Project Topic

Our topic for the term project is Image Classification.

Group Members

Group 5 has two members:

- Chinmay Sharma (300157594)
- Sahibdeep Singh (300156800)

Goals/Results

The model is going to be able to classify the input image according to 10 classes and will provide high accuracy in classified images.

Libraries/Tools

We are using tools like TensorFlow, Keras and Convolutional Neural Networks for image classification. Up till now we have used NumPy for data processing and Matplotlib for data visualization. Along with these, we are planning on using Scikit-learn and Pandas for data processing.

Dataset

For our dataset we are using CIFAR-10, which consists of 60,000 32 * 32 colour images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test images.

Algorithms/Models

Our project up till now trains a convolutional neural network to perform multi-class classification on images at random. Right now, the average accuracy of our model is about 70% but we hope to create and fine-tune the model ourselves to provide results with higher accuracy.

Screenshots

These are all the screenshots of the results we obtained:

Image 1:

Image 2:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
dropout (Dropout)            (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 512)               4194816
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                5130
=================================================================
Total params: 4,210,090
Trainable params: 4,210,090
Non-trainable params: 0
_________________________________________________________________
```

Image 3:

```
Epoch 1/10
1094/1094 [==============================] - 7s 6ms/step - loss: 1.8091 - accuracy: 0.3464 - val_loss: 1.4971 - val_accuracy: 0.4559
Epoch 2/10
1094/1094 [==============================] - 6s 6ms/step - loss: 1.4337 - accuracy: 0.4838 - val_loss: 1.3936 - val_accuracy: 0.5005
Epoch 3/10
1094/1094 [==============================] - 6s 6ms/step - loss: 1.2581 - accuracy: 0.5506 - val_loss: 1.1977 - val_accuracy: 0.5759
Epoch 4/10
1094/1094 [==============================] - 6s 6ms/step - loss: 1.1255 - accuracy: 0.5945 - val_loss: 1.1306 - val_accuracy: 0.6004
Epoch 5/10
1094/1094 [==============================] - 6s 6ms/step - loss: 1.0202 - accuracy: 0.6379 - val_loss: 1.0554 - val_accuracy: 0.6250
Epoch 6/10
1094/1094 [==============================] - 6s 6ms/step - loss: 0.9299 - accuracy: 0.6690 - val_loss: 1.0391 - val_accuracy: 0.6375
Epoch 7/10
1094/1094 [==============================] - 6s 6ms/step - loss: 0.8452 - accuracy: 0.7010 - val_loss: 1.0075 - val_accuracy: 0.6468
Epoch 8/10
1094/1094 [==============================] - 6s 6ms/step - loss: 0.7716 - accuracy: 0.7289 - val_loss: 0.9945 - val_accuracy: 0.6545
Epoch 9/10
1094/1094 [==============================] - 6s 6ms/step - loss: 0.7021 - accuracy: 0.7496 - val_loss: 0.9965 - val_accuracy: 0.6561
Epoch 10/10
1094/1094 [==============================] - 6s 6ms/step - loss: 0.6372 - accuracy: 0.7746 - val_loss: 0.9758 - val_accuracy: 0.6711
```

Image 4:

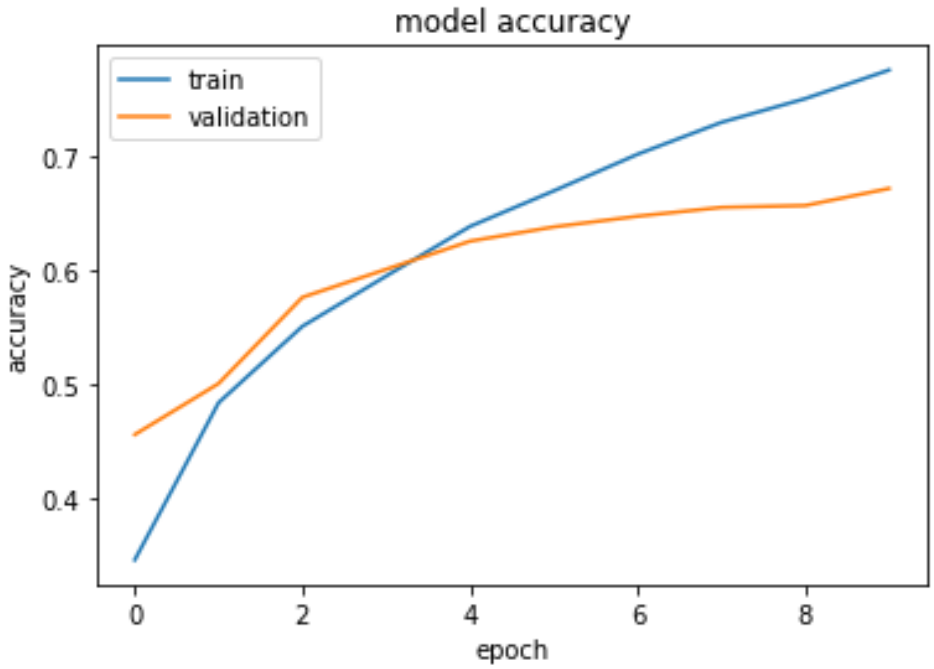


Image 5:

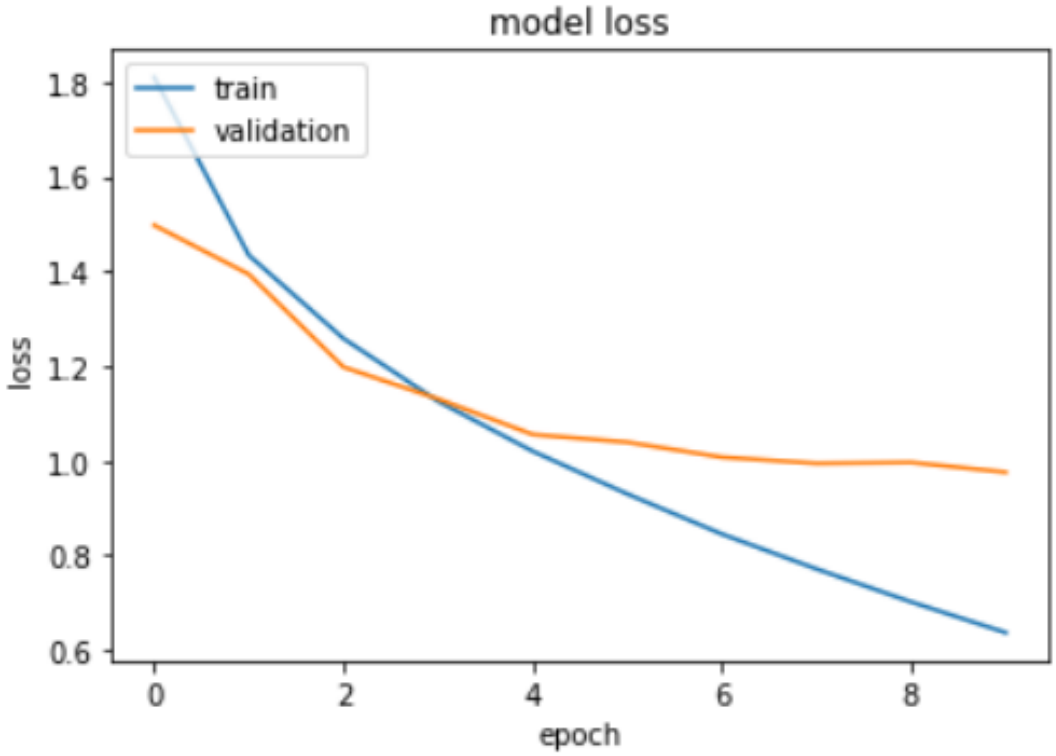


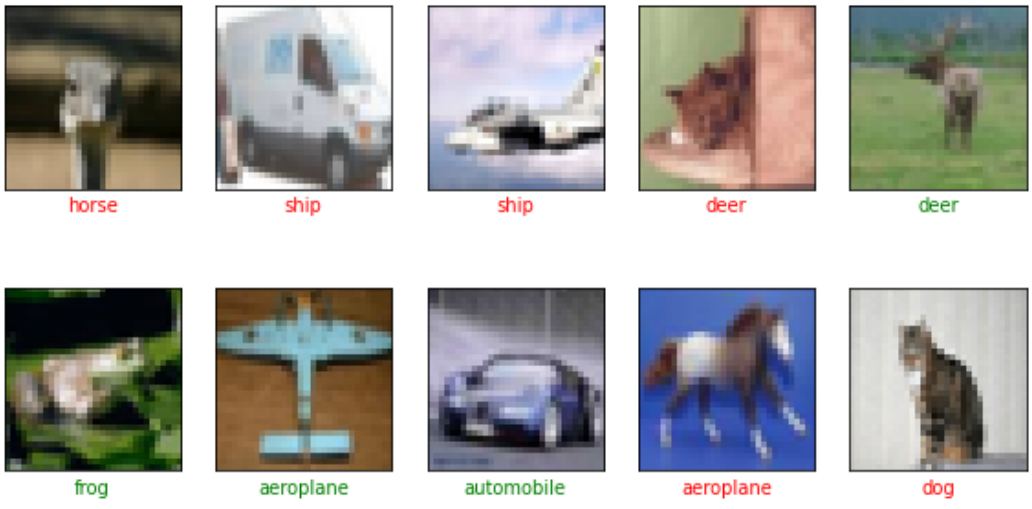
Image 6:

