Project Readme

Introduction

Food image recognition refers to identifying food items by seeing the food image. Different types of food can be recognized from different images. Our project has recognized around 25 types of food that include foods like ‘bruschetta’, ‘chocolate cake’, ‘French toast’, ‘chocolate mousse, etc. This has been achieved by training different models to accurately predict the distinct classes and thus come up with an app-based user interface that allows for checking recognized dish information and opt between checking the dish recipe or order the food based on the list of nearby venues and placing the order on the Zomato app linked to our app user interface.

Dataset

We have created a dataset from scratch for our project using 25 classes each for distinct food types. The unique classes identified in the dataset are as shown below:

```
array(['chocolate ', 'cup       ', 'donuts    ', 'dumplings ',
       'french    ', 'fried     ', 'garlic    ', 'gnocchi   ',
       'grilled   ', 'hamburger ', 'hot       ', 'ice       ',
       'macarons  ', 'mussels   ', 'nachos    ', 'pancakes  ',
       'pizza     ', 'red       ', 'samosa    ', 'spring    ',
       'strawberry', 'sushi     ', 'waffles   '], dtype='<U10')
```

This dataset is then stored in .mat format and then split into 80% train set and 20% test set. The count of the images for different classes in these train and test sets are shown below:

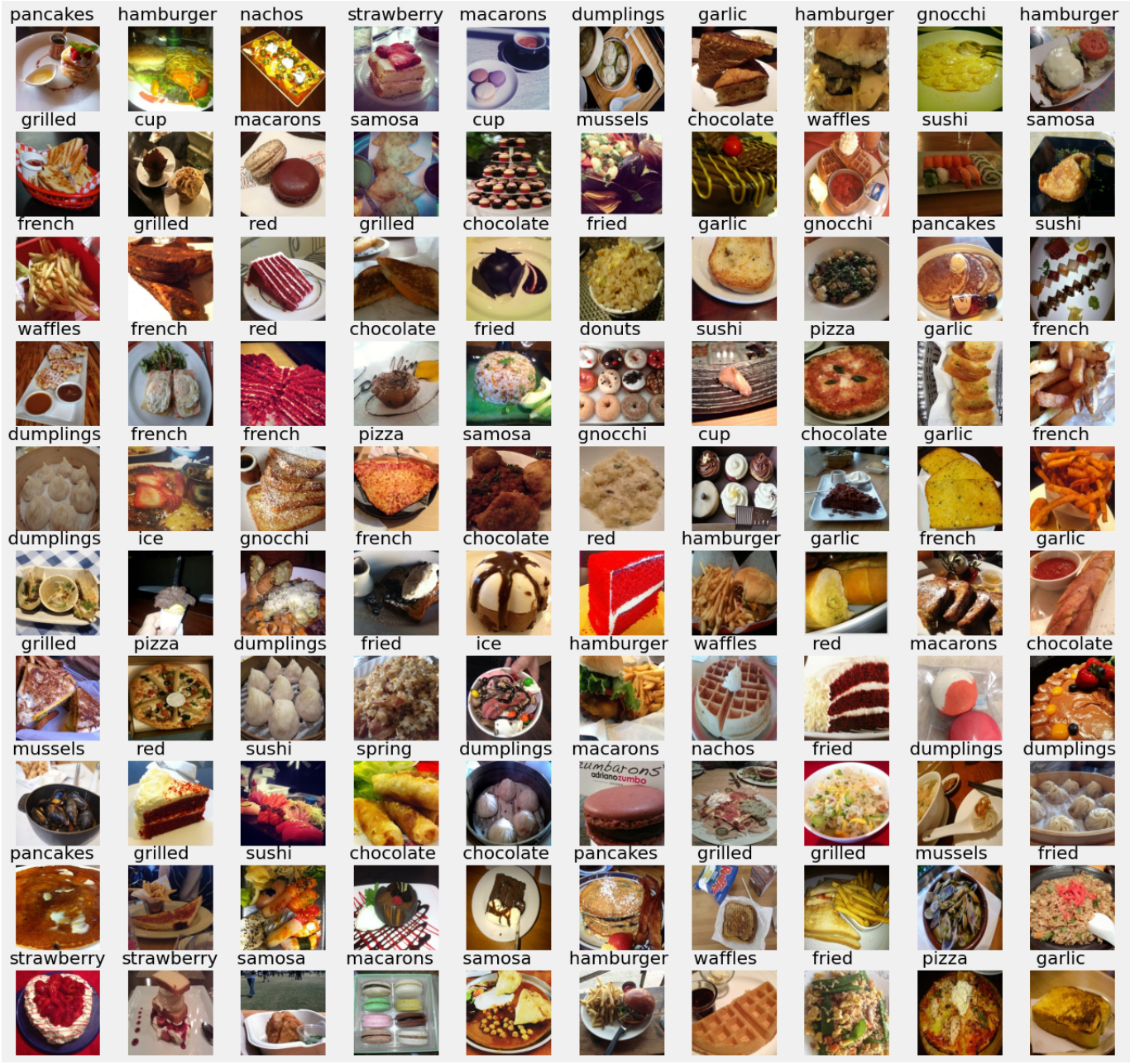
For train set:

```
{'chocolate ': 1432,
 'cup       ': 787,
 'donuts    ': 564,
 'dumplings ': 770,
 'french    ': 1412,
 'fried     ': 604,
 'garlic    ': 594,
 'gnocchi   ': 808,
 'grilled   ': 798,
 'hamburger ': 538,
 'hot       ': 709,
 'ice       ': 713,
 'macarons  ': 750,
 'mussels   ': 577,
 'nachos    ': 599,
 'pancakes  ': 653,
 'pizza     ': 562,
 'red       ': 630,
 'samosa    ': 630,
 'spring    ': 421,
 'strawberry': 587,
 'sushi     ': 538,
 'waffles   ': 635}
```

