NGEE ANN POLYTECHNIC

School of InfoComm Technology

Deep Learning Assignment

Diploma in CSF / FI / IT

April 2020 Semester

ASSIGNMENT 1

(30% of DL Module)

16th Nov 2020 – 23rd Dec 2020

Submission Deadline:

Presentation: 16th Dec 2020 (Wednesday), 11:59PM

Report: 23rd Dec 2020 (Wednesday), 11:59PM

Tutorial Group	**:**	**P01**
Student Name	**:**	Victor Jong Soon Peng
Student Number	**:**	S10186045H

Penalty for late submission:

10% of the marks will be deducted every calendar day after the deadline.

NO submission will be accepted after 30th Dec 2021 (Wednesday), 11:59PM.

Table of Contents

Overview

1.1 Problem

The well-being of humans is becoming a hot topic in recent years and an important factor related to the improvements in the quality of human life. With the modern technology, there are wearable devices such as smart watches, health bands, smart glasses, and smart shoes to gather relevant information from human's activities such as steps walked, heart rate, temperature, respiration etc. and analyze this information in terms of the amount of calories consumed, level of stress, exercise intensity, duration and quality of sleep etc. The devices will then recommend a series of actions you can take to improve your quality of life. For example, the Apple watch will have a recommendation such as "You have not exercised for the past 7 days, start exercising now to meet your target" if it detects changes in your exercise habit.

People are becoming more and more health conscious nowadays and they care very much about their dietary intake since it is the leading factor to numerous health problems, such as obesity and diabetes. Accurately labelling food items becomes extremely important to help us keep fit and live a healthy life. For example, a food understanding engine installed in wearable cameras can create food-logs of the daily intake of a patient; these information can help experts to have a better understanding in terms of the dietary habits, behavior and/or eating disorders of a patient.

An automatic image-based dietary assessment system follows the basic steps of food image detection, quantity or weight estimation, and finally caloric and nutritional value assessment (Giovanni Maria Farinella, 2016). However, it is certainly not easy to access the nutritional value of food and beverage consumed by humans accurately in an automatic way. It is extremely difficult to accurately identify every food item because many of the food items consist of different colors, shapes and some are not even distinguishable to human eyes. A plate which consists of a mixture of multiple foods is even harder to recognize by the machine. Today, with the exponential explosion of available data due to advancement of the Internet, and the advancement in computing hardware and software, it is now even more possible to accurately detect and label food with the use of deep learning.

1.2 Objective

This assignment explores food image classification with convolutional neural networks (CNNs) for better image labelling and classification by dish, which is the foundation to determine the nutritional value of the dish. The objective of this assignment is to, given an image of a dish as the input to the model, output the correct categorization of the food image. The 10 dishes that the model will recognize are beet salad, ceviche, chicken quesadilla, creme brulee, garlic bread, macaroni and cheese, miso soup, pad Thai, shrimp and grits, and sushi.

1.3 Approach

There will be three final models in this assignment, one built from scratch, two built from pre-trained weights learned on a larger image dataset (transfer learning). For the build from scratch model, I will start with 1 convolutional and 1 fully-connected layer. This model will be optimized through the use of network topology, data augmentation, dropout, learning rate, batches and epochs, optimization and loss, and regularization. For the pre-trained model, I will be using Vgg16 and InceptionV3. This pre-trained model will be optimized through data augmentation, unfreezing one layer or more for fine-tuning. These three final models will be evaluated using test images. A detailed analysis of the model performance will be done to determine the best model. The best model will be used to further proof the accuracy of image recognition by predicting images that was never seen by the model from google images.

2 Data Pre-processing and Data Loading

The Food-101 dataset contains 101 classes of food, with 1000 images for each class (DanB, 2017). In this assignment, there are only 10 classes of food (beet salad, ceviche, chicken quesadilla, creme brulee, garlic bread, macaroni and cheese, miso soup, pad Thai, shrimp and grits, and sushi) to be trained for the model. Therefore, there is a need to extract these 10 classes of food from the 101 classes of food into training, validation, and testing images. The training images are the sample of data used to fit the model. The model will “see” and “learns” (weights and biases are adjusted in the neural network) from this training data. The validation images are used to provide an unbiased evaluation of the trained model so that we can fine-tune the model hyperparameters to further improve the accuracy of the model. The testing images are used to provide unbiased evaluation of the final model trained by the training dataset. This evaluation of the testing images will be used to identify the best model.

A total of 10,000 images from 10 classes of food were used from the Food-101 dataset, with 1000 images for each class. Of the 1000 images from each class, 750 images were separated into training images, 200 images were used for validation images and 50 images were used for testing images. Manual inspection of the 10 classes of food is done to ensure there are no mislabeled images. During the manual inspection, images from the dataset have different lighting, framing, coloring, and sizes. This could potentially lead to lower accuracy.

To load the images into Jupyter notebook, there is a need to “import os” so that Jupyter notebook can access the images via the base, training, and testing directory. All the images were resized to 150x150 before passing into the model for the training phase. The reason for choosing an image size of 150x150 as compared to 50x50 is because a bigger image size will result in a better accuracy in most cases. I did not go beyond 150x150 image size because the filter will take longer and the memory requirement will rise, resulting in a longer training time.

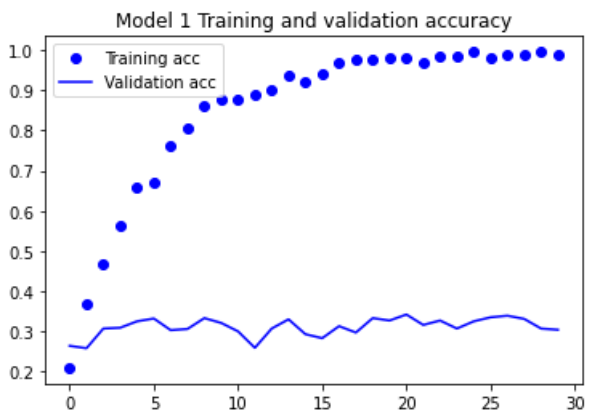
During the training phase, the images were augmented through rotation, shifting, horizontal flipping, and enlarging to avoid overfitting. In order to make use of data augmentation, it is required to import ImageDataGenerator from Keras.

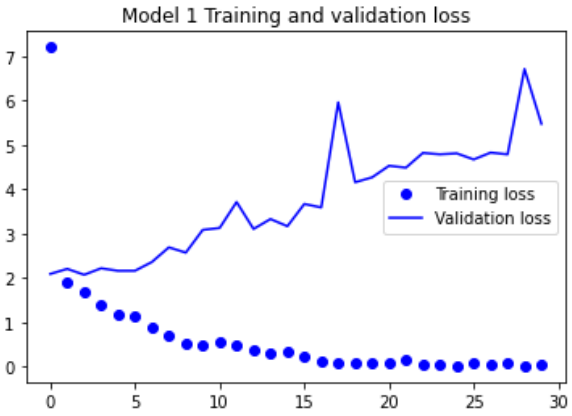
3 Develop the Image Classification Model

3.1 Build from scratch

3.1.1 Model 1

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-

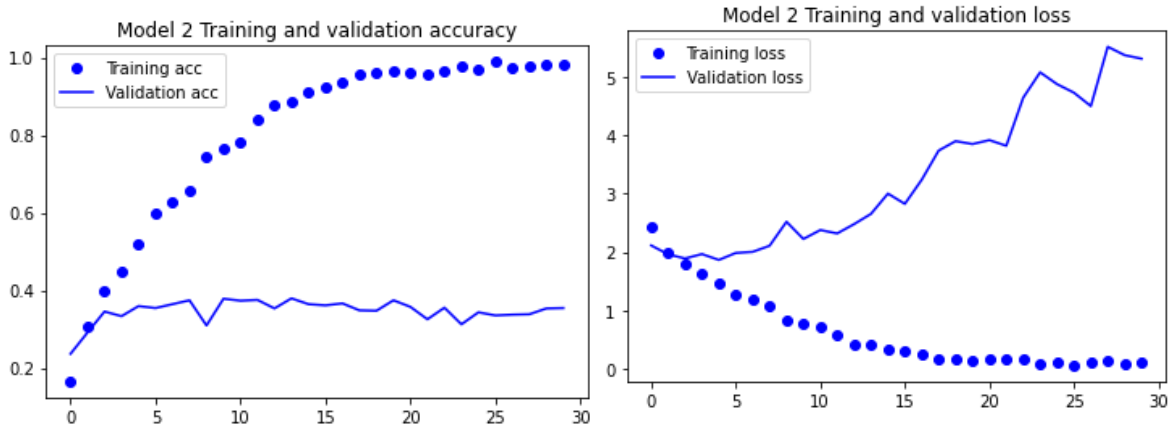




Model 1 is trained with **1 convolutional** and **1 fully-connected layer**. The validation accuracy achieved is about 30% and the model begins to overfit within 5 epochs. This low validation accuracy is because the model is too simple.

3.1.2 Model 2

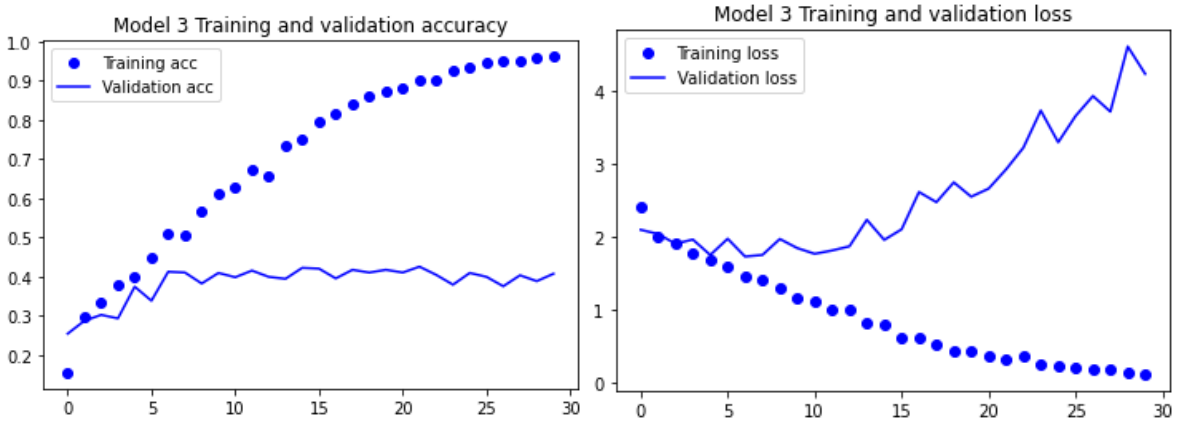
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	**32**	**Relu**	**3x3**	**-**	**-**
Max Pooling	**-**	**-**	**2x2**	**-**	**-**
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 2 is trained with **2 convolutional** and **1 fully-connected layer**. The validation accuracy achieved is about 35% and the model begins to overfit within 5 epochs. This is about a 5% increase in validation accuracy as compared to Model 1. By increasing one more convolutional layer, the model becomes more complex and yields higher validation accuracy. However, there must be a balance between a too complex and too simple model. Therefore, we will try different layers in the next few models to determine what is the best number of layers.

