Image Classification Using CNN

CS 6301: Practical Aspects of Data Science

Summer 2020: Assignment 2b

Team Member: -
Rajarshi Chattopadhyay (RXC170010)

Introduction

This project involves using Convolutional Neural Network for image classification.

A TensorFlow technique for image classification has been used in with the help of the Keras API of TensorFlow in Python.

Data

CIFAR-10 - Object Recognition in Images http://www.cs.utoronto.ca/~kriz/cifar.html
The data is also stored in https://www.utdallas.edu/~rxc170010/cifar-10-batches-py/

The CIFAR-10 is labeled subset of the 80 million tiny images dataset, consisting of 60000 32x32 color images in 10 classes, with 6000 images per class.

The data, which comes as vectors, has been reshaped to 32x32x3, indicating a 32x32 image with 3 channels for RGB. This is necessary for using as input for the 2D Convolutional layers of the model.

Training examples 45000, Testing examples 5000

Models

Models were created with different architectures

- The number of neurons in each kernel and the number of hidden layers were modified to test and improve the model.
- Row and column size from input to hidden layer has been kept the same.
- Batch normalization was used in the input layer to ensure there is not much covariance shift from its output.
- Max-pooling was used in the hidden Conv layer to reduce row and column dimensions to half.
- Dropout was used to inactivate randomly 50% of the nodes in Conv layers and 40% in Dense layers for each data example at the beginning. Then changed later.

Library used: Keras https://www.tensorflow.org/guide/keras/sequential_model

Training

The models were trained and tuned to find the best classifier

- Validation was done during training with 33% of the training data
- Batch size of 32 was used
- Number of epochs were changed to improve the model
- Early stopping was done to stop training if there is no significant improvement in validation accuracy after every 3 epochs.
- Checkpointing has been performed to save the best performing model for later use during prediction.

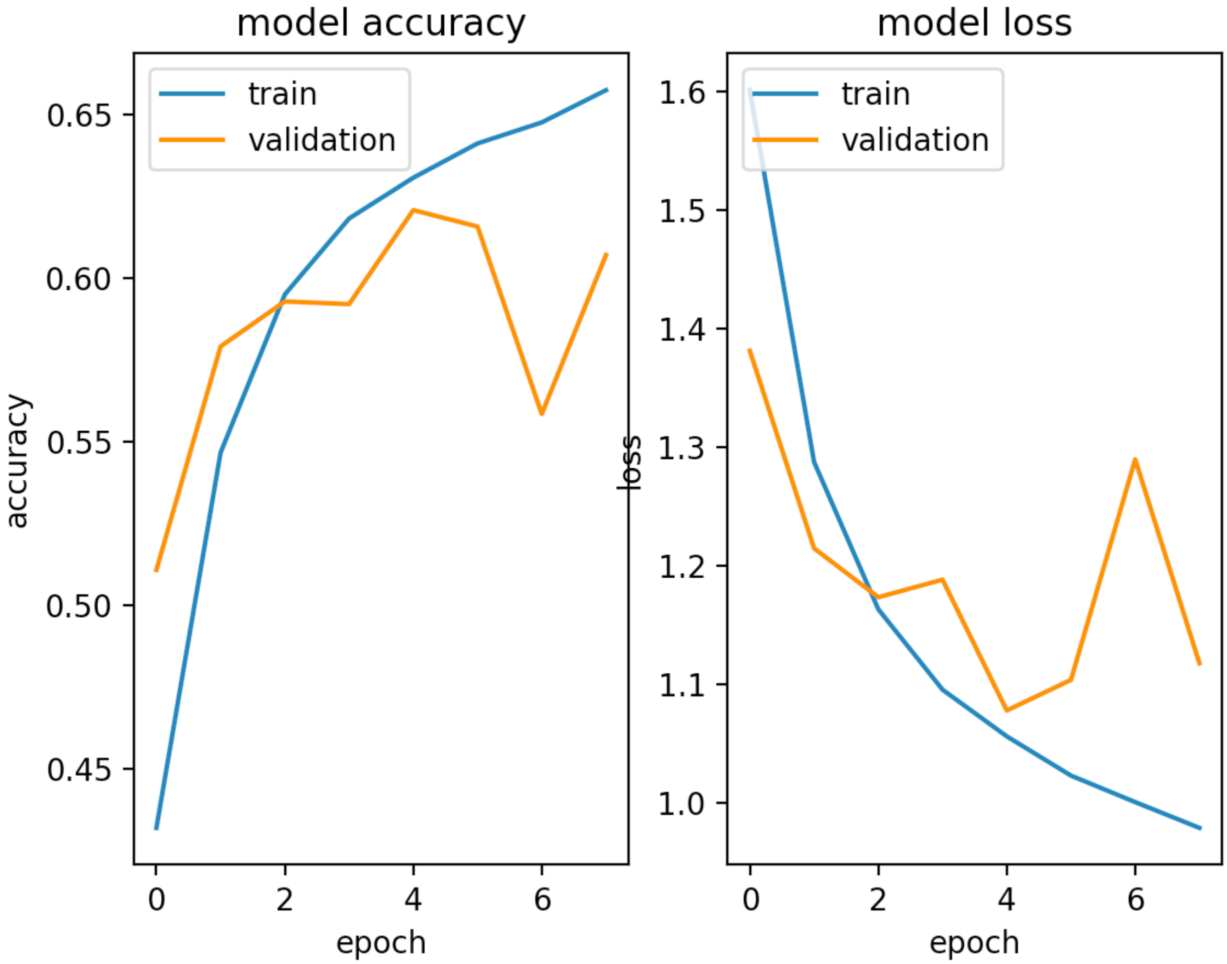
Testing

Each model was tested with test data to obtain the test accuracy and loss.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 32)        128
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 16)        4624
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 16)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 16)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3600)              0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                36010
=================================================================
Total params: 41,658
Trainable params: 41,594
Non-trainable params: 64
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/10
30149/30149 [==============================] - 59s 2ms/step - loss: 1.6011 - accuracy: 0.4320 - val_loss: 1.3811 - val_accuracy: 0.5109
Epoch 2/10
30149/30149 [==============================] - 63s 2ms/step - loss: 1.2873 - accuracy: 0.5468 - val_loss: 1.2145 - val_accuracy: 0.5792
Epoch 3/10
30149/30149 [==============================] - 53s 2ms/step - loss: 1.1631 - accuracy: 0.5951 - val_loss: 1.1732 - val_accuracy: 0.5930
Epoch 4/10
30149/30149 [==============================] - 52s 2ms/step - loss: 1.0952 - accuracy: 0.6183 - val_loss: 1.1880 - val_accuracy: 0.5921
Epoch 5/10
30149/30149 [==============================] - 51s 2ms/step - loss: 1.0558 - accuracy: 0.6308 - val_loss: 1.0777 - val_accuracy: 0.6209
Epoch 6/10
30149/30149 [==============================] - 52s 2ms/step - loss: 1.0227 - accuracy: 0.6413 - val_loss: 1.1034 - val_accuracy: 0.6159
Epoch 7/10
30149/30149 [==============================] - 51s 2ms/step - loss: 1.0003 - accuracy: 0.6477 - val_loss: 1.2895 - val_accuracy: 0.5585
Epoch 8/10
```