For test set:

```
{'chocolate ': 363,
 'cup       ': 173,
 'donuts    ': 143,
 'dumplings ': 210,
 'french    ': 363,
 'fried     ': 168,
 'garlic    ': 163,
 'gnocchi   ': 177,
 'grilled   ': 173,
 'hamburger ': 138,
 'hot       ': 189,
 'ice       ': 168,
 'macarons  ': 192,
 'mussels   ': 145,
 'nachos    ': 138,
 'pancakes  ': 172,
 'pizza     ': 142,
 'red       ': 146,
 'samosa    ': 162,
 'spring    ': 121,
 'strawberry': 154,
 'sushi     ': 130,
 'waffles   ': 148}
```

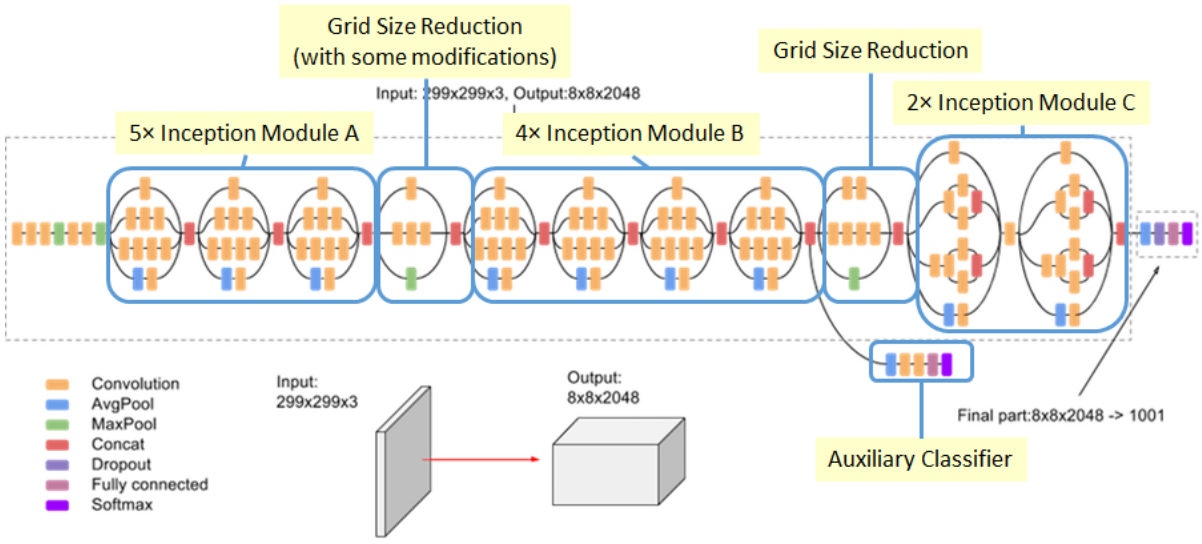
Images of different classes in the dataset are as shown in the form of grid view as below:



Models

We have implemented four models for image classification-

- Inception Model
    1. Model Architecture



    2. Model Summary

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 192, 192, 3)	0	
block1_conv1 (Conv2D)	(None, 95, 95, 32)	864	input_1[0][0]
block1_conv1_bn (BatchNormaliza	(None, 95, 95, 32)	128	block1_conv1[0][0]
block1_conv1_act (Activation)	(None, 95, 95, 32)	0	block1_conv1_bn[0][0]
block1_conv2 (Conv2D)	(None, 93, 93, 64)	18432	block1_conv1_act[0][0]
block1_conv2_bn (BatchNormaliza	(None, 93, 93, 64)	256	block1_conv2[0][0]
block1_conv2_act (Activation)	(None, 93, 93, 64)	0	block1_conv2_bn[0][0]
block2_sepconv1 (SeparableConv2	(None, 93, 93, 128)	8768	block1_conv2_act[0][0]
block2_sepconv1_bn (BatchNormal	(None, 93, 93, 128)	512	block2_sepconv1[0][0]

    3. Model Training

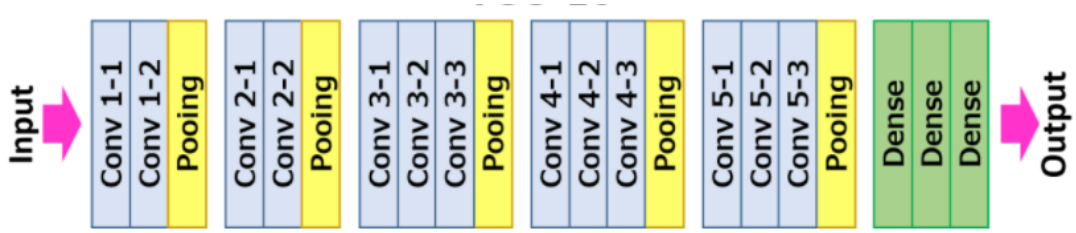
```
Epoch 1/5
164/164 [==============================] - 1237s 8s/step - loss: 2.4342 - acc: 0.2775 - val_loss: 10.5983 - val_acc: 0.2803
Epoch 2/5
164/164 [==============================] - 1235s 8s/step - loss: 1.0177 - acc: 0.7164 - val_loss: 1.4918 - val_acc: 0.6631
Epoch 3/5
164/164 [==============================] - 1211s 7s/step - loss: 0.6532 - acc: 0.8228 - val_loss: 1.1400 - val_acc: 0.7224
Epoch 4/5
164/164 [==============================] - 1222s 7s/step - loss: 0.3951 - acc: 0.8931 - val_loss: 1.2846 - val_acc: 0.6773
Epoch 5/5
164/164 [==============================] - 1219s 7s/step - loss: 0.5223 - acc: 0.8599 - val_loss: 1.0993 - val_acc: 0.7202
```

4. Model Evaluation

```
INFO:tensorflow:Assets written to: /content/drive/MyDrive/IR project-Food Segmentation/inception_model.pb/assets
16/16 [==============================] - 40s 3s/step - loss: 1.0993 - acc: 0.7202
Test loss: 1.0992960929870605
Test accuracy:  0.7202059626579285
```