3.1.3 Model 3

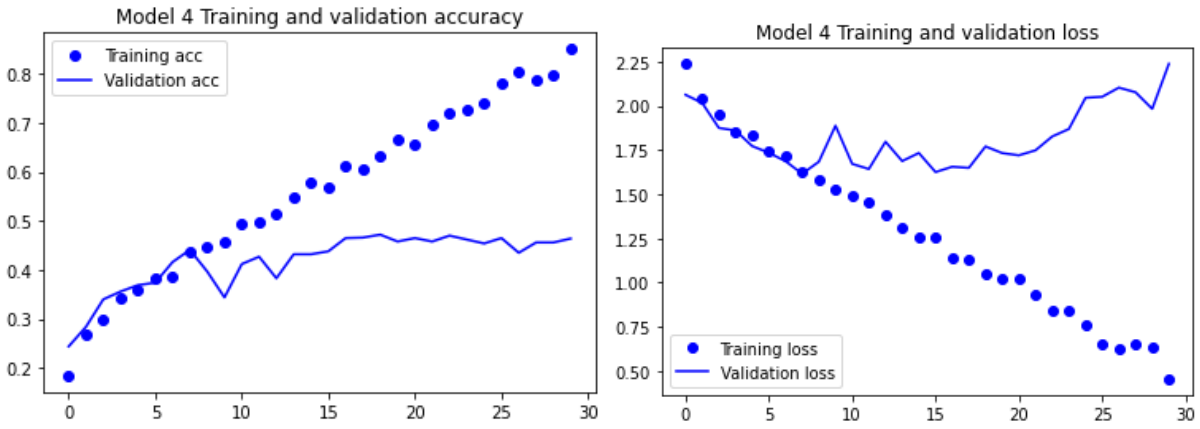
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	**32**	**Relu**	**3x3**	**-**	**-**
Max Pooling	**-**	**-**	**2x2**	**-**	**-**
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 3 is trained with **3 convolutional** and **1 fully-connected layer**. The validation accuracy achieved is about 40% and the model begins to overfit within 10 epochs. This is about a 5% increase in validation accuracy as compared to Model 2. We will continue to add more convolutional layers until there is no or minimum increase in validation accuracy.

3.1.4 Model 4

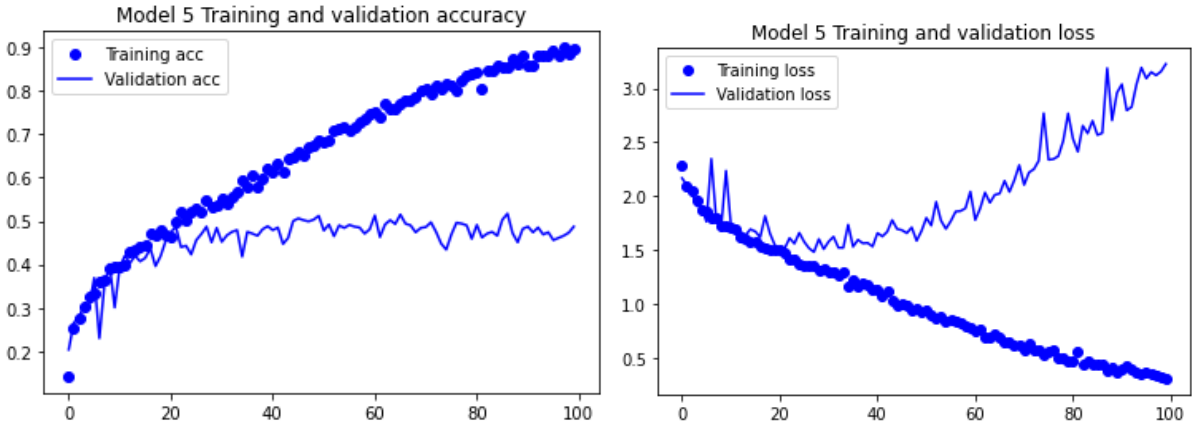
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	**32**	**Relu**	**3x3**	**-**	**-**
Max Pooling	**-**	**-**	**2x2**	**-**	**-**
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 4 is trained with **4 convolutional** and **1 fully-connected layer**. The validation accuracy achieved is about 45% and the model begins to overfit within 15 epochs. This is about a 5% increase in validation accuracy as compared to Model 3. We will continue to add more convolutional layers until there is no or minimum increase in validation accuracy.

3.1.5 Model 5

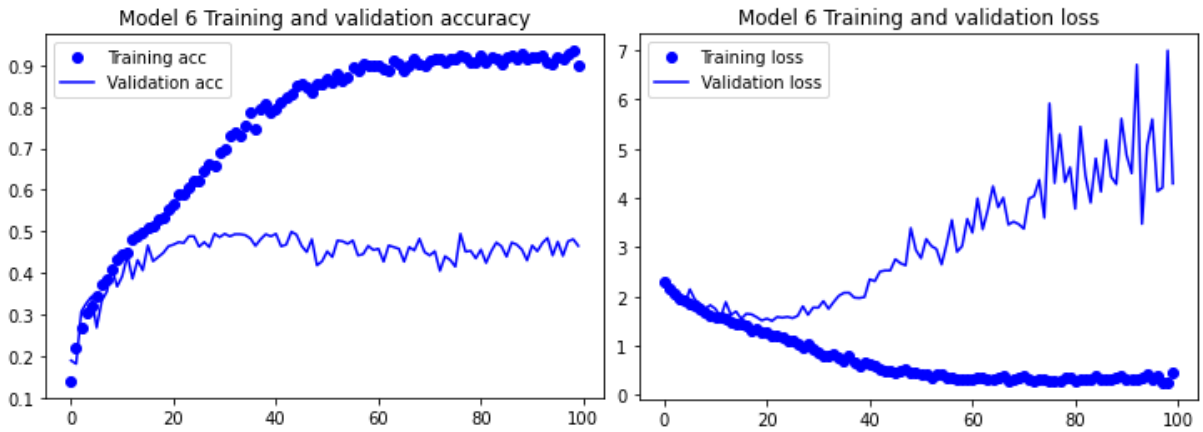
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	**32**	**Relu**	**3x3**	**-**	**-**
Max Pooling	**-**	**-**	**2x2**	**-**	**-**
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 4 is trained with **5 convolutional** and **1 fully-connected layer**. The validation accuracy achieved is about 49% and the model begins to overfit within 25 epochs. This is about a 4% increase in validation accuracy as compared to Model 4. We will use this model for further improvement.

3.1.6 Model 6

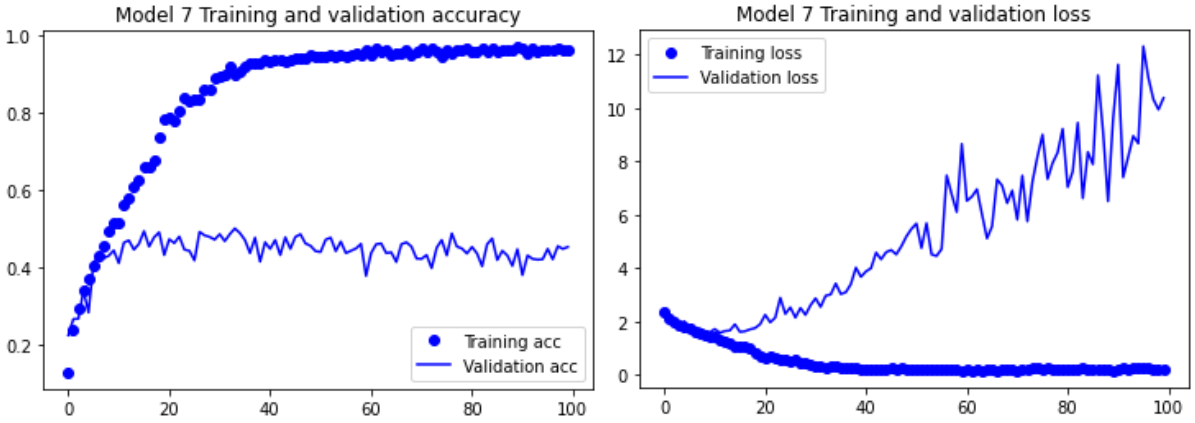
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	**32**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**64**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**128**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**128**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**128**	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 6 is trained with **5 convolutional** and **1 fully-connected layer**. I have **increased the number of neurons** for every convolutional network. The validation accuracy achieved is about 49% and the model begins to overfit within 25 epochs. By increasing the number of neurons, the model is more complex. However, we can no longer see improvements in validation accuracy.