```
30149/30149 [==============================] - 56s 2ms/step - loss: 0.9786 - accuracy:
0.6576 - val_loss: 1.1175 - val_accuracy: 0.6072
5000/5000 [==============================] - 3s 515us/step
```

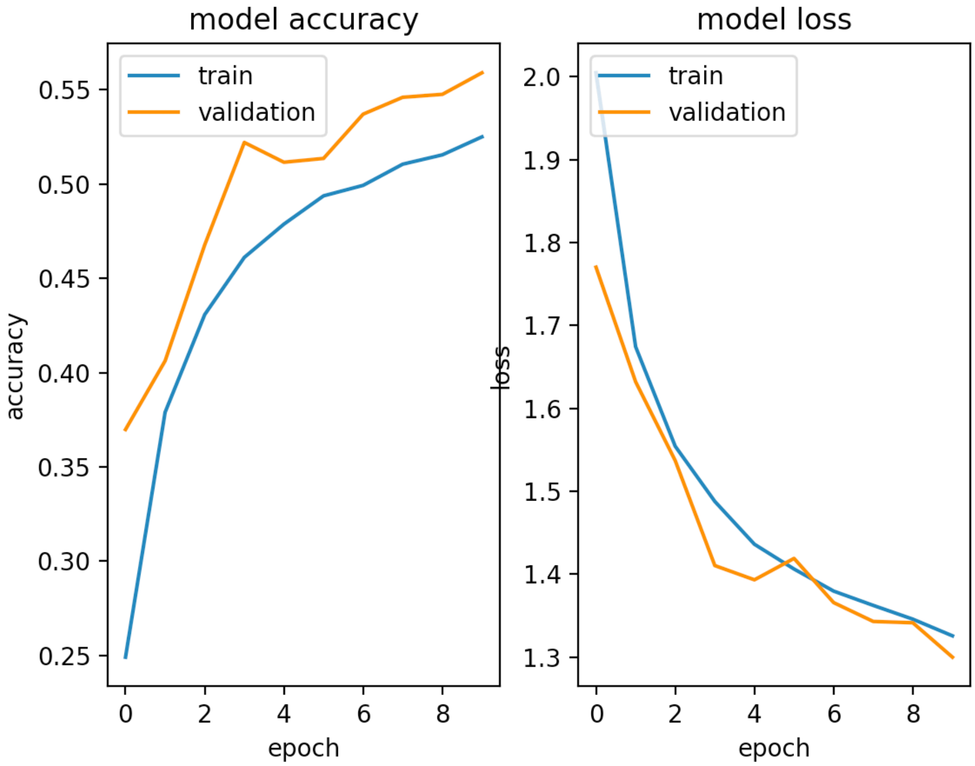
- Train result: train_loss: 0.9786 - train_accuracy: 65.7567
- Validation result: val_loss: 1.1175 - val_accuracy: 60.7165
- Test result: test_loss: 1.0774 - test_accuracy: 62.1800



```
Model: "sequential_2"
_________________________________________________________________
Layer (type)             Output Shape           Param #
=================================================================
conv2d_1 (Conv2D)           (None, 32, 32, 32)       896
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 32)       128
_________________________________________________________________
conv2d_2 (Conv2D)           (None, 30, 30, 16)       4624
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 16)       0
_________________________________________________________________
dropout_1 (Dropout)         (None, 15, 15, 16)       0
_________________________________________________________________
conv2d_3 (Conv2D)           (None, 13, 13, 8)        1160
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 8)          0
_________________________________________________________________
dropout_2 (Dropout)         (None, 6, 6, 8)          0
_________________________________________________________________
flatten_1 (Flatten)         (None, 288)              0
_________________________________________________________________
dense_1 (Dense)             (None, 10)               2890
=================================================================
Total params: 9,698
Trainable params: 9,634
Non-trainable params: 64
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/10
30149/30149 [==============================] - 63s 2ms/step - loss: 2.0049 - accuracy:
0.2490 - val_loss: 1.7704 - val_accuracy: 0.3698
Epoch 2/10
30149/30149 [==============================] - 58s 2ms/step - loss: 1.6745 - accuracy:
0.3790 - val_loss: 1.6322 - val_accuracy: 0.4062
Epoch 3/10
30149/30149 [==============================] - 57s 2ms/step - loss: 1.5544 - accuracy:
0.4308 - val_loss: 1.5371 - val_accuracy: 0.4678
Epoch 4/10
30149/30149 [==============================] - 58s 2ms/step - loss: 1.4878 - accuracy:
0.4611 - val_loss: 1.4105 - val_accuracy: 0.5221
Epoch 5/10
30149/30149 [==============================] - 60s 2ms/step - loss: 1.4362 - accuracy:
0.4788 - val_loss: 1.3933 - val_accuracy: 0.5116
Epoch 6/10
```

```
30149/30149 [==============================] - 59s 2ms/step - loss: 1.4063 - accuracy: 0.4938 - val_loss: 1.4191 - val_accuracy: 0.5136
Epoch 7/10
30149/30149 [==============================] - 61s 2ms/step - loss: 1.3797 - accuracy: 0.4994 - val_loss: 1.3658 - val_accuracy: 0.5371
Epoch 8/10
30149/30149 [==============================] - 60s 2ms/step - loss: 1.3624 - accuracy: 0.5106 - val_loss: 1.3430 - val_accuracy: 0.5461
Epoch 9/10
30149/30149 [==============================] - 72s 2ms/step - loss: 1.3458 - accuracy: 0.5155 - val_loss: 1.3414 - val_accuracy: 0.5476
Epoch 10/10
30149/30149 [==============================] - 66s 2ms/step - loss: 1.3257 - accuracy: 0.5251 - val_loss: 1.3000 - val_accuracy: 0.5591
5000/5000 [==============================] - 2s 461us/step
```

- Train result: train_loss: 1.3257 - train_accuracy: 52.5092
- Validation result: val_loss: 1.3000 - val_accuracy: 55.9087
- Test result: test_loss: 1.2767 - test_accuracy: 56.8000



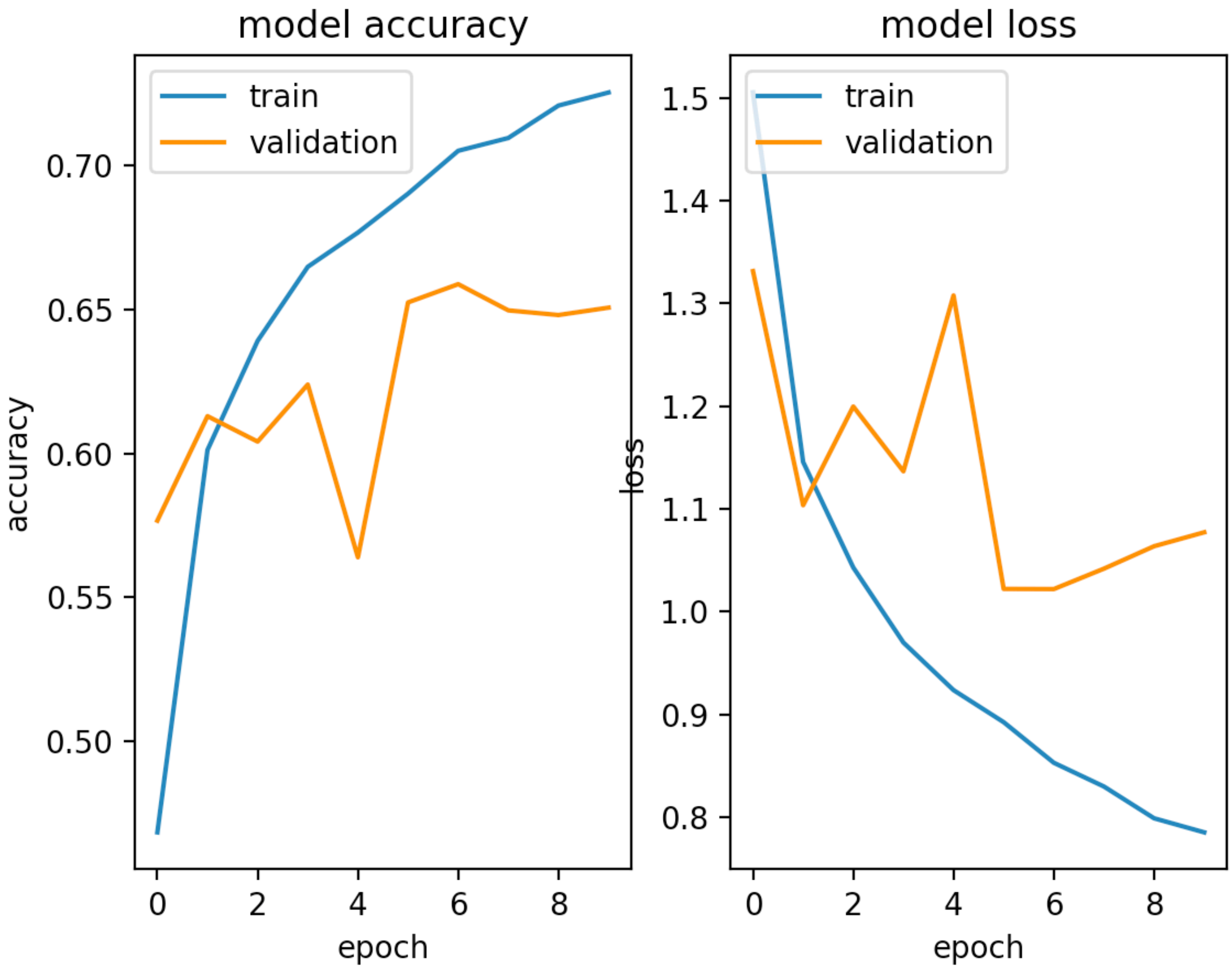
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 64)        1792
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 64)        256
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 32)        18464
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 13, 13, 8)         2312
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 8)           0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 8)           0
_________________________________________________________________
flatten_1 (Flatten)          (None, 288)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2890
=================================================================
Total params: 25,714
Trainable params: 25,586
Non-trainable params: 128
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 93s 3ms/step - loss: 2.0687 - accuracy:
0.2219 - val_loss: 1.8164 - val_accuracy: 0.3779
Epoch 2/20
30149/30149 [==============================] - 62s 2ms/step - loss: 2.0563 - accuracy:
0.2337 - val_loss: 1.8974 - val_accuracy: 0.3553
Epoch 3/20
30149/30149 [==============================] - 57s 2ms/step - loss: 1.6948 - accuracy:
0.3720 - val_loss: 1.6212 - val_accuracy: 0.4274
```

