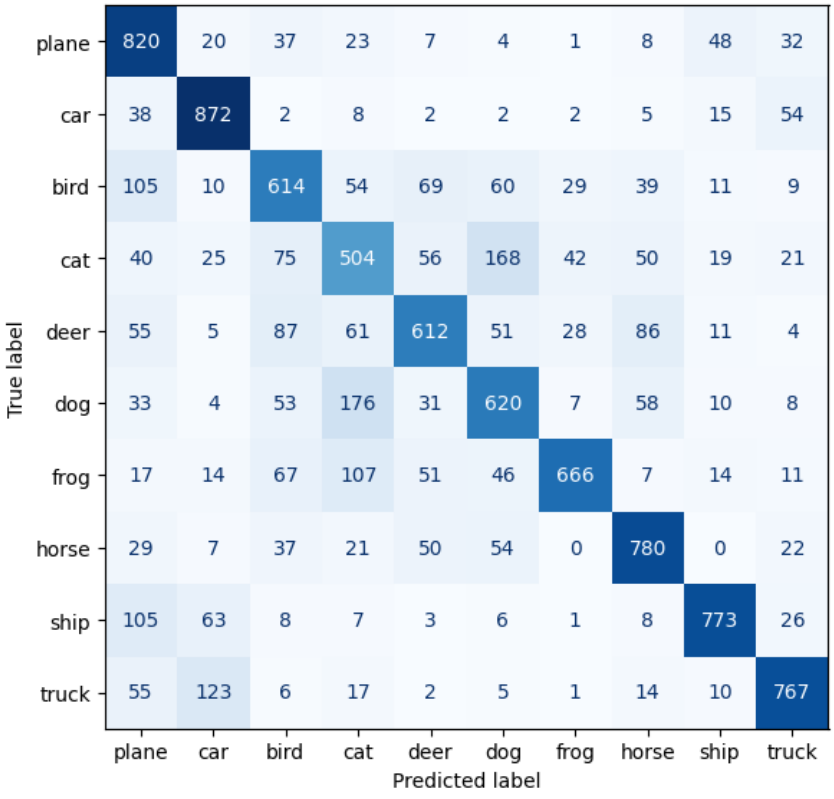

Figure 16: Confusion matrix of experiment 6.

From the results in figure 15 and figure 16, the model in experiment 6 achieve improvement like that in experiment 1. As indicated by the diagonal line, the optimized model successfully classified most of the cifar-10 test images into the appropriate classes. But some classes were incorrectly categorized. The following summarizes the

confusion matrix. It recorded 70% accuracy which is better than that in practical 9 and slightly better than the experiment 1.

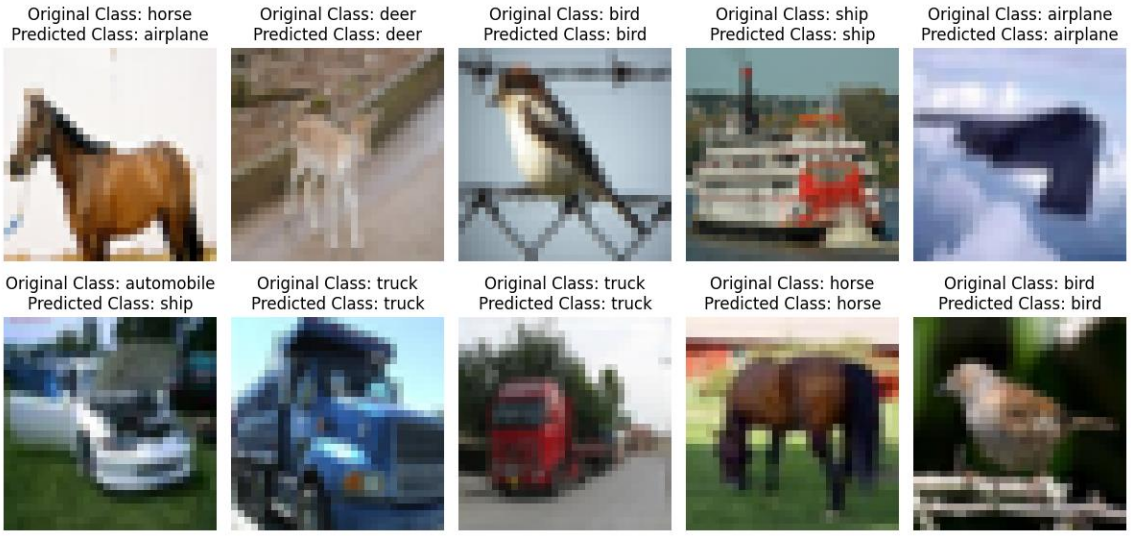

Figure 17: Model testing results

The results in figure 17 show that the model generated at the final experiment demonstrates best performance ability in image classification from real world dataset.

3.2. 1 Observations

Eight of the ten (10) test images that were selected at random could be accurately predicted by the final model.

With a sigmoid activation function, a dropout rate of 0.4, a regularizer of 0.01, a learning rate of 0.01 and an rmsprop optimizer, the optimized model shows the best performance to date.

Overfitting is there, but it's not noticeable because there isn't much of a difference between the training and validation accuracy.

Despite multiple efforts to enhance it, the model was unable to exceed a 70% performance criterion in terms of accuracy and precision. This constraint may result from the CIFAR-10 dataset's drastic reduction in size to 32 by 32 pixels, which might not be enough to capture fine-grained details.

Future research can investigate further strategies to enhance performance, such as experimenting with other cutting-edge transfer learning architectures and incorporating sophisticated data augmentation techniques.

4 Conclusion

The droplet recognition from a video using a computer vision algorithm was successfully carried out using a YOLO algorithm which is based on CNN layers. The objectives of this

work were achieved as well. The inner and outer wraps of the droplets in the video were recognized by fixing square frames on each droplet. The successfully formed droplets were recognized and counted, and 11 droplets were counted. The centre of mass/blob of the droplets were detected and marked by adding a dot at the centre of each droplet.

In section 2, the aim of the study which is to classify the images in a dataset called CIFAR-10 dataset through the experimentation of different convolutional layers using Convolutional Neural Networks (CNNs) was successfully achieved. Many experiments were carried out using various CNN model enhancement techniques. The CNN model was able to record above 70% accuracy at the last experiment.

Reference

[1] Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A Review of Yolo algorithm developments. *Procedia computer science*, *199*, 1066-1073.

APPENDIX

Section 1 Code:

https://colab.research.google.com/drive/1vysIQY9L-Z-L4wuZceKH7eDBYUA0oHmC#scrollTo=Wo0PChE3iLDi

Section 2 Code:

https://colab.research.google.com/drive/1xZEJu_hfpmAOkkni66WWzsu8vamMsSYm#scrollTo=0CMF1DOg6RTe