3.1.7 Model 7

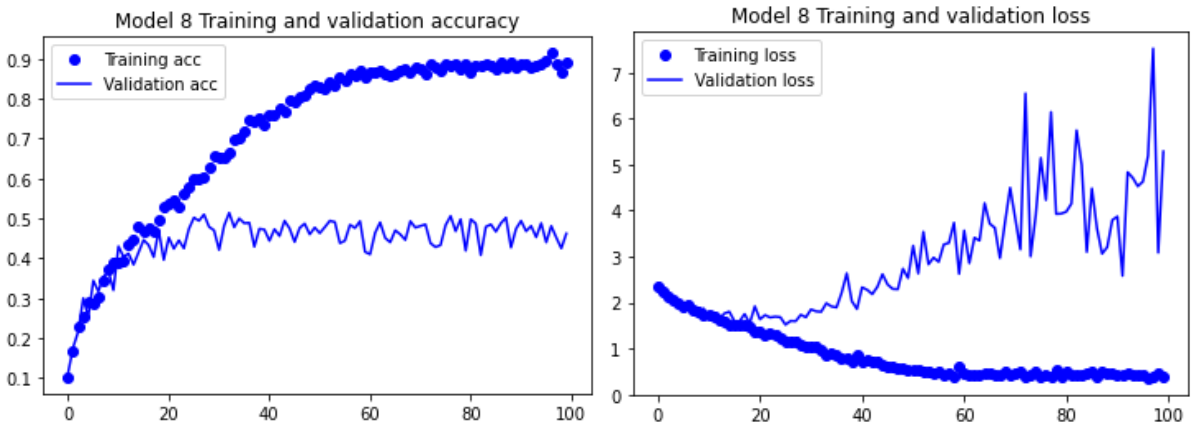
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	**32**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**64**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**128**	Relu	3x3	-	-
Max Pooling	**-**	-	2x2	-	-
Convolutional	**128**	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Fully Connected	512	Relu	-	-	-
Output	10	softmax	-	-	-



Model 7 is trained with **4 convolutional** and **1 fully-connected layer**. I have **increased the number of neurons for every convolutional network**. The validation accuracy achieved is about 45% and the model begins to overfit within 25 epochs. The validation is 4% lesser than model 6. Therefore, I will continue to improve Model 6.

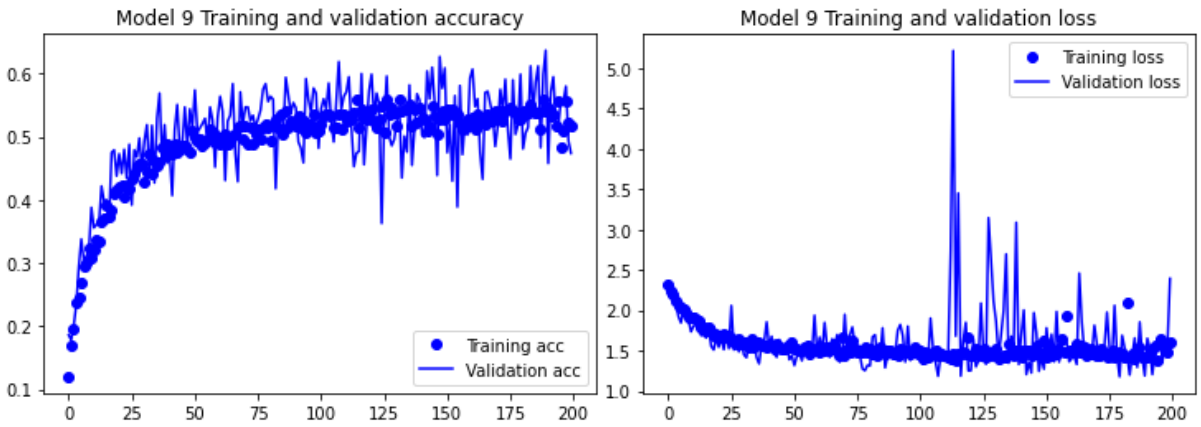
3.1.8 Model 8

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	64	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Fully Connected	**256**	**Relu**	**-**	**-**	**-**
Fully Connected	**256**	**Relu**	**-**	**-**	**-**
Output	10	softmax	-	-	-



Model 8 is trained with **5 convolutional** and **2 fully-connected layers** with a **smaller number of neurons**. The validation accuracy achieved is about 50% and the model begins to overfit within 25 epochs. This is about 1% more in terms of validation accuracy as compared to model 6. Therefore, model 8 will be used for further improvement (reduce overfitting).

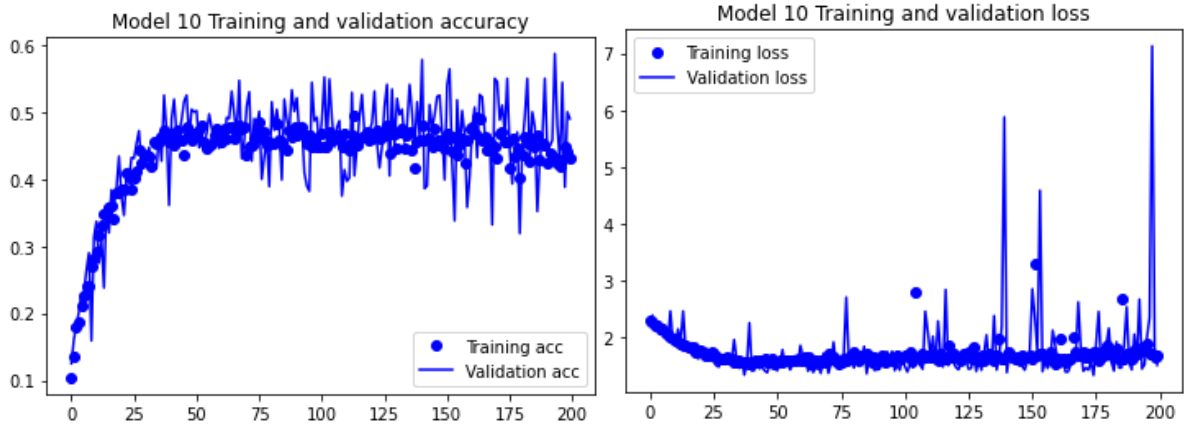
3.1.9 Model 9



Model 9 is the continuous of Model 8. **Data augmentation** is used to reduce overfitting. The validation accuracy achieved is about 50%. Most importantly, the validation accuracy is closer to the training model. However, the validation loss is fluctuating. To reduce this fluctuation, I can try adding dropout and regularizers. **In the future build from scratch models, data augmentation will be used.**

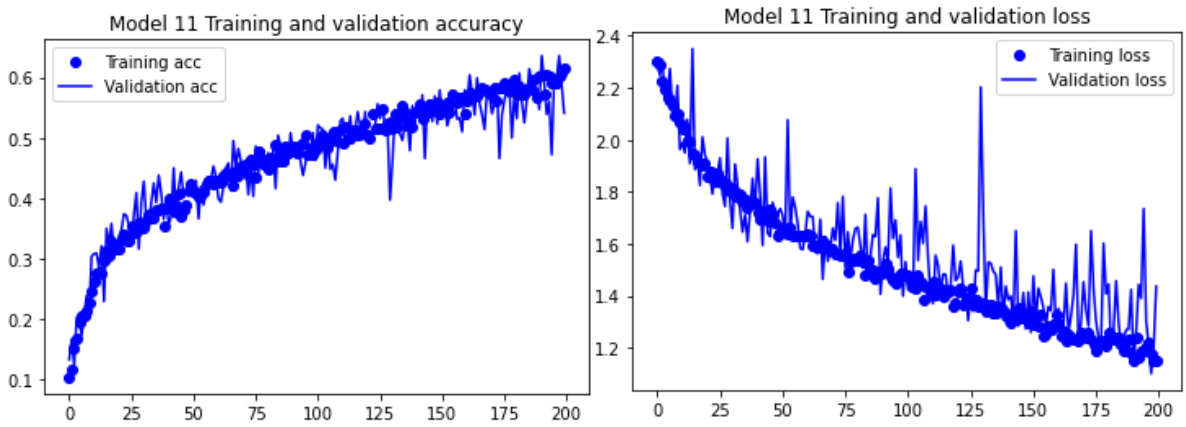
3.1.10 Model 10

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	64	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Dropout	-	-	-	-	-
Fully Connected	256	Relu	**-**	**-**	**-**
Fully Connected	256	Relu	**-**	**-**	**-**
Output	10	softmax	-	-	-



Model 10 is trained with **dropout** to prevent overfitting. The validation accuracy remains at 48%. This is a 2% decrease of validation accuracy from model 9. The training accuracy starts decreasing after epoch 50, indicating that the learning rate was too high. This model will be used to further improve accuracy.

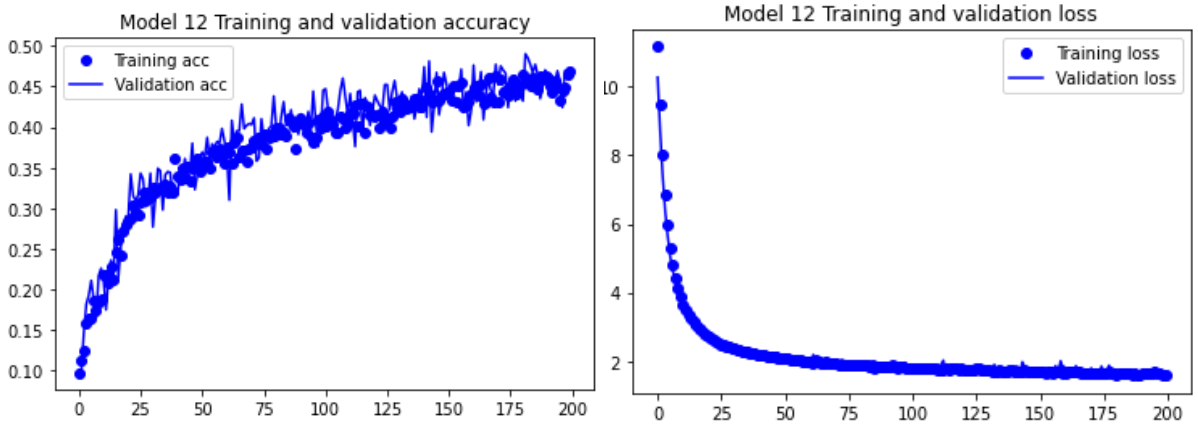
3.1.11 Model 11



Model 11 is the continuation from model 10, I have changed the **learning rate from 0.001 to 0.0001** in Model 11. The validation accuracy increased significantly to 60%. This is a 13% increase of validation accuracy from model 10. This model will be used to further improve accuracy. More epochs should be added to evaluate the model. Its validation loss is very noisy.