```
[15] #Step 4: Evaluation
     loss, accuracy_= model.evaluate(X_Test, Y_Test)
     print("Accuracy of this model is: ", accuracy_*100)
```

```
313/313 [==============================] - 1s 3ms/step - loss: 0.9857 - accuracy: 0.6590
Accuracy of this model is:  65.89999794960022
```

Image 7:

```
[19] show_random_examples(X_Test,Y_Test,preds)
```



Questions Answered

Here are all the typical questions we have answered by now, using our code. We have also tried to explain how our code works in all the answers to these questions below.

1. How can you build models on this dataset?

Answer 1: CIFAR10 (Canadian Institute for Advanced Research) is the standard image classification dataset. It consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dimension of each image is 32x32x3 as each color image contains 3 dimensions of RGB and we use CNNs which employs filters to extract only the information required for classification from the image and uses it for that purpose.

2. Where would be the source of raw dataset

Answer 2: The CIFAR-10 and CIFAR-100 are labeled subsets of the 80 million tiny images dataset. They were collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton.

The raw dataset can be found on: https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz

3. How to detect outliers

Answer 3: One of the most common outliers we found were misclassification between animals like cat/dog/horse. We believe that this may be because our CNN is not much deep, and we cannot detect the dividing features between these animals.

4. **How can you analyze data? what tools or APIs you used and which ones works the best for you (compare them in your report)**

Answer 4: For data analysis, we have used NumPy and Matplotlib. We also tried using Pandas or seaborn, but we found that NumPy and Matplotlib were a lot easier to use and we were able to find support for our obstacles easier.

5. **How did you train the system (if applicable)?**

Answer 5: We tried using educated guesses for tuning our neural net and we also tried to use advanced optimizer algorithms like Stochastic Gradient Descent (SGD) to fine tune our network. Using tools like TensorFlow and Keras made the job a lot easier and we were able to quickly iterate to find the current optimal values.

Future Goals

Our future plans involve that:

- We will make a simple visualization web app using Streamlit.
- The web app will take an image as input from the user and will give its prediction to classify the image.
- We will also try to base our network on previously made networks like ResNet and try to achieve a similar accuracy but with less layers.

References

Chansung, P. (2018, June 15). CIFAR-10 Image Classification in TensorFlow. Retrieved November 13, 2020, from https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c

CIFAR-10 - Keras. (n.d.). Retrieved November 13, 2020, from http://home.mit.bme.hu/~hadhazi/Oktatas/NN18/dem3/html_demo/CIFAR-10Demo.html

Convolutional Neural Network (CNN): TensorFlow Core. (n.d.). Retrieved November 13, 2020, from https://www.tensorflow.org/tutorials/images/cnn

Image classification: TensorFlow Core. (n.d.). Retrieved November 13, 2020, from https://www.tensorflow.org/tutorials/images/classification

Leskovec, J., Rajaraman, A., & Ullman, J. D. (2020). *Mining of massive datasets* (3rd ed.) [3rd edition]. New York, NY: Cambridge University Press. Retrieved November 13, 2020, from http://www.mmds.org/#ver30