Not much accuracy increase, so forcefully interrupted training.

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)             Output Shape           Param #
=================================================================
conv2d_1 (Conv2D)           (None, 32, 32, 64)       1792
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 64)       256
_________________________________________________________________
conv2d_2 (Conv2D)           (None, 30, 30, 32)       18464
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)       0
_________________________________________________________________
dropout_1 (Dropout)         (None, 15, 15, 32)       0
_________________________________________________________________
flatten_1 (Flatten)         (None, 7200)             0
_________________________________________________________________
dense_1 (Dense)             (None, 10)               72010
=================================================================
Total params: 92,522
Trainable params: 92,394
Non-trainable params: 128
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 95s 3ms/step - loss: 1.5054 - accuracy:
0.4685 - val_loss: 1.3315 - val_accuracy: 0.5767
Epoch 2/20
30149/30149 [==============================] - 92s 3ms/step - loss: 1.1457 - accuracy:
0.6012 - val_loss: 1.1035 - val_accuracy: 0.6129
Epoch 3/20
30149/30149 [==============================] - 91s 3ms/step - loss: 1.0429 - accuracy:
0.6390 - val_loss: 1.1997 - val_accuracy: 0.6041
Epoch 4/20
30149/30149 [==============================] - 90s 3ms/step - loss: 0.9699 - accuracy:
0.6647 - val_loss: 1.1366 - val_accuracy: 0.6239
Epoch 5/20
30149/30149 [==============================] - 103s 3ms/step - loss: 0.9235 - accuracy:
0.6766 - val_loss: 1.3078 - val_accuracy: 0.5639
Epoch 6/20
30149/30149 [==============================] - 100s 3ms/step - loss: 0.8923 - accuracy:
0.6901 - val_loss: 1.0221 - val_accuracy: 0.6523
Epoch 7/20
30149/30149 [==============================] - 93s 3ms/step - loss: 0.8528 - accuracy:
0.7049 - val_loss: 1.0219 - val_accuracy: 0.6587
Epoch 8/20
```

```
30149/30149 [==============================] - 108s 4ms/step - loss: 0.8297 - accuracy:
0.7094 - val_loss: 1.0418 - val_accuracy: 0.6496
Epoch 9/20
30149/30149 [==============================] - 96s 3ms/step - loss: 0.7988 - accuracy:
0.7206 - val_loss: 1.0636 - val_accuracy: 0.6480
Epoch 10/20
30149/30149 [==============================] - 94s 3ms/step - loss: 0.7851 - accuracy:
0.7252 - val_loss: 1.0772 - val_accuracy: 0.6505
5000/5000 [==============================] - 5s 923us/step
```

- Train result: train_loss: 0.7851 - train_accuracy: 72.5165
- Validation result: val_loss: 1.0772 - val_accuracy: 65.0529
- Test result: test_loss: 1.0292 - test_accuracy: 67.7200

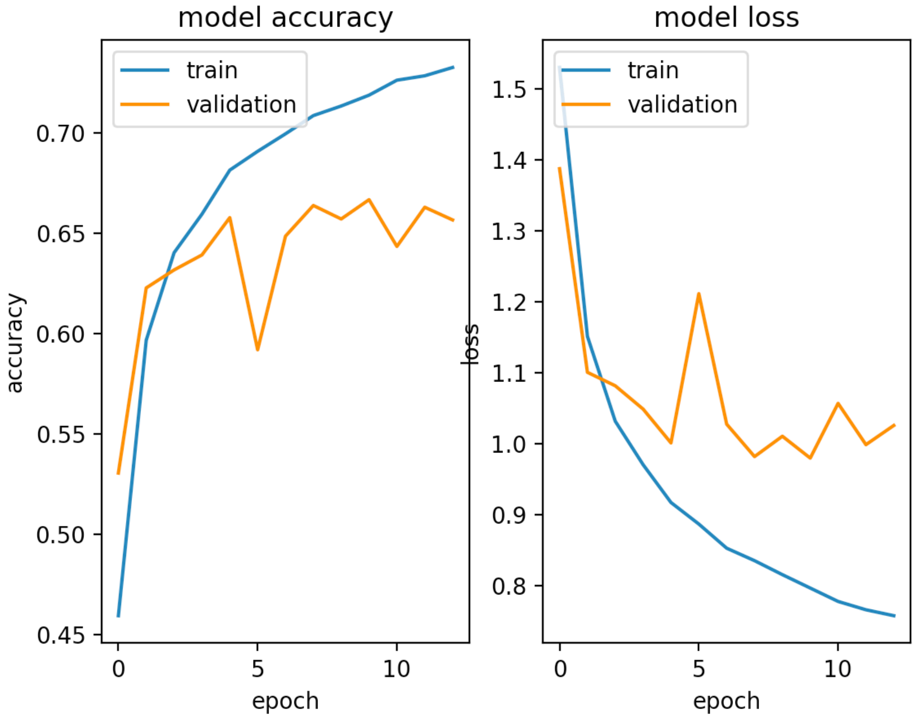