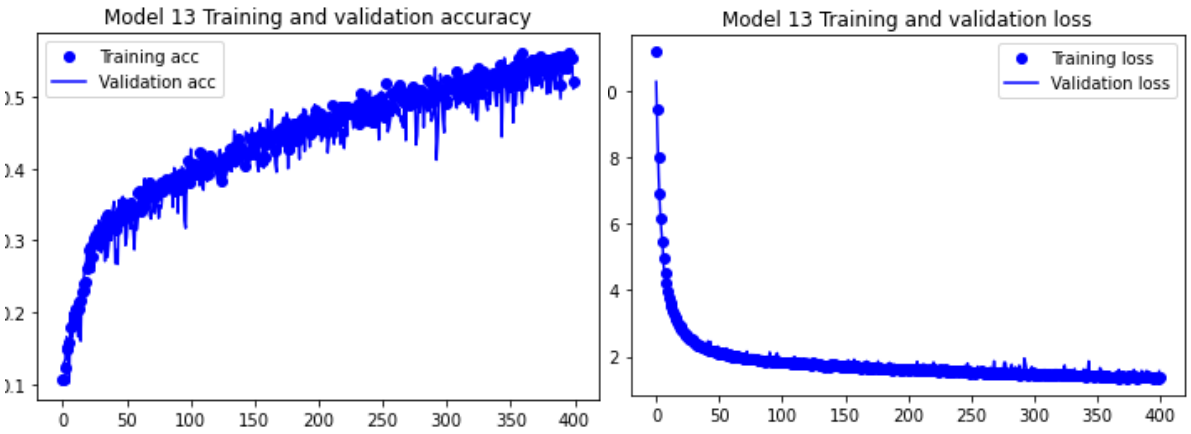
3.1.12 Model 12

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Convolutional	32	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	64	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Convolutional	128	Relu	3x3	-	-
Max Pooling	-	-	2x2	-	-
Fully Connected	256	Relu	**-**	**0.001**	**0.001**
Fully Connected	256	Relu	**-**	**0.001**	**0.001**
Output	10	softmax	-	-	-



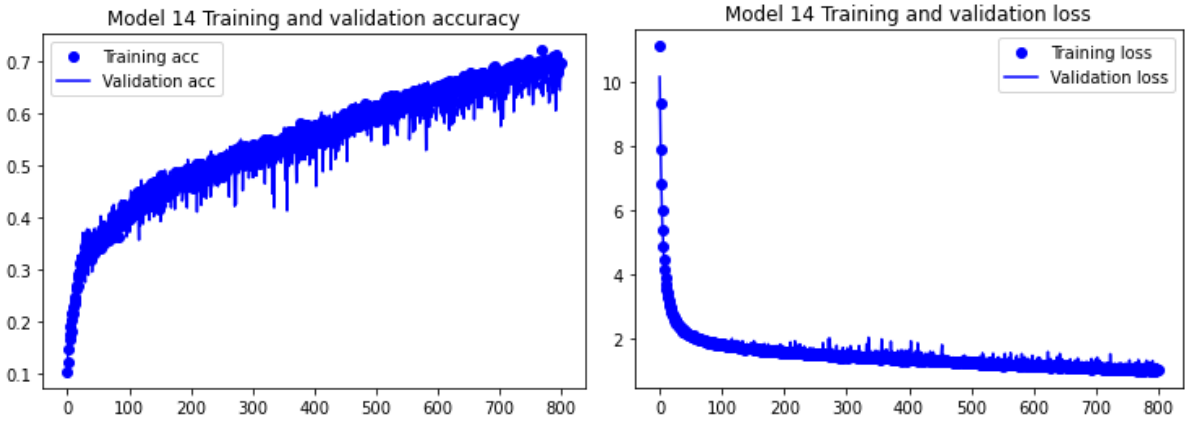
I have added **regularizers (l1=0.001, l2=0.001) to the 2 fully-connected layers** to prevent overfitting in Model 12. There is lesser noise in the validation loss. The validation accuracy is increasing, therefore more epochs should be added to evaluate the model.

3.1.13 Model 13



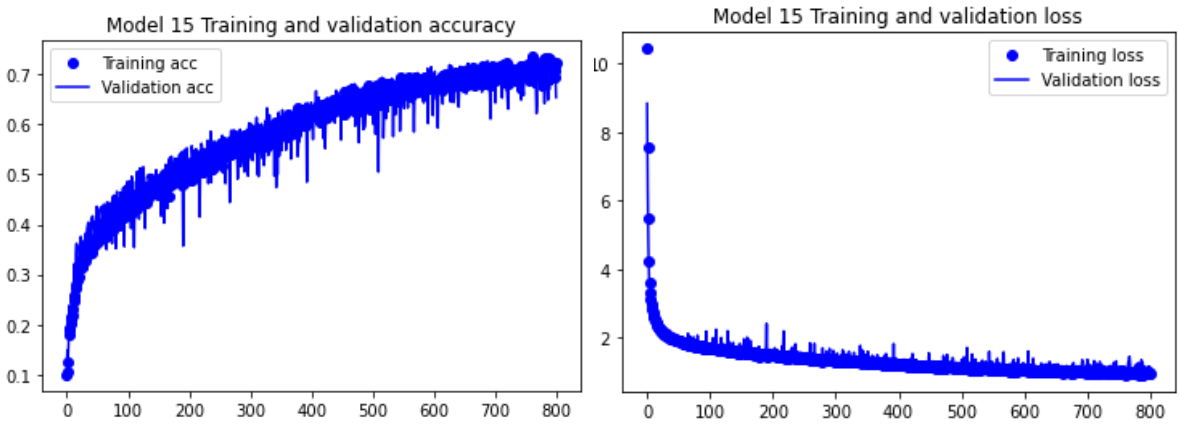
Model 13 is the continuation from model 12. I have **added more epoch (from 200 to 400)** in model 13 to better evaluate the model. The highest validation accuracy is at 53.40%. However, it seems like the validation accuracy is still increasing and overfitting is not occurring yet, therefore more epochs will be needed to evaluate the model.

3.1.14 Model 14



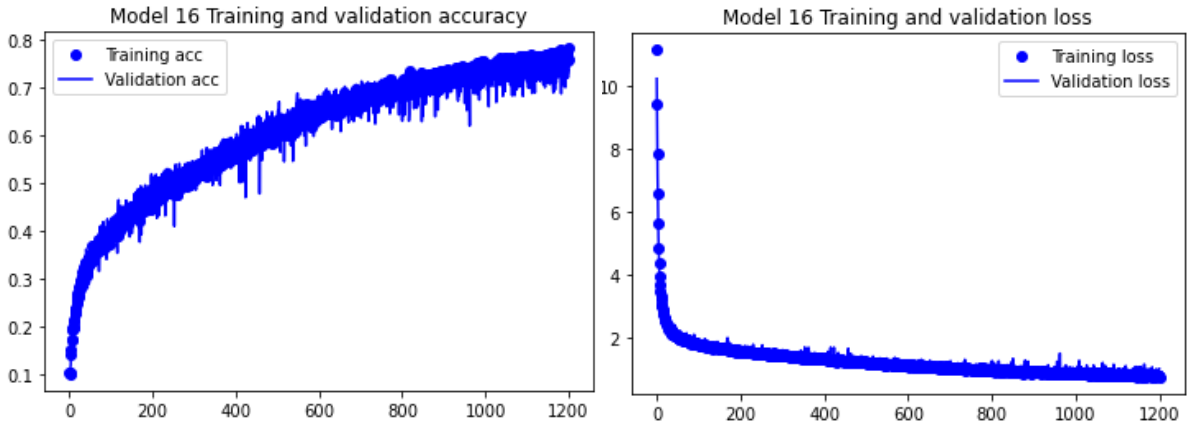
Model 13 is the continuation from model 13. I have **added more epoch (from 400 to 800)** in model 14 to better evaluate the model. The highest validation accuracy is at 67.80%. However, it seems like the validation accuracy is still increasing and overfitting is not occurring yet, therefore more epochs will be needed to evaluate the model.

3.1.15 Model 15



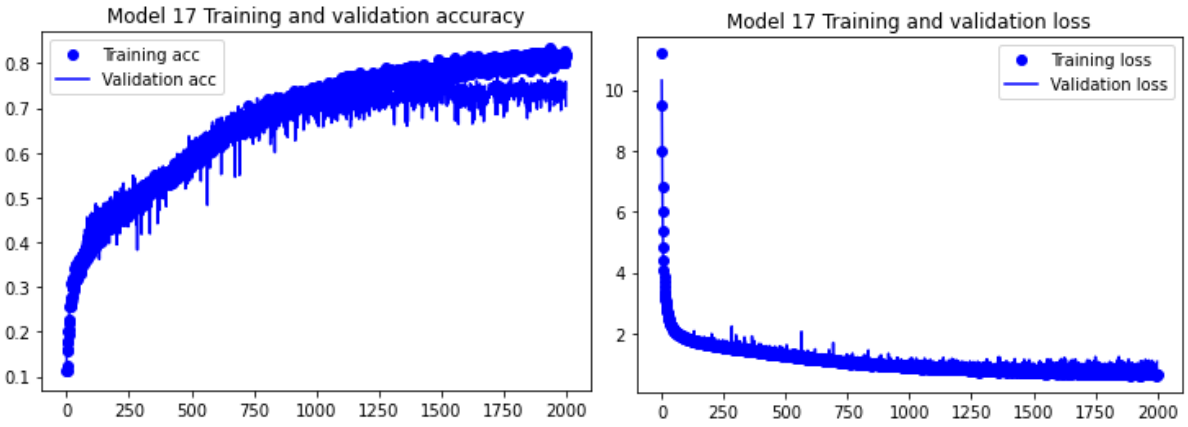
In model 15, I have **reduced the training batch size and increased step per epoch**. The highest validation accuracy is at 72.10%. This is an increase of 4.3% from Model 14. Although the learning is faster as compared to model 14, the validation loss becomes noisier. Therefore, I prefer to continue to improve model 15.