- VGG19

1. Model Architecture



2. Model Summary

activation_88 (Activation)	(None, 4, 4, 384)	0	batch_normalization_88[0][0]
activation_91 (Activation)	(None, 4, 4, 384)	0	batch_normalization_91[0][0]
activation_92 (Activation)	(None, 4, 4, 384)	0	batch_normalization_92[0][0]
batch_normalization_93 (BatchNo	(None, 4, 4, 192)	576	conv2d_93[0][0]
activation_85 (Activation)	(None, 4, 4, 320)	0	batch_normalization_85[0][0]
mixed9_1 (Concatenate)	(None, 4, 4, 768)	0	activation_87[0][0] activation_88[0][0]
concatenate_1 (Concatenate)	(None, 4, 4, 768)	0	activation_91[0][0] activation_92[0][0]
activation_93 (Activation)	(None, 4, 4, 192)	0	batch_normalization_93[0][0]
mixed10 (Concatenate)	(None, 4, 4, 2048)	0	activation_85[0][0] mixed9_1[0][0] concatenate_1[0][0] activation_93[0][0]
flatten (Flatten)	(None, 32768)	0	mixed10[0][0]
dense (Dense)	(None, 23)	753687	flatten[0][0]

```
Total params: 22,556,471
Trainable params: 16,969,623
Non-trainable params: 5,586,848
```

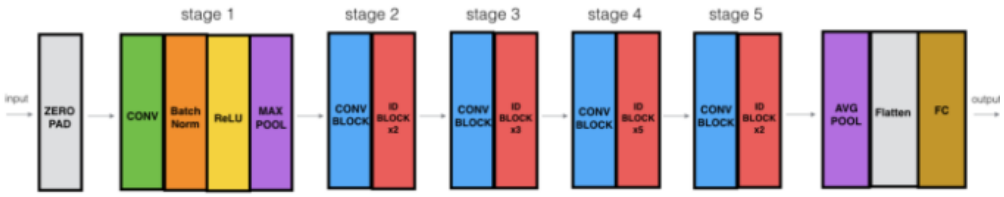
3. Model Training

```
Epoch 1/10
510/510 [==============================] - 100s 119ms/step - loss: 0.2103 - acc: 0.1399 - val_loss: 0.1989 - val_acc: 0.1523
Epoch 2/10
510/510 [==============================] - 55s 108ms/step - loss: 0.1624 - acc: 0.2322 - val_loss: 0.2144 - val_acc: 0.1604
Epoch 3/10
510/510 [==============================] - 55s 109ms/step - loss: 0.1475 - acc: 0.3061 - val_loss: 0.2062 - val_acc: 0.1844
Epoch 4/10
510/510 [==============================] - 55s 108ms/step - loss: 0.1334 - acc: 0.3727 - val_loss: 0.1965 - val_acc: 0.1851
Epoch 5/10
510/510 [==============================] - 55s 108ms/step - loss: 0.1140 - acc: 0.4737 - val_loss: 0.1786 - val_acc: 0.2401
Epoch 6/10
510/510 [==============================] - 55s 108ms/step - loss: 0.0964 - acc: 0.5797 - val_loss: 0.1705 - val_acc: 0.2602
Epoch 7/10
510/510 [==============================] - 55s 109ms/step - loss: 0.0752 - acc: 0.7151 - val_loss: 0.1802 - val_acc: 0.2374
Epoch 8/10
510/510 [==============================] - 55s 108ms/step - loss: 0.0564 - acc: 0.8297 - val_loss: 0.1875 - val_acc: 0.2496
Epoch 9/10
510/510 [==============================] - 55s 108ms/step - loss: 0.0387 - acc: 0.9188 - val_loss: 0.1974 - val_acc: 0.2773
Epoch 10/10
510/510 [==============================] - 55s 108ms/step - loss: 0.0230 - acc: 0.9768 - val_loss: 0.1877 - val_acc: 0.2916
```

4. Model Evaluation

```
16/16 [==============================] - 16s 704ms/step - loss: 0.1877 - acc: 0.2916
Test loss: 0.1876789629459381
Test accuracy:  0.2915644943714142
```

- ResNet-50
  1. Model Architecture



ResNet-50 Model

2. Model Summary

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_2 (InputLayer)            [(None, 192, 192, 3) 0
__________________________________________________________________________________________________
conv1_pad (ZeroPadding2D)       (None, 198, 198, 3)  0           input_2[0][0]
__________________________________________________________________________________________________
conv1_conv (Conv2D)             (None, 96, 96, 64)   9472        conv1_pad[0][0]
__________________________________________________________________________________________________
conv1_bn (BatchNormalization)   (None, 96, 96, 64)   256         conv1_conv[0][0]
__________________________________________________________________________________________________
conv1_relu (Activation)         (None, 96, 96, 64)   0           conv1_bn[0][0]
__________________________________________________________________________________________________
pool1_pad (ZeroPadding2D)       (None, 98, 98, 64)   0           conv1_relu[0][0]
__________________________________________________________________________________________________
pool1_pool (MaxPooling2D)       (None, 48, 48, 64)   0           pool1_pad[0][0]
__________________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 48, 48, 64)   4160        pool1_pool[0][0]
__________________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 48, 48, 64)   256         conv2_block1_1_conv[0][0]
```

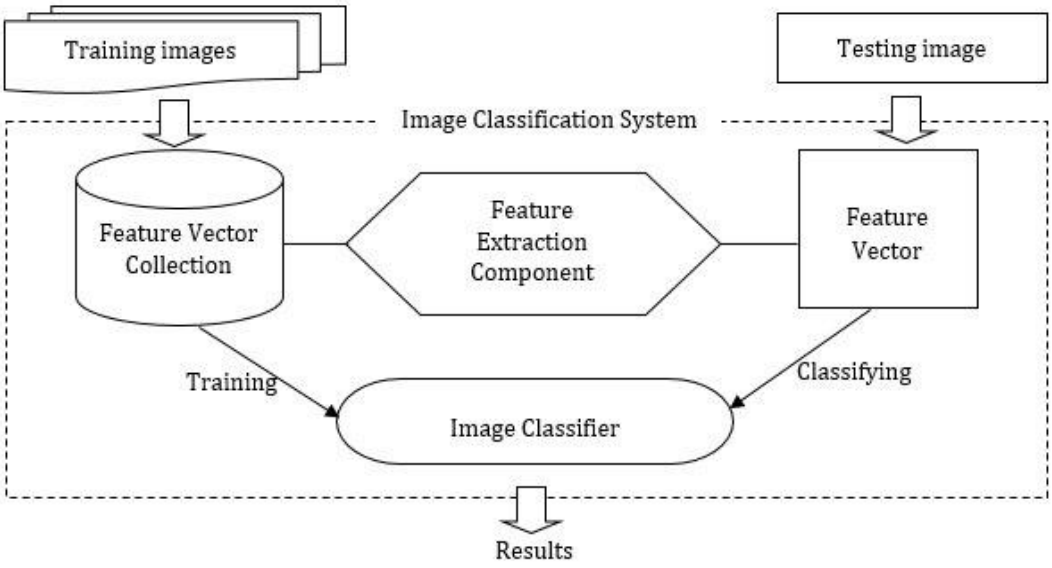
3. Model Training

```
Epoch 1/5
164/164 [==============================] - 1055s 6s/step - loss: 2.8471 - acc: 0.1692 - val_loss: 25.3989 - val_acc: 0.0508
Epoch 2/5
164/164 [==============================] - 1056s 6s/step - loss: 2.1741 - acc: 0.3505 - val_loss: 8.7925 - val_acc: 0.0758
Epoch 3/5
164/164 [==============================] - 1036s 6s/step - loss: 1.7886 - acc: 0.4592 - val_loss: 4.7902 - val_acc: 0.1525
Epoch 4/5
164/164 [==============================] - 1032s 6s/step - loss: 1.5432 - acc: 0.5262 - val_loss: 3.0078 - val_acc: 0.3477
Epoch 5/5
164/164 [==============================] - 1017s 6s/step - loss: 1.2421 - acc: 0.6212 - val_loss: 4.6531 - val_acc: 0.2805
```

4. Model Evaluation

```
INFO:tensorflow:Assets written to: /content/drive/MyDrive/IR project-Food Segmentation/inception_model.pb/assets
16/16 [==============================] - 42s 3s/step - loss: 4.6531 - acc: 0.2805
Test loss: 4.6531243324279785
Test accuracy:  0.2805296778678894
```

- Model Maker
  1. Model Architecture



  2. Model Summary