```
Model: "sequential_5"
_________________________________________________________________
```

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 16)        448
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 16)        64
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 7200)              0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                72010
=================================================================
Total params: 77,162
Trainable params: 77,130
Non-trainable params: 32
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 48s 2ms/step - loss: 1.5301 - accuracy:
0.4595 - val_loss: 1.3875 - val_accuracy: 0.5305
Epoch 2/20
30149/30149 [==============================] - 44s 1ms/step - loss: 1.1513 - accuracy:
0.5968 - val_loss: 1.1007 - val_accuracy: 0.6228
Epoch 3/20
30149/30149 [==============================] - 42s 1ms/step - loss: 1.0318 - accuracy:
0.6403 - val_loss: 1.0817 - val_accuracy: 0.6317
Epoch 4/20
30149/30149 [==============================] - 42s 1ms/step - loss: 0.9705 - accuracy:
0.6595 - val_loss: 1.0489 - val_accuracy: 0.6392
Epoch 5/20
30149/30149 [==============================] - 52s 2ms/step - loss: 0.9173 - accuracy:
0.6815 - val_loss: 1.0012 - val_accuracy: 0.6578
Epoch 6/20
30149/30149 [==============================] - 42s 1ms/step - loss: 0.8869 - accuracy:
0.6908 - val_loss: 1.2114 - val_accuracy: 0.5919
Epoch 7/20
30149/30149 [==============================] - 48s 2ms/step - loss: 0.8529 - accuracy:
0.6996 - val_loss: 1.0276 - val_accuracy: 0.6486
Epoch 8/20
30149/30149 [==============================] - 48s 2ms/step - loss: 0.8353 - accuracy:
0.7086 - val_loss: 0.9819 - val_accuracy: 0.6639
Epoch 9/20
```

```
30149/30149 [==============================] - 44s 1ms/step - loss: 0.8156 - accuracy:
0.7134 - val_loss: 1.0106 - val_accuracy: 0.6571
Epoch 10/20
30149/30149 [==============================] - 45s 1ms/step - loss: 0.7968 - accuracy:
0.7188 - val_loss: 0.9798 - val_accuracy: 0.6668
Epoch 11/20
30149/30149 [==============================] - 44s 1ms/step - loss: 0.7779 - accuracy:
0.7263 - val_loss: 1.0569 - val_accuracy: 0.6435
Epoch 12/20
30149/30149 [==============================] - 44s 1ms/step - loss: 0.7662 - accuracy:
0.7285 - val_loss: 0.9988 - val_accuracy: 0.6630
Epoch 13/20
30149/30149 [==============================] - 47s 2ms/step - loss: 0.7578 - accuracy:
0.7326 - val_loss: 1.0258 - val_accuracy: 0.6567
5000/5000 [==============================] - 2s 375us/step
```

- Train result: train_loss: 0.7578 - train_accuracy: 73.2628
- Validation result: val_loss: 1.0258 - val_accuracy: 65.6656
- Test result: test_loss: 1.0088 - test_accuracy: 67.2400

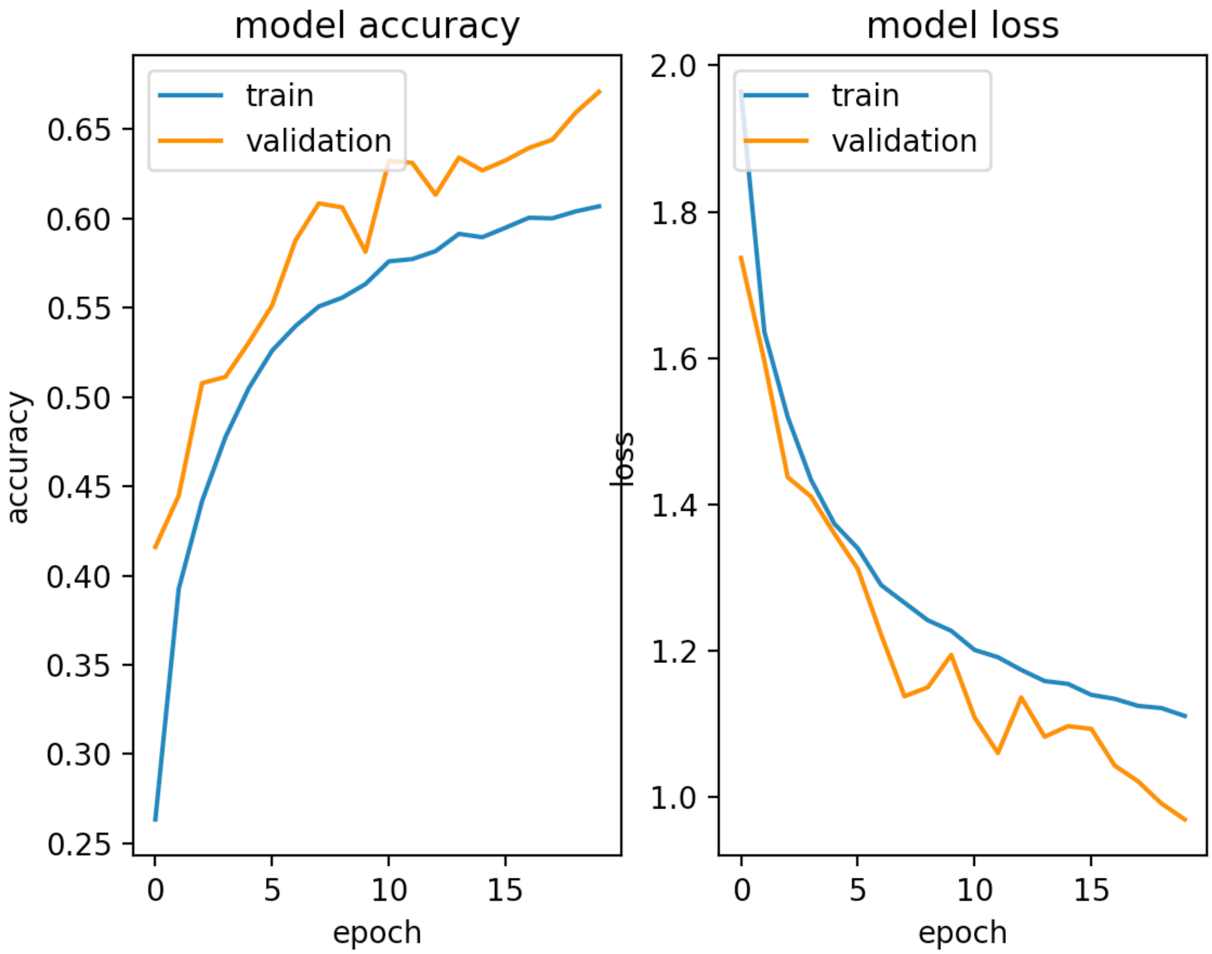


```
Model: "sequential_6"
_________________________________________________________________
Layer (type)              Output Shape           Param #
=================================================================
conv2d_1 (Conv2D)          (None, 32, 32, 16)       448
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 16)       64
_________________________________________________________________
conv2d_2 (Conv2D)          (None, 30, 30, 32)       4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)       0
_________________________________________________________________
dropout_1 (Dropout)         (None, 15, 15, 32)       0
_________________________________________________________________
conv2d_3 (Conv2D)          (None, 13, 13, 16)       4624
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 16)         0
_________________________________________________________________
dropout_2 (Dropout)         (None, 6, 6, 16)         0