3.1.16 Model 16



Model 16 is the continuation from model 14. I have **added more epoch (from 800 to 1200)** in model 16 to better evaluate the model. The highest validation accuracy is at 72.90%. However, it seems like the validation accuracy is still increasing and overfitting is not occurring yet, therefore more epochs will be needed to evaluate the model.

3.1.17 Model 17

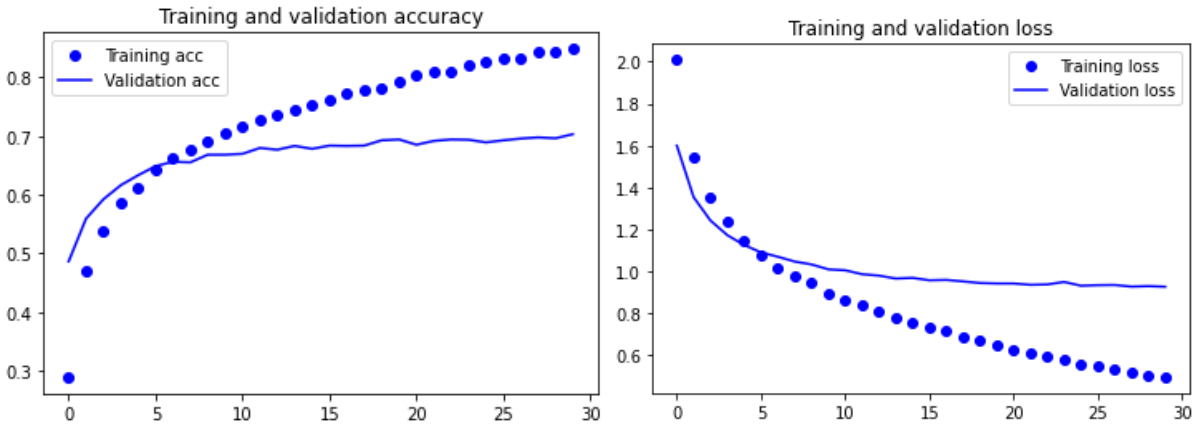


Model 17 is the continuation from model 16. I have **added more epoch (from 1200 to 800)** in model 17 to better evaluate the model. The highest validation accuracy is at 74.10%. Overfitting occurs at about 1500 epochs. This will be my final model in build from scratch model.

3.2 VGG 16

3.2.1 Model 20

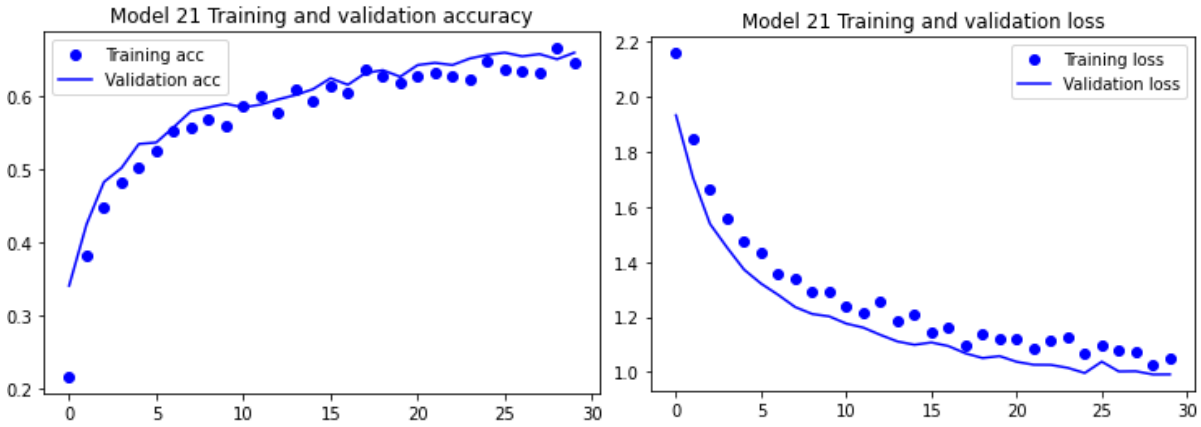
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
vgg16	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 20 is trained with **vgg16 (pre-trained model)** and **1 fully-connected layer**. The validation accuracy achieved is about 70% and the model begins to overfit within 15 epochs. The validation started out high because the pre-trained model is trained with many images.

3.2.2 Model 21

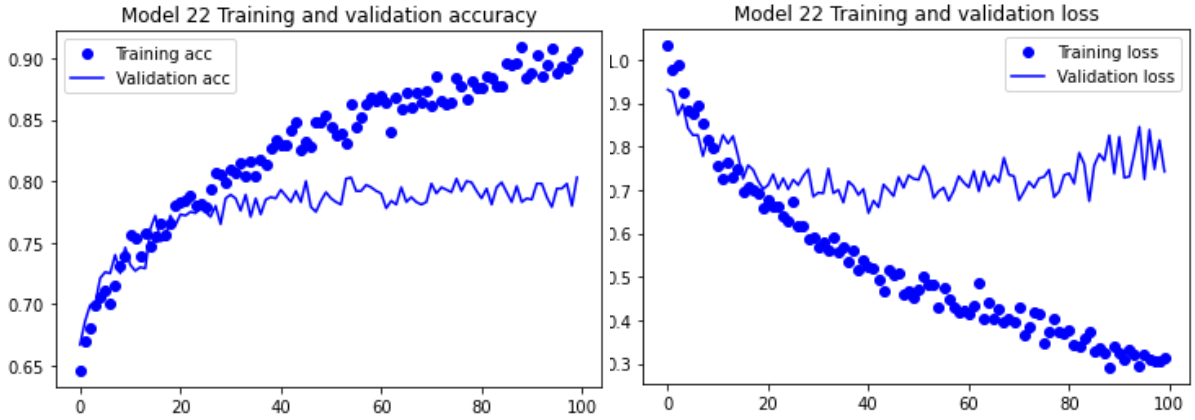
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
vgg16	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 21 is trained with **vgg16 (pre-trained model)** and **1 fully-connected layer**. **Data augmentation** is included to reduce overfitting. The validation accuracy achieved is about 65.90% and the validation accuracy is still increasing. With data augmentation, overfitting is reduced. **In the future build from scratch models, data augmentation will be used.**

3.2.3 Model 22

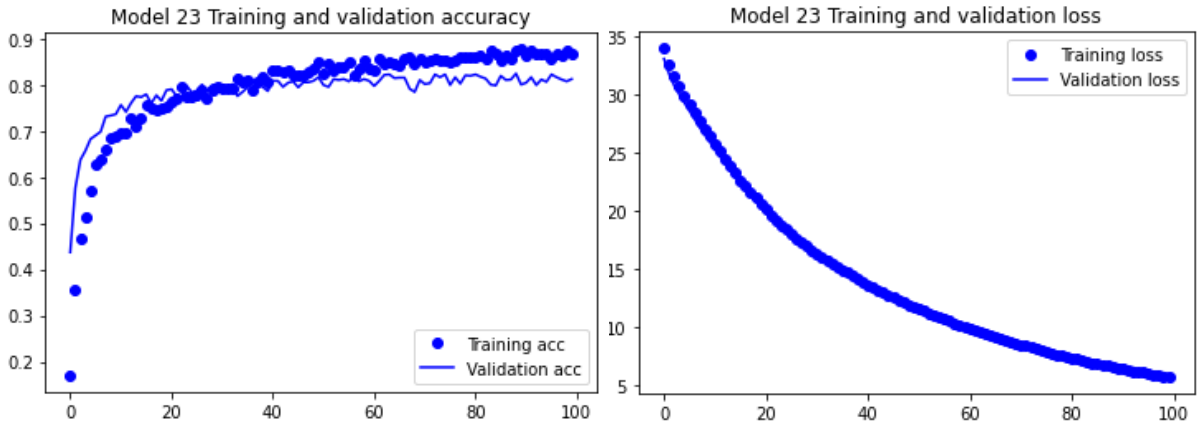
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Vgg16 (unfreeze block 5)	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 22 is trained with **vgg16 (pre-trained model) and 1 fully-connected layer**. In this model, I have **unfreeze the 5th block in the vgg16 model**. The validation accuracy achieved is about 79% with overfitting occurring within 40 epochs. This is an increase of about 13% validation accuracy as compared to model 21. More have to be done to further reduce overfitting.

3.2.4 Model 23

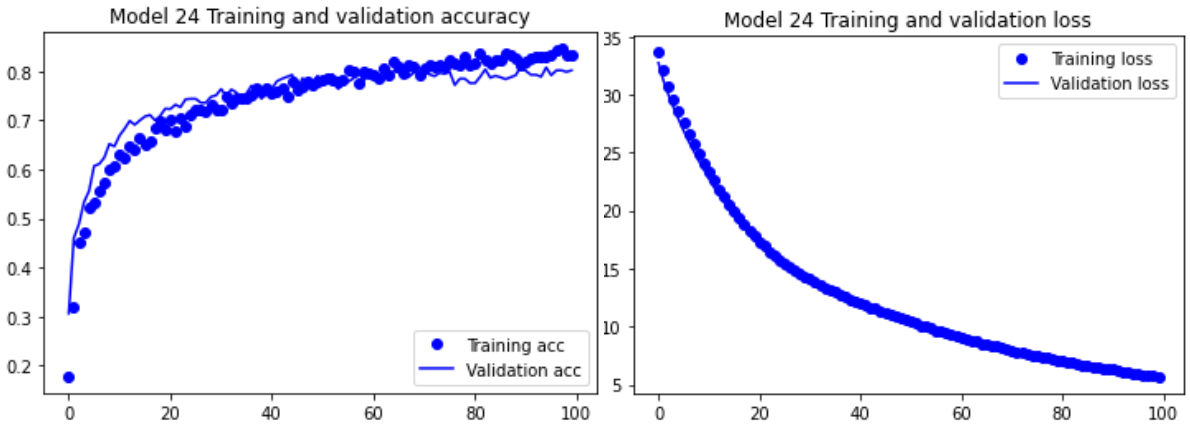
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Vgg16 (unfreeze block 5)	-	-	-	-	-
Fully Connected	256	Relu	-	**0.001**	**0.001**
Dropout	-	-	-	-	-
Fully Connected	**256**	**Relu**	**-**	**0.001**	**0.001**
Output	10	softmax	-	-	-



Model 23 is trained with **vgg16 (pre-trained model) and 2 fully-connected layers**. In this model, I have unfreeze the 5th block in the vgg16 model and **increase one more fully-connected layer**. **Both the fully-connected layers are added to both L1 and L2 regularizers**. The validation accuracy achieved is about 81.60% with reduced overfitting. This is an increase of about 2% validation accuracy as compared to model 22.