```
INFO:tensorflow:Retraining the models...
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
hub_keras_layer_v1v2 (HubKer (None, 1280)              3413024
_________________________________________________________________
dropout (Dropout)            (None, 1280)              0
_________________________________________________________________
dense (Dense)                (None, 25)                32025
=================================================================
Total params: 3,445,049
Trainable params: 32,025
Non-trainable params: 3,413,024
_________________________________________________________________
```

3. Model Training

```
Epoch 1/5
573/573 [==============================] - 195s 285ms/step - loss: 1.9632 - accuracy: 0.5555
Epoch 2/5
573/573 [==============================] - 163s 284ms/step - loss: 1.2822 - accuracy: 0.7874
Epoch 3/5
573/573 [==============================] - 168s 294ms/step - loss: 1.2187 - accuracy: 0.8151
Epoch 4/5
573/573 [==============================] - 170s 297ms/step - loss: 1.1908 - accuracy: 0.8285
Epoch 5/5
573/573 [==============================] - 169s 295ms/step - loss: 1.1732 - accuracy: 0.8323
```

4. Model Evaluation

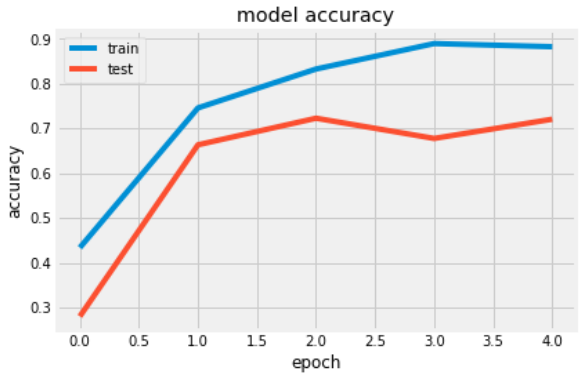
```
loss,accuracy=model.evaluate(test_data)
```

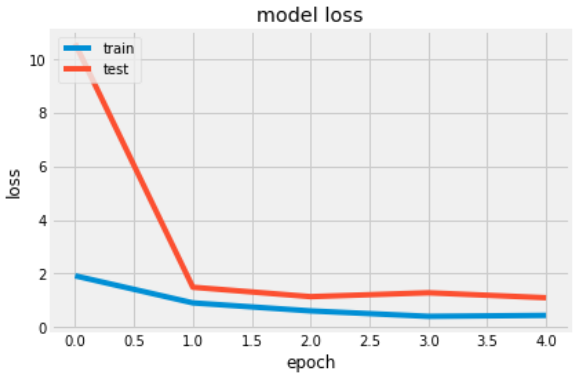
```
64/64 [==============================] - 41s 297ms/step - loss: 1.1634 - accuracy: 0.8288
```

Comparison between models

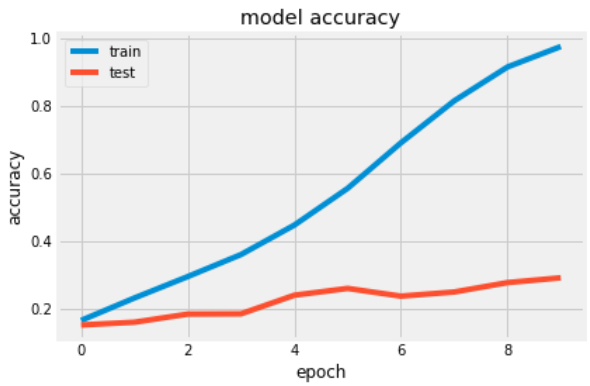
Here we are showing the accuracy and loss graph of each model.

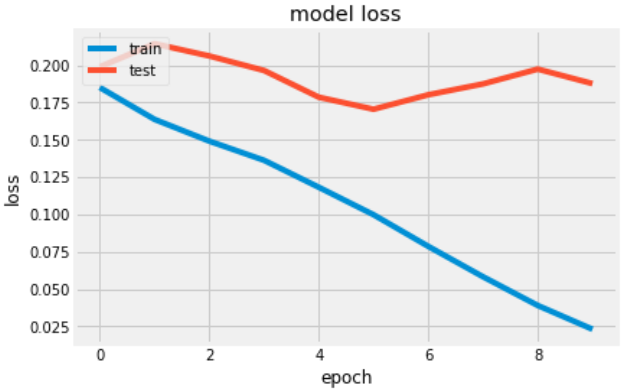
- Inception Model



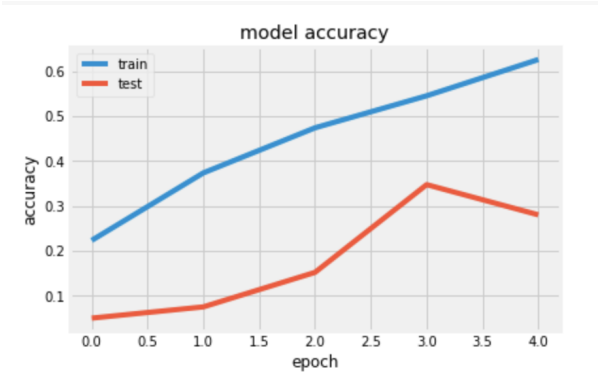


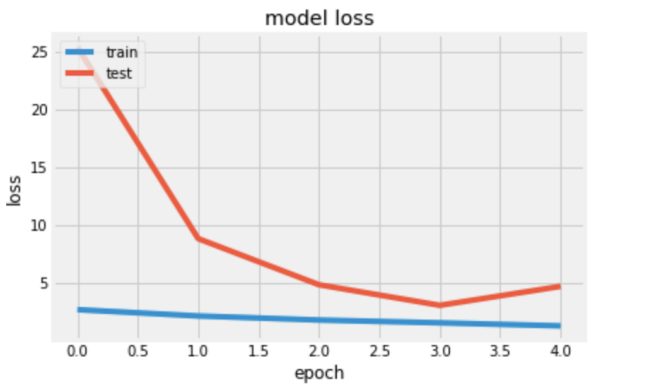
- VGG19 Model



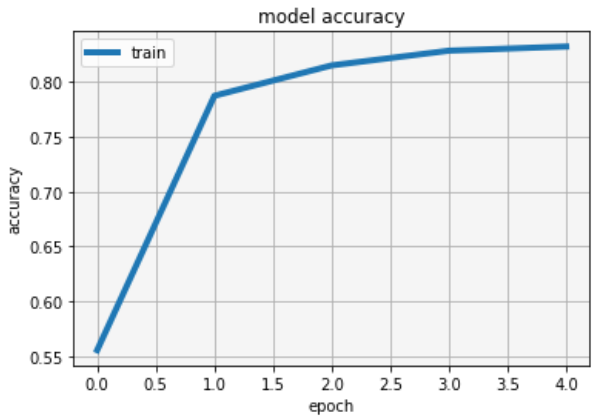


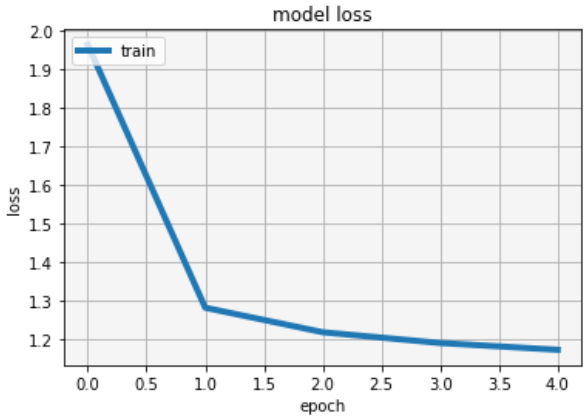
- ResNet50 Model





- Model Maker





As compared to all the models above, we can clearly observe that the model maker is performing better than other models. This model has the best accuracy of 83% and the least loss value of 1.1634.

Challenges

- For implementing the large dataset and different neural networking models, we faced a computational challenge where we sampled our dataset and implemented models with fewer epochs.
- As we have to maintain dataset transparency we had to build the dataset of 25 classes from our own which was challenging. We also removed noise from the dataset manually.
- One of the main challenges was retrieving nearby restaurants by using Foursquare API.

Future Work

- Utilization of a larger dataset with more classes to recognize a better broad spectrum of food types.
- Training the data on a larger number of epochs on a platform that provides better computational features.
- Modification of the app user interfaces by providing in-app ordering features instead of providing options to switch to other food-ordering apps.