_________________________________________________________________
flatten_1 (Flatten)         (None, 576)              0
_________________________________________________________________
dense_1 (Dense)            (None, 100)              57700
_________________________________________________________________
dropout_3 (Dropout)         (None, 100)              0
_________________________________________________________________
dense_2 (Dense)            (None, 10)               1010
=================================================================
Total params: 68,486
Trainable params: 68,454
Non-trainable params: 32
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 51s 2ms/step - loss: 1.9638 - accuracy:
0.2634 - val_loss: 1.7368 - val_accuracy: 0.4160
Epoch 2/20
30149/30149 [==============================] - 57s 2ms/step - loss: 1.6356 - accuracy:
0.3926 - val_loss: 1.5957 - val_accuracy: 0.4448
Epoch 3/20
30149/30149 [==============================] - 48s 2ms/step - loss: 1.5193 - accuracy:
0.4417 - val_loss: 1.4372 - val_accuracy: 0.5078
Epoch 4/20
30149/30149 [==============================] - 49s 2ms/step - loss: 1.4334 - accuracy:
0.4775 - val_loss: 1.4105 - val_accuracy: 0.5113
Epoch 5/20
```

```
30149/30149 [==============================] - 48s 2ms/step - loss: 1.3734 - accuracy:
0.5049 - val_loss: 1.3598 - val_accuracy: 0.5305
Epoch 6/20
30149/30149 [==============================] - 57s 2ms/step - loss: 1.3400 - accuracy:
0.5260 - val_loss: 1.3120 - val_accuracy: 0.5513
Epoch 7/20
30149/30149 [==============================] - 45s 1ms/step - loss: 1.2897 - accuracy:
0.5397 - val_loss: 1.2218 - val_accuracy: 0.5877
Epoch 8/20
30149/30149 [==============================] - 44s 1ms/step - loss: 1.2656 - accuracy:
0.5507 - val_loss: 1.1377 - val_accuracy: 0.6084
Epoch 9/20
30149/30149 [==============================] - 47s 2ms/step - loss: 1.2416 - accuracy:
0.5556 - val_loss: 1.1500 - val_accuracy: 0.6062
Epoch 10/20
30149/30149 [==============================] - 44s 1ms/step - loss: 1.2273 - accuracy:
0.5633 - val_loss: 1.1943 - val_accuracy: 0.5814
Epoch 11/20
30149/30149 [==============================] - 44s 1ms/step - loss: 1.2011 - accuracy:
0.5760 - val_loss: 1.1089 - val_accuracy: 0.6322
Epoch 12/20
30149/30149 [==============================] - 45s 2ms/step - loss: 1.1911 - accuracy:
0.5772 - val_loss: 1.0602 - val_accuracy: 0.6311
Epoch 13/20
30149/30149 [==============================] - 46s 2ms/step - loss: 1.1738 - accuracy:
0.5817 - val_loss: 1.1360 - val_accuracy: 0.6133
Epoch 14/20
30149/30149 [==============================] - 49s 2ms/step - loss: 1.1586 - accuracy:
0.5914 - val_loss: 1.0824 - val_accuracy: 0.6341
Epoch 15/20
30149/30149 [==============================] - 47s 2ms/step - loss: 1.1547 - accuracy:
0.5895 - val_loss: 1.0969 - val_accuracy: 0.6269
Epoch 16/20
30149/30149 [==============================] - 47s 2ms/step - loss: 1.1397 - accuracy:
0.5948 - val_loss: 1.0932 - val_accuracy: 0.6326
Epoch 17/20
30149/30149 [==============================] - 45s 1ms/step - loss: 1.1342 - accuracy:
0.6004 - val_loss: 1.0431 - val_accuracy: 0.6395
Epoch 18/20
30149/30149 [==============================] - 45s 1ms/step - loss: 1.1246 - accuracy:
0.6001 - val_loss: 1.0215 - val_accuracy: 0.6441
Epoch 19/20
30149/30149 [==============================] - 47s 2ms/step - loss: 1.1217 - accuracy:
0.6040 - val_loss: 0.9914 - val_accuracy: 0.6593
Epoch 20/20
```

```
30149/30149 [==============================] - 51s 2ms/step - loss: 1.1109 - accuracy:
0.6068 - val_loss: 0.9694 - val_accuracy: 0.6709
```

- Train result: train_loss: 1.1109 - train_accuracy: 60.6786
- Validation result: val_loss: 0.9694 - val_accuracy: 67.0931
- Test result: test_loss: 0.9389 - test_accuracy: 68.8400

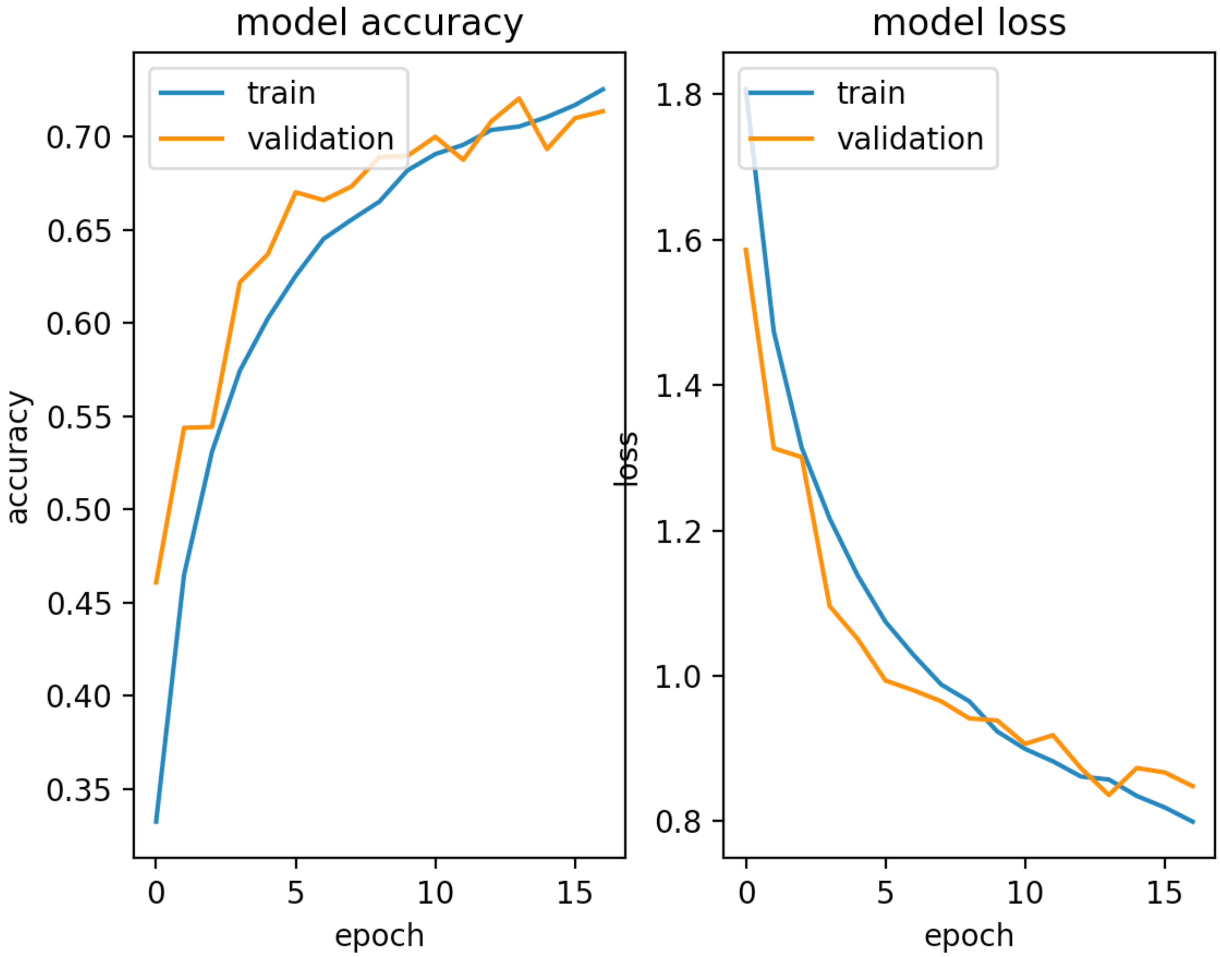


```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 16)        448
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 16)        64
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_1 (Dense)              (None, 100)               230500
_________________________________________________________________
dropout_3 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_2 (Dense)              (None, 100)               10100
_________________________________________________________________
dropout_4 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                1010
=================================================================
Total params: 265,258
Trainable params: 265,226
Non-trainable params: 32
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 56s 2ms/step - loss: 1.8070 - accuracy:
0.3323 - val_loss: 1.5862 - val_accuracy: 0.4607
Epoch 2/20
30149/30149 [==============================] - 58s 2ms/step - loss: 1.4738 - accuracy:
0.4647 - val_loss: 1.3131 - val_accuracy: 0.5437
Epoch 3/20
30149/30149 [==============================] - 51s 2ms/step - loss: 1.3139 - accuracy:
0.5307 - val_loss: 1.3009 - val_accuracy: 0.5441
```

```
Epoch 4/20
30149/30149 [==============================] - 53s 2ms/step - loss: 1.2165 - accuracy:
0.5743 - val_loss: 1.0960 - val_accuracy: 0.6217
Epoch 5/20
30149/30149 [==============================] - 74s 2ms/step - loss: 1.1384 - accuracy:
0.6023 - val_loss: 1.0509 - val_accuracy: 0.6368
Epoch 6/20
30149/30149 [==============================] - 56s 2ms/step - loss: 1.0744 - accuracy:
0.6253 - val_loss: 0.9934 - val_accuracy: 0.6701
Epoch 7/20
30149/30149 [==============================] - 54s 2ms/step - loss: 1.0289 - accuracy:
0.6451 - val_loss: 0.9802 - val_accuracy: 0.6659
Epoch 8/20
30149/30149 [==============================] - 52s 2ms/step - loss: 0.9878 - accuracy:
0.6554 - val_loss: 0.9649 - val_accuracy: 0.6732
Epoch 9/20
30149/30149 [==============================] - 53s 2ms/step - loss: 0.9650 - accuracy:
0.6651 - val_loss: 0.9414 - val_accuracy: 0.6890
Epoch 10/20
30149/30149 [==============================] - 54s 2ms/step - loss: 0.9234 - accuracy:
0.6818 - val_loss: 0.9386 - val_accuracy: 0.6896
Epoch 11/20
30149/30149 [==============================] - 51s 2ms/step - loss: 0.8994 - accuracy:
0.6906 - val_loss: 0.9064 - val_accuracy: 0.6999
Epoch 12/20
30149/30149 [==============================] - 59s 2ms/step - loss: 0.8823 - accuracy:
0.6955 - val_loss: 0.9184 - val_accuracy: 0.6875
Epoch 13/20
30149/30149 [==============================] - 51s 2ms/step - loss: 0.8614 - accuracy:
0.7035 - val_loss: 0.8729 - val_accuracy: 0.7081
Epoch 14/20
30149/30149 [==============================] - 51s 2ms/step - loss: 0.8573 - accuracy:
0.7054 - val_loss: 0.8359 - val_accuracy: 0.7205
Epoch 15/20
30149/30149 [==============================] - 52s 2ms/step - loss: 0.8347 - accuracy:
0.7105 - val_loss: 0.8732 - val_accuracy: 0.6933
Epoch 16/20
30149/30149 [==============================] - 56s 2ms/step - loss: 0.8189 - accuracy:
0.7169 - val_loss: 0.8671 - val_accuracy: 0.7099
Epoch 17/20
30149/30149 [==============================] - 52s 2ms/step - loss: 0.7993 - accuracy:
0.7253 - val_loss: 0.8482 - val_accuracy: 0.7136
5000/5000 [==============================] - 2s 400us/step
```

- Train result: train_loss: 0.7993 - train_accuracy: 72.5331
- Validation result: val_loss: 0.8482 - val_accuracy: 71.3555

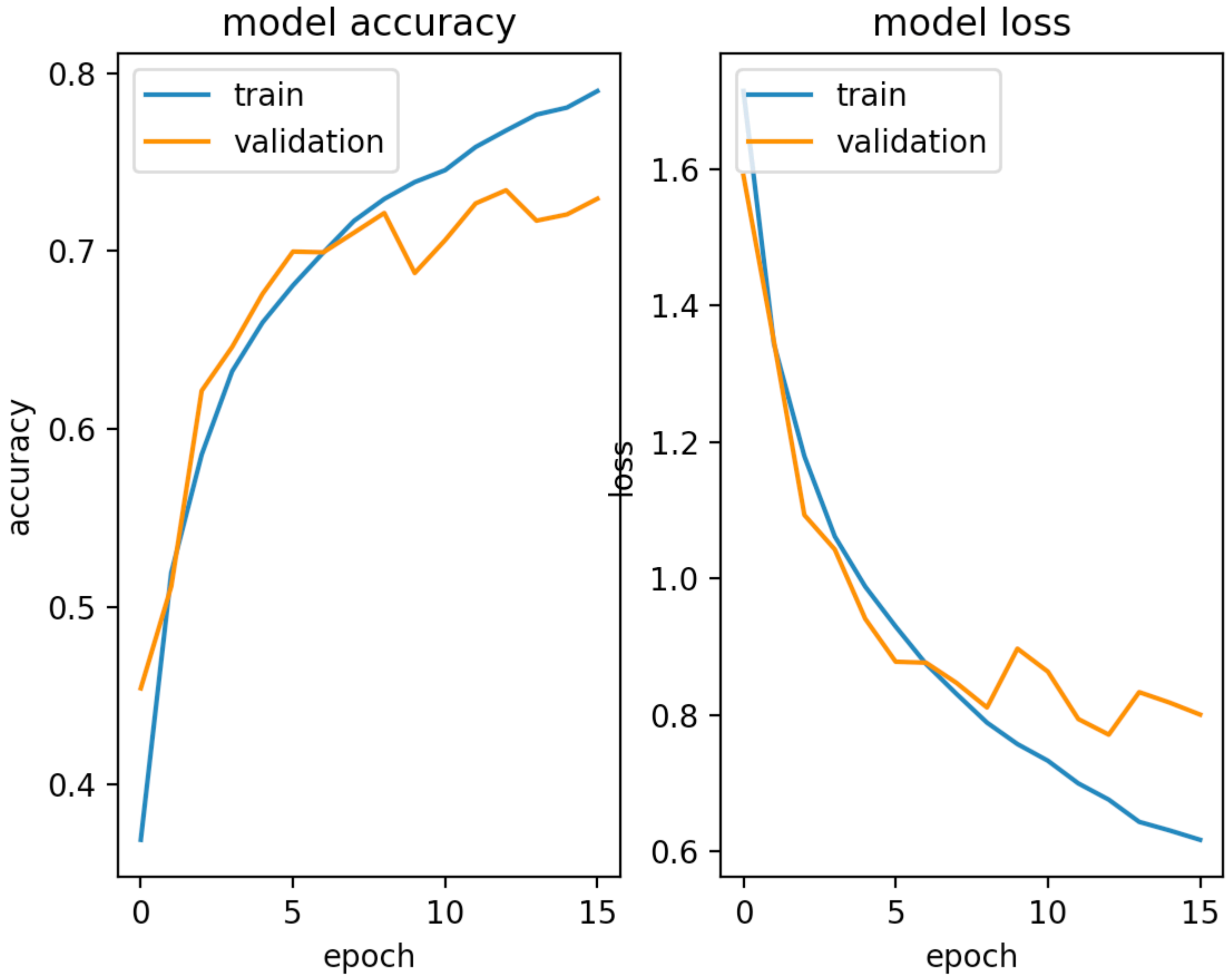
- Test result: test_loss: 0.8135 - test_accuracy: 72.6800



```
Model: "sequential_8"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 32)        128
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 13, 13, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 4608)              0
_________________________________________________________________
dense_1 (Dense)              (None, 200)               921800
_________________________________________________________________
dropout_3 (Dropout)          (None, 200)               0
_________________________________________________________________
dense_2 (Dense)              (None, 100)               20100
_________________________________________________________________
dropout_4 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                1010
=================================================================
Total params: 1,036,286
Trainable params: 1,036,222
Non-trainable params: 64
_________________________________________________________________
Train on 30149 samples, validate on 14851 samples
Epoch 1/20
30149/30149 [==============================] - 133s 4ms/step - loss: 1.7147 - accuracy:
0.3689 - val_loss: 1.5906 - val_accuracy: 0.4541
Epoch 2/20
30149/30149 [==============================] - 109s 4ms/step - loss: 1.3440 - accuracy:
0.5190 - val_loss: 1.3475 - val_accuracy: 0.5113
Epoch 3/20
30149/30149 [==============================] - 103s 3ms/step - loss: 1.1793 - accuracy:
0.5855 - val_loss: 1.0930 - val_accuracy: 0.6214
```

```
Epoch 4/20
30149/30149 [==============================] - 108s 4ms/step - loss: 1.0616 - accuracy:
0.6323 - val_loss: 1.0425 - val_accuracy: 0.6459
Epoch 5/20
30149/30149 [==============================] - 108s 4ms/step - loss: 0.9879 - accuracy:
0.6598 - val_loss: 0.9408 - val_accuracy: 0.6758
Epoch 6/20
30149/30149 [==============================] - 113s 4ms/step - loss: 0.9294 - accuracy:
0.6806 - val_loss: 0.8778 - val_accuracy: 0.6996
Epoch 7/20
30149/30149 [==============================] - 106s 4ms/step - loss: 0.8744 - accuracy:
0.6991 - val_loss: 0.8763 - val_accuracy: 0.6992
Epoch 8/20
30149/30149 [==============================] - 107s 4ms/step - loss: 0.8305 - accuracy:
0.7168 - val_loss: 0.8467 - val_accuracy: 0.7102
Epoch 9/20
30149/30149 [==============================] - 110s 4ms/step - loss: 0.7885 - accuracy:
0.7292 - val_loss: 0.8107 - val_accuracy: 0.7213
Epoch 10/20
30149/30149 [==============================] - 109s 4ms/step - loss: 0.7570 - accuracy:
0.7387 - val_loss: 0.8970 - val_accuracy: 0.6876
Epoch 11/20
30149/30149 [==============================] - 116s 4ms/step - loss: 0.7326 - accuracy:
0.7454 - val_loss: 0.8633 - val_accuracy: 0.7061
Epoch 12/20
30149/30149 [==============================] - 112s 4ms/step - loss: 0.6994 - accuracy:
0.7584 - val_loss: 0.7937 - val_accuracy: 0.7267
Epoch 13/20
30149/30149 [==============================] - 107s 4ms/step - loss: 0.6755 - accuracy:
0.7677 - val_loss: 0.7709 - val_accuracy: 0.7341
Epoch 14/20
30149/30149 [==============================] - 108s 4ms/step - loss: 0.6429 - accuracy:
0.7766 - val_loss: 0.8332 - val_accuracy: 0.7170
Epoch 15/20
30149/30149 [==============================] - 113s 4ms/step - loss: 0.6303 - accuracy:
0.7806 - val_loss: 0.8178 - val_accuracy: 0.7205
Epoch 16/20
30149/30149 [==============================] - 124s 4ms/step - loss: 0.6166 - accuracy:
0.7898 - val_loss: 0.8003 - val_accuracy: 0.7293
5000/5000 [==============================] - 5s 1ms/step
```

- Train result: train_loss: 0.6166 - train_accuracy: 78.9844
- Validation result: val_loss: 0.8003 - val_accuracy: 72.9311
- Test result: test_loss: 0.7737 - test_accuracy: 73.9600


model accuracy
train
validation
accuracy
0.8
0.7
0.6
0.5
0.4
0
5
10
15
epoch
model loss
train
validation
loss
1.6
1.4
1.2
1.0
0.8
0.6
0
5
10
15
epoch

Iteration	Parameters	Accuracy

1	Input layer = 1 Conv
	Input layer neurons = 32
	Hidden layers = 1 Conv
	Hidden layers neurons = 16
	Hidden layer dropout = 0.5
	Activation Function = relu
	Batch size = 32
	Epochs = 10, ***early stopped at 9***
		Train = 65.76%
		Validation = 60.72%
		Test = 62.18%

2	Input layer = 1 Conv
	Input layer neurons = 32
	Hidden layers = 2 Conv
	Hidden layers neurons = [16, 8]
	Hidden layers dropout = [0.5, 0.5]
	Activation Function = relu
	Batch size = 32
	Epochs = 10
		Train = 52.51%
		Validation = 55.91%
		Test = 56.80%

3	Input layer = 1 Conv Input layer neurons = 32 Hidden layers = 2 Conv Hidden layers neurons = [32, 8] Hidden layers dropout = [0.5, 0.5] Activation Function = relu Batch size = 50 Epochs = 20, ***interrupted at 4***	***Interrupted training in between as accuracy was coming very low in range of 25-35%***

4	Input layer = 1 Conv
	Input layer neurons = 64
	Hidden layers = 1 Conv
	Hidden layers neurons = 32
	Hidden layers dropout = 0.5
	Activation Function = relu
	Batch size = 50
	Epochs = 20, ***early stopped at 11***
		Train = 72.52%
		Validation = 65.05%
		Test = 67.72%

5	Input layer = 1 Conv
	Input layer neurons = 16
	Hidden layers = 2 Conv
	Hidden layers neurons = [32, 16]
	Hidden layers dropout = [0.5, 0.5]
	Activation Function = relu
	Batch size = 50
	Epochs = 20, ***early stopped at 14***
		Train = 73.26%
		Validation = 65.67%
		Test = 67.24%

6	Input layer = 1 Conv
	Input layer neurons = 16
	Hidden layers = 2 Conv, 1 Dense
	Hidden layers neurons = [32, 16], 100
	Hidden layers dropout = [0.5, 0.5], 0.4
	Activation Function = relu
	Batch size = 50
	Epochs = 20
		Train = 60.67%
		Validation = 67.09%
		Test = 68.84%

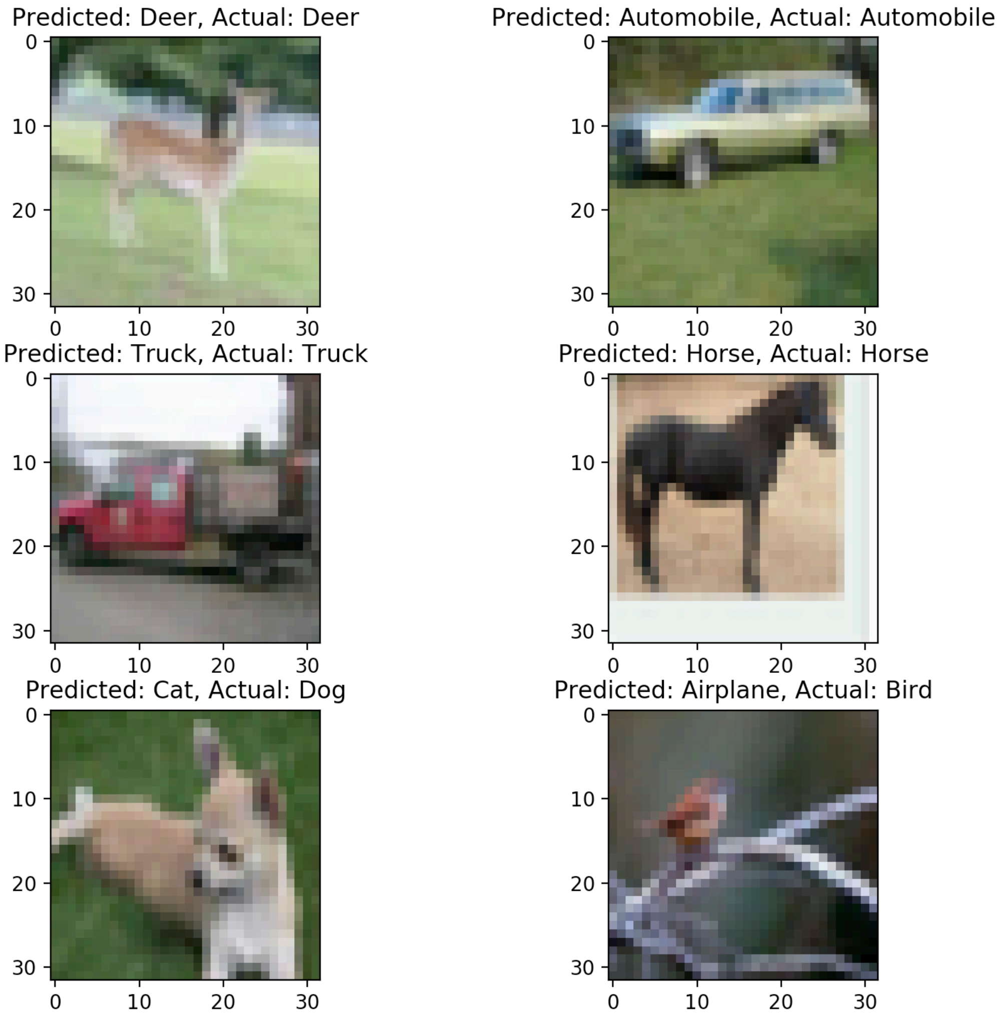
7	Input layer = 1 Conv
	Input layer neurons = 16
	Hidden layers = 2 Conv, 2 Dense
	Hidden layers neurons = [32, 64], [100, 100]
	Hidden layers dropout = [0.25, 0.25], [0.4, 0.3]
	Activation Function = relu
	Batch size = 50
	Epochs = 20, ***early stopped at 18***
		Train = 72.53%
		Validation = 71.35%
		Test = 72.68%

8	Input layer = 1 Conv Input layer neurons = 32 Hidden layers = 2 Conv, 2 Dense Hidden layers neurons = [64, 128], [200, 100] Hidden layers dropout = [0.25, 0.25], [0.4, 0.3] Activation Function = relu Batch size = 50 Epochs = 20, ***early stopped at 17***	**Train = 78.98%** **Validation = 72.93%** **Test = 73.96%**

Prediction using best model

Predictions are made using the best model. The training accuracy of the best model is ~79%.

- 4597: Predicted: Deer, Actual: Deer
- 4674: Predicted: Automobile, Actual: Automobile
- 2174: Predicted: Truck, Actual: Truck
- 4978: Predicted: Horse, Actual: Horse
- 4172: Predicted: Cat, Actual: Dog
- 3501: Predicted: Airplane, Actual: Bird


Predicted: Deer, Actual: Deer
Predicted: Automobile, Actual: Automobile
Predicted: Truck, Actual: Truck
Predicted: Horse, Actual: Horse
Predicted: Cat, Actual: Dog
Predicted: Airplane, Actual: Bird
0
10
20
30

The result is good but could be better with an improved model.
More models could be trained to obtain a better classifier, but infrastructure limitations (e.g. lack of a powerful computer) has limited the CNN model training scope.

Summary and Conclusion

CNN performed slightly better than the normal NN models created using Keras in Python. The NN model built using H2O in R underperformed relative to the one built using Keras in Python.

- Python Keras NN Model accuracy: Training ~75%, Test ~62%
- Python Keras CNN Model accuracy: Training ~79%, Test ~74%

Using H2O in R, the Model training was limited due to infrastructure limitations:

- Rectified Linear Units as the activation function
- Neuron architecture of 3072-128-64-10
- 20 epochs

Using Keras in Python, further training of more deep models could not be done due to infrastructure limitations.

A major advantage of CNN over NN is that we do not have to flatten the input images to 1D as they are capable of working with image data in 2D. This helps in retaining the "spatial" properties of images.
That is the reason, the data that was available as vectors, could be used in NN, but for CNN, a 2D image data could be used.
However, the dataset used for this project was available as 1D vectors.

The model that performed the best had the following architecture:

- Input layer Conv2D that takes in image data of size (32,32,3) i.e. 32x32 RGB image data
- Batch normalization layer following the input layer to ensure there is not much covariance shift from its output.
- 2 Conv2D hidden layers with 64 and 128 neurons and 'relu' activation function
- Max-pooling was used in the hidden Conv layers to reduce row and column dimensions.
- Dropout was used to inactivate randomly some of the nodes in Conv layers.
- Flatten layer to flatten the 2D data into 1D Vector for the following dense layers.
- 2 Dense layers with 200 and 100 neurons and 'relu' activation function.
- Dropout was used to inactivate randomly some of the nodes in Dense layers.
- Dense output layer with 10 neurons and 'softmax' activation function that is used to obtain probabilities of each of the 10 class.

The models were trained and tuned to find the best classifier

- Validation was done during training with 33% of the training data
- Batch size of 50 was used during training
- Number of epochs were set to 20
- Early stopping was done to stop training if there is no significant improvement in validation accuracy after every 3 epochs. The training stopped after 17 epochs, as a result.
- Checkpointing has been performed to save the best performing model for later use during prediction.

- More improved model could have been found but further training of more deep models could not be done due to infrastructure limitations.

CNN is found to be very useful for the image classification of its high accuracy.
The CNN follows a hierarchical (sequential) model which works on building a network, giving a fully connected layer where all the neurons are connected to each other and the output is processed.

On the input image (32×32), a filter (say 3x3) can be used to run along all the pixels (rows, columns) of the image which captures the data and then is passed on to the pooling layer where it performs a mathematical computation and gives out a specific result. Here, the filter actually is run across all the values in the pixel matrix and a dot product of the weights and the pixels is calculated.

In data in the convolution layers, is passed into the Rectified Linear Unit (ReLU) activation function to produce corresponding output, which in turn is fed into the next Convolution layer.

Before the output of the hidden convolution layer is fed to the Dense layers, it can be flattened.

Thus, this full ‘convoluted’ inter-connected network helps in training the model efficiently, making CNN as the best choice for Image Classification.

References

- Image plotting: https://matplotlib.org/api/_as_gen/matplotlib.pyplot.imshow.html
- Keras: https://www.tensorflow.org/guide/keras/sequential_model