3.2.5 Model 24

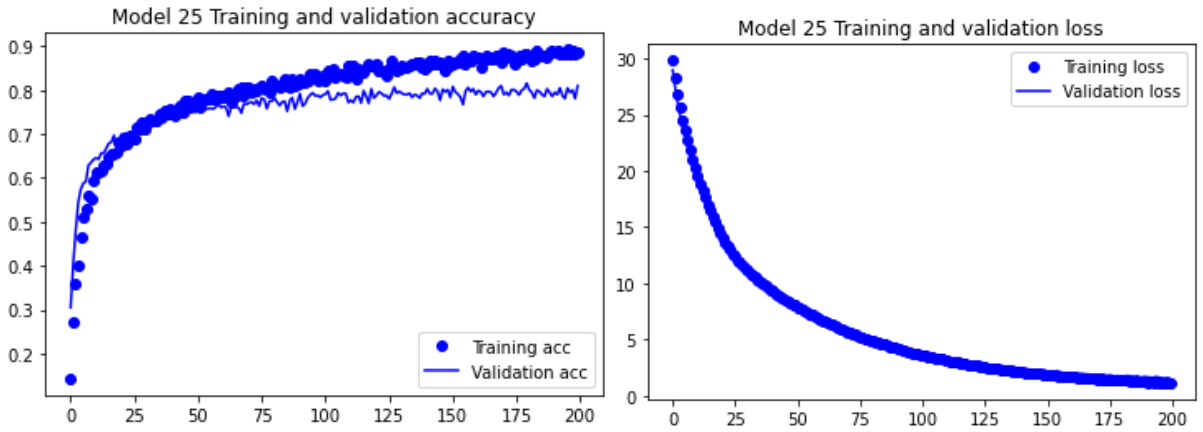
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Vgg16 (unfreeze block 5)	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Output	10	softmax	-	-	-



In Model 24, I have reduced the dropout from 0.5 to 0.2 to experiment and determine the differences. The validation accuracy achieved is about 81%. This is a slight decrease of about 0.2% validation accuracy as compared to model 23.

3.2.6 Model 25

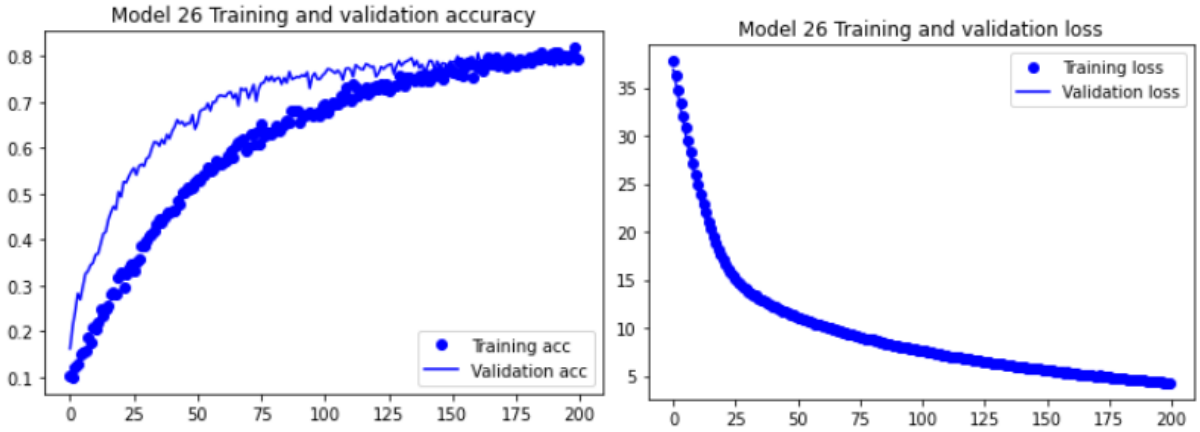
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Vgg16 (unfreeze block 5)	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Fully Connected	256	Relu	-	0.001	0.001
Output	10	softmax	-	-	-



In Model 25, I have changed the **dropout from 0.2 to 0.5** since decreasing it will only reduce validation accuracy. I have **added an additional fully-connected layer with 256 neurons and L1 and L2 regularizers**. The validation accuracy achieved is about 81.60%. This is about the same validation accuracy as compared to Model 23 (highest validation accuracy reached).

3.2.7 Model 26

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
Vgg16 (unfreeze block 5)	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-

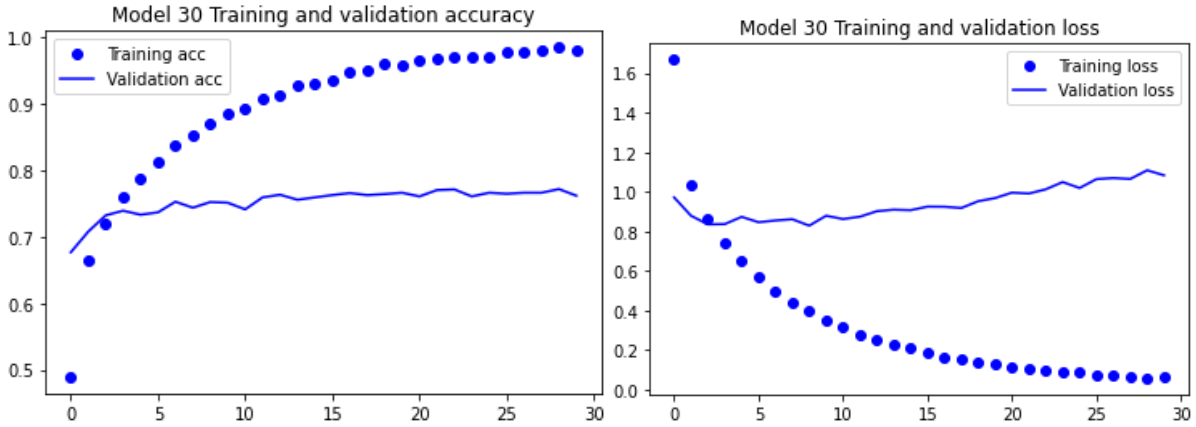


In Model 26, I have added two dropouts, each to the other two fully-connected layers to further reduce overfitting. The validation accuracy achieved is about 80.60%. This is about 1% decrease in validation accuracy from Model 26. Since there is no further improvement in validation accuracy after many attempts, I will stop here for the vgg16 model.

3.3 InceptionV3

3.3.1 Model 30

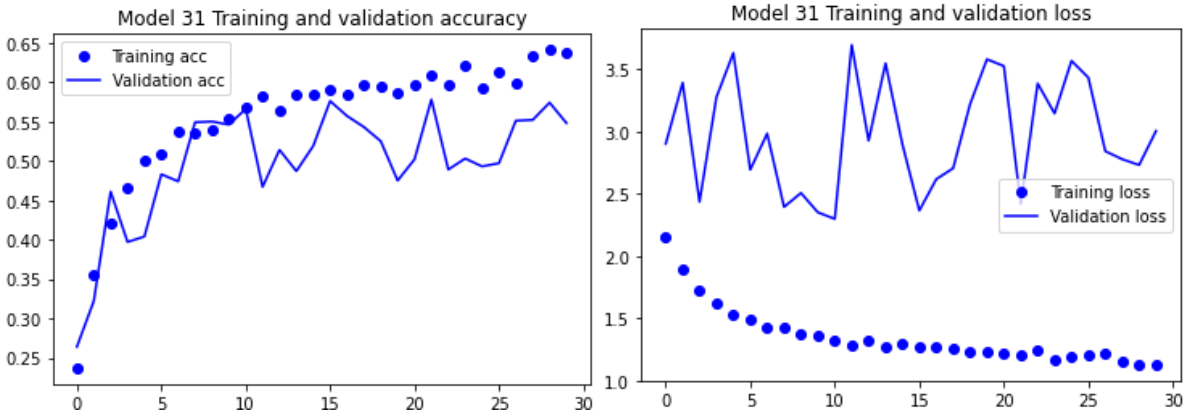
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 30 is trained with **InceptionV3 (pre-trained model)** and **1 fully-connected layer**. The validation accuracy achieved is about 76% and the model begins to overfit within 10 epochs. The validation started out high because the pre-trained model is trained with many images.

3.3.2 Model 31

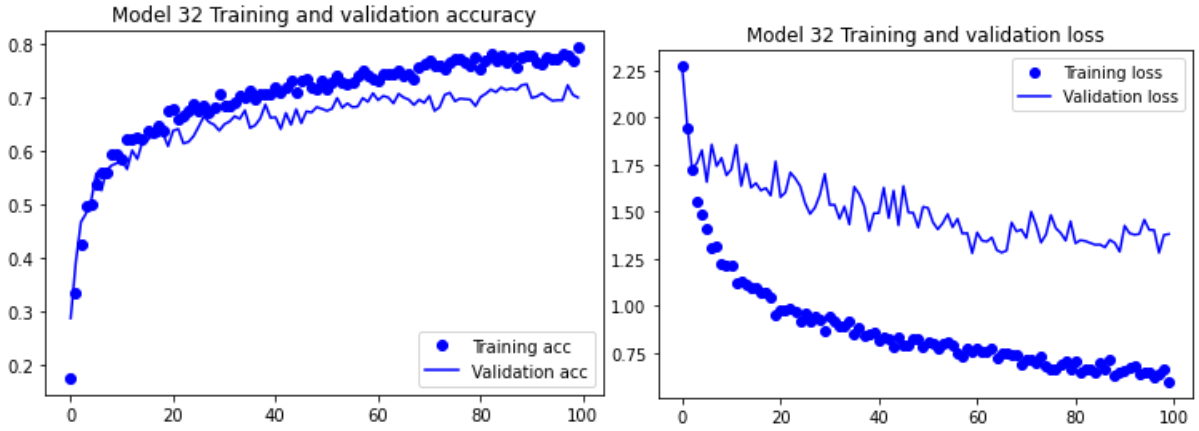
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 31 is trained with **inceptionv3 (pre-trained model)** and **1 fully-connected layer** with **data augmentation**. The validation accuracy achieved is about 50% and the validation accuracy is closer to training accuracy. This is a decrease of about 20% in terms of validation accuracy as compared to model 30. With **data augmentation**, overfitting is reduced. However, the validation loss is fluctuating and is not decreasing. **In the future build from scratch models, data augmentation will be used.**

3.3.3 Model 32

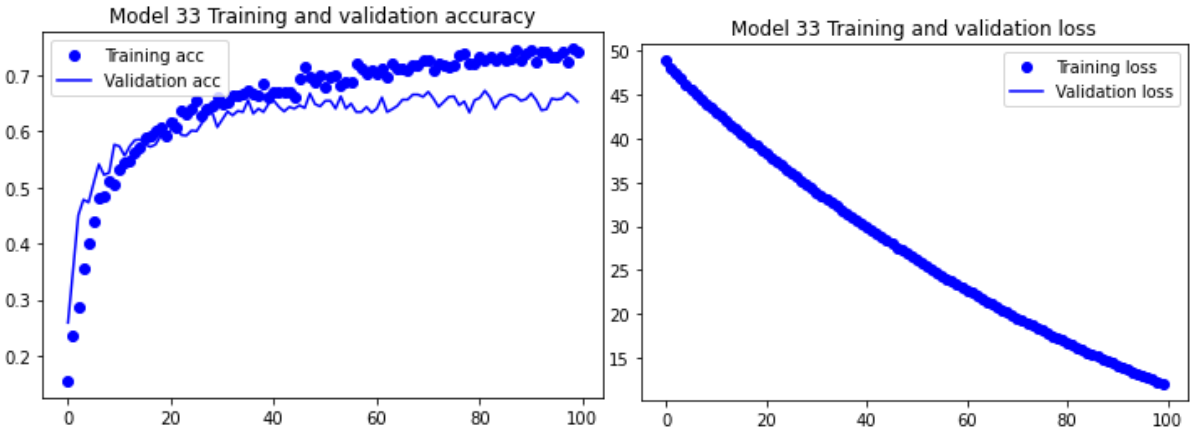
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3 (Unfreeze mixed 8 onwards)	-	-	-	-	-
Fully Connected	256	Relu	-	-	-
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



Model 32 is trained with **InceptionV3 (pre-trained model) and 1 fully-connected layer**. In this model, I have **unfreeze the mixed 8 onwards in the InceptionV3 model**. The validation accuracy achieved is about 70% with reduced overfitting. This is an increase of about 20% in terms of validation accuracy as compared to model 31.

3.3.4 Model 33

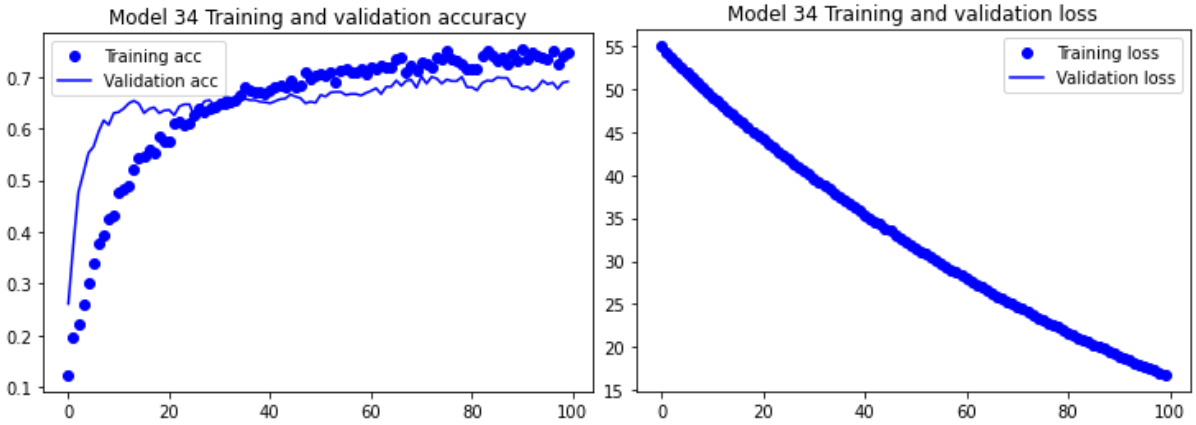
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3 (Unfreeze mixed 8 onwards)	-	-	-	-	-
Fully Connected	256	Relu	-	**0.001**	**0.001**
Dropout	-	-	-	-	-
Fully Connected	**256**	**Relu**	**-**	**0.001**	**0.001**
Dropout	**-**	**-**	**-**	**-**	**-**
Output	10	softmax	-	-	-



Model 32 is trained with **InceptionV3 (pre-trained model) and 2 fully-connected layers**. In this model, I have unfreeze the mixed 8 onwards in the InceptionV3 model and added one more fully-connected layer with 256 neurons and a dropout of 0.5. The validation accuracy achieved is about 65% with reduced overfitting. This is a decrease of about 5% in terms of validation accuracy as compared to model 32. This could be due to the complexity of the model which causes overfitting to occur at around epoch 40.

3.3.5 Model 34

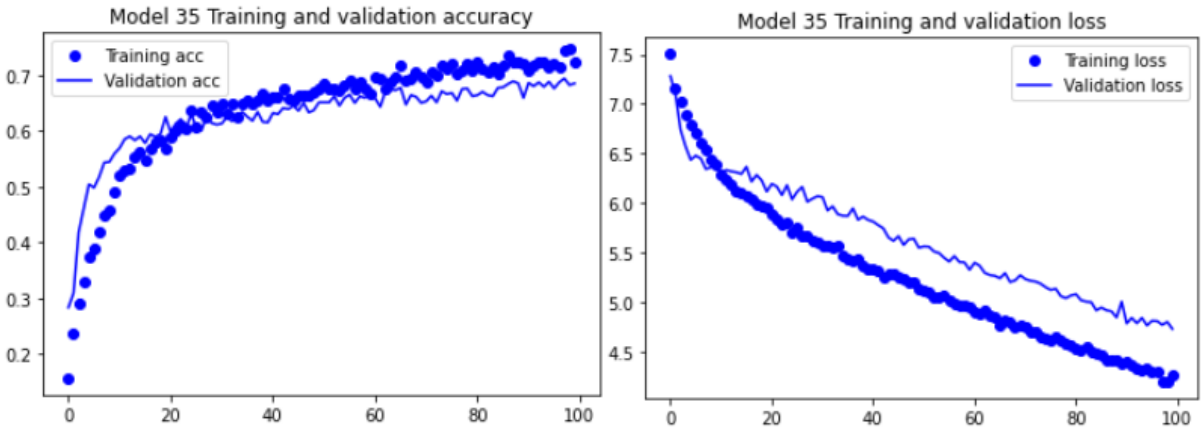
Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3 (Unfreeze mixed 8 onwards)	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	256	Relu	-	0.001	0.001
Dropout	-	-	-	-	-
Fully Connected	**512**	**Relu**	**-**	**0.001**	**0.001**
Dropout	**-**	**-**	**-**	**-**	**-**
Output	10	softmax	-	-	-



To try with more experiment results, I have **added one more fully-connected layer to model 34 with 512 neurons and L1 and L2 regularizers**. The validation remains around 69% and overfitting occurs within 60 epochs. This is when I realized that the more complex this model gets, the lower the validation accuracy.

3.3.6 Model 35

Layer	Number of Neurons	Activation Function	Kernel Size	Regularizer (L1)	Regularizer (L2)
InceptionV3 (Unfreeze mixed 8 onwards)	-	-	-	-	-
Fully Connected	256	Relu	-	-	0.01
Dropout	-	-	-	-	-
Output	10	softmax	-	-	-



To try with more experiment results, I have **decreased the complexity of the model.** Model 35 is trained with InceptionV3 (unfreeze from mixed 8 onwards) and a fully-connected layer with only L2 regularizer. The validation accuracy is at about 68% and overfitting occurs within 80 epochs. This is when I realized that with InceptionV3, the validation accuracy remains at 65-72% and I cannot improve it further.

Evaluate models using Test images

3.4 Build from Scratch

Model	Conv Layer	Dense Layer	Data Augmentation	Regularizer	Dropout	Learning rate	Testing Accuracy
1	1	1	-	-	-	0.001	30.50%
2	2	1	-	-	-	0.001	33.00%
3	3	1	-	-	-	0.001	40.50%
4	4	1	-	-	-	0.001	45.50%
5	5	1	-	-	-	0.001	49.00%
6	5 (increased neurons)	1	-	-	-	0.001	46.00%
7	4 (increased neurons)	1	-	-	-	0.001	45.00%
8	5 (increased neurons)	2 (decreased neurons)	-	-	-	0.001	44.50%
9	5 (increased neurons)	2 (decreased neurons)	yes	-	-	0.001	44.00%
10	5 (increased neurons)	2 (decreased neurons)	yes	-	0.5	0.001	50.00%
11	5 (increased neurons)	2 (decreased neurons)	yes	-	0.5	0.0001	55.00%
12	5 (increased neurons)	2 (decreased neurons)	yes	L1 and L2 0.001	0.5	0.0001	42.00%
13	5 (increased neurons)	2 (decreased neurons)	yes	L1 and L2 0.001	0.5	0.0001	48.50%
14	5 (increased neurons)	2 (decreased neurons)	yes	L1 and L2 0.001	0.5	0.0001	70.50%
15	5	2	yes	L1 and L2	0.5	0.0001	66.00%

	(increased neurons)	(decreased neurons)		0.001			
16	5 (increased neurons)	2 (decreased neurons)	yes	L1 and L2 0.001	0.5	0.0001	72.50%
17	5 (increased neurons)	2 (decreased neurons)	yes	L1 and L2 0.001	0.5	0.0001	72.50%

In the testing phase, 500 testing images were passed into every model that I have trained (model 1 to model 17). In the table above, Model 16 and 17 yields the same testing accuracy at 72.50% and they have the highest testing accuracy as compared to the rest of the models. In addition, the validation accuracy is close to the training accuracy, indicating that overfitting is reduced. In this case, I would select model 16 to be my best model for build from scratch models because model 16 requires lesser time for training.

3.5 Vgg16

Model	Conv Layer	Dense Layer	Data Augmentation	Regularizer	Dropout	Testing Accuracy
20	vgg16	1	-	-	0.5	-
21	vgg16	1	yes	-	-	-
22	Vgg16 (unfreeze)	1	yes	-	-	79.50%
23	Vgg16 (unfreeze	2	yes	L1 and L2 0.001	0.5	50.50%
24	Vgg16 (unfreeze	2	yes	L1 and L2 0.001	0.5	78.50%
25	Vgg16 (unfreeze	3	yes	L1 and L2 0.001	0.5	80.00%
26	Vgg16 (unfreeze	3	yes	L1 and L2 0.001	0.5	83.00%

In the testing phase, 500 testing images were passed into every model that I have trained (model 20 to model 26). In the table above, Model 26 yields the highest testing accuracy at 83.00% as compared to the rest of the model. In addition, the validation accuracy is close to the training accuracy, indicating that overfitting is reduced. In this case, model 26 will represent the best model in the Vgg16 model building process.

3.6 InceptionV3

Model	Conv Layer	Dense Layer	Data Augmentation	Regularizer	Dropout	Testing Accuracy
30	InceptionV3	1	-	-	0.5	-
31	InceptionV3	1	yes	-	0.5	53.50%
32	InceptionV3 (unfreeze)	1	yes	-	0.5	64.00%
33	InceptionV3 (unfreeze)	2	yes	0.001	0.5	66.00%
34	InceptionV3 (unfreeze)	3 (increased neurons)	yes	0.001	0.5	72.50%
35	InceptionV3	1	yes	0.01	0.5	76.00%

In the testing phase, 500 testing images were passed into every model that I have trained (model 20 to model 26). In the table above, Model 35 yields the highest testing accuracy at 76.00% as compared to the rest of the model. In addition, the validation accuracy is close to the training accuracy, indicating that overfitting is reduced. In this case, model 35 will represent the best model in the InceptionV3 model building process.

3.7 Best Model

Model Name	Model No.	Testing accuracy
Build from scratch	16	72.50%
Vgg16	26	83.00%
InceptionV3	35	76.00%

After selecting the best model from each training phase, the best model is the Vgg16 model which yields the highest testing accuracy at 83.00%. This is 10.50% higher than the build from scratch model and 7% higher than the InceptionV3 model.

4 Use the Best Model to perform classification

After we have selected the best model (vgg16, model 26), I downloaded a few images from the google images to test the prediction of the images with the model. The images that I have downloaded are not seen by the model before to ensure a fair test. The images will be converted to size 150x150 because this is the similar size that we used to train the model. All the images will be converted to a numpy array so that the model is able to "read" it.

4.1 Image 1

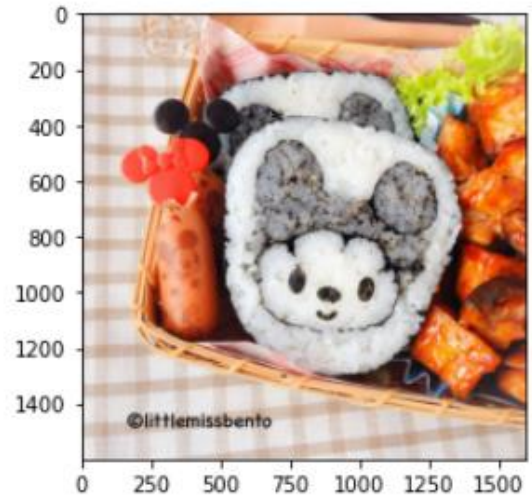

```
The prediction is:  sushi
```

	beet_salad	ceviche	chicken_quesadilla	creme_brulee	garlic_bread	\
0	0.000005	0.000003	2.857398e-07	1.878839e-07	1.328215e-07	

	macaroni_and_cheese	miso_soup	pad_thai	shrimp_and_grits	sushi
0	2.260727e-08	2.463613e-09	1.039404e-08	2.624815e-09	0.999992

Through our naked eyes, we can identify this image to be a sushi briefly. Similarly, with the model, the model can predict this image to be a sushi and the prediction is 99.99% accurate. This shows that the model is very accurate in identifying the images that we have trained.

4.2 Image 2



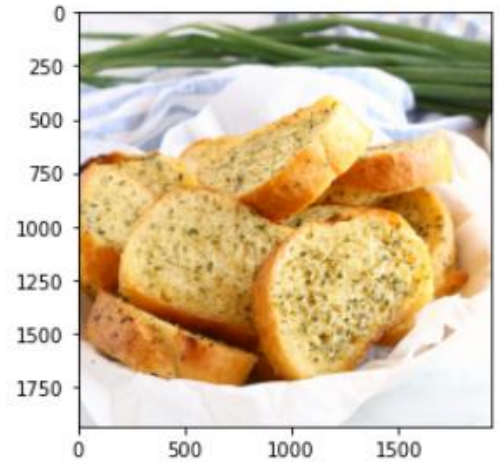

```
The prediction is:  sushi
```

	beet_salad	ceviche	chicken_quesadilla	creme_brulee	garlic_bread	\
0	0.05481	0.069791	0.01855	0.024919	0.031668	

	macaroni_and_cheese	miso_soup	pad_thai	shrimp_and_grits	sushi
0	0.011239	0.005792	0.006586	0.025788	0.750857

Through our naked eyes, we might have trouble recognizing this image to be a sushi because a typical sushi will not have a cartoon image on the sushi rice. However, if we take a longer time to analyze the image, we can guess that this is a sushi image because of the seaweed surrounding the rice. This is a difficult task for the model because not only of the cartoon image, there are different foods surrounding the sushi, which makes it harder for the model to predict. In this case, the model is still able to predict that this image is a sushi with 75.00% accuracy. This is a very accurate prediction.

4.3 Image 3




```
The prediction is:  garlic_bread
```

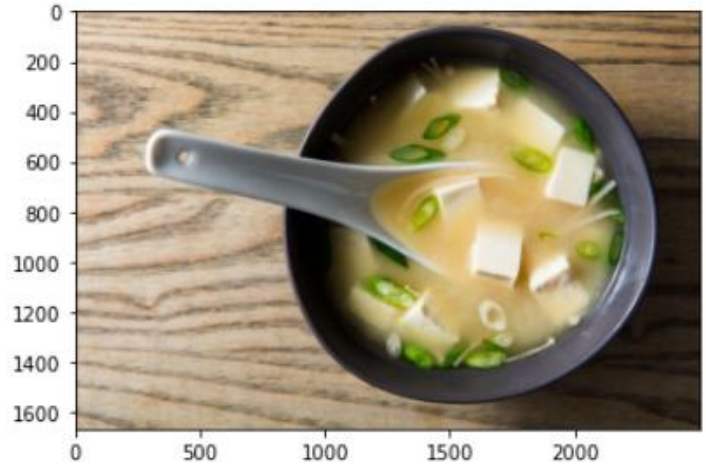
	beet_salad	ceviche	chicken_quesadilla	creme_brulee	garlic_bread	\
0	2.184544e-10	4.459771e-09	0.000139	2.150800e-09	0.999861	

	macaroni_and_cheese	miso_soup	pad_thai	shrimp_and_grits	\
0	2.533126e-07	1.498811e-09	9.706181e-10	4.565725e-08	

	sushi
0	3.316182e-08

Through our naked eyes, we can identify this image to be garlic bread at first glance. Similarly, with the model, the model can predict this image to be a garlic bread and the prediction is 99.98% accurate. This shows that the model is very accurate in identifying the images that we have trained.

4.4 Image 4




```
The prediction is:  miso_soup
```

	beet_salad	ceviche	chicken_quesadilla	creme_brulee	garlic_bread	\
0	6.200786e-12	1.067547e-11	8.950566e-12	3.992036e-09	2.149058e-12	

	macaroni_and_cheese	miso_soup	pad_thai	shrimp_and_grits	\
0	1.046857e-10	1.0	1.458036e-12	5.738937e-12	

	sushi
0	6.816271e-13

Through our naked eyes, we can identify this image to be miso soup at first glance. Similarly, with the model, the model can predict this image to be a miso soup and the prediction is 100% accurate. This shows that the model is very accurate in identifying the images that we have trained.

4.5 Image 5


```
The prediction is:  chicken_quesadilla
```

	beet_salad	ceviche	chicken_quesadilla	creme_brulee	garlic_bread	\
0	0.00375	0.020686	0.942464	0.001579	0.01328	

	macaroni_and_cheese	miso_soup	pad_thai	shrimp_and_grits	sushi
0	0.002049	0.000756	0.004512	0.001917	0.009008

Through our naked eyes, we can easily identify this picture to be a bowl of chicken rice at first glance. However, because we did not train the model to recognize chicken rice, the prediction of this picture becomes inaccurate. Interestingly, because of the chicken on the chicken rice, the model can predict the closest food that we have trained, which is chicken quesadilla at 94.24% accuracy. It is expected for this prediction to be inaccurate because we did not train the model to recognize chicken rice.

5 Summary

In conclusion, the model has significantly improved the recognition accuracy with the help of fine-tuning such as dropout, regularizers, learning rate, batch-size, step-per-epoch, data augmentation etc. The final result showed that the vgg16 model has a better prediction rate at 83.00% as compared to the build from scratch model at 72.50% and the InceptionV3 model at 76.00% because of the residual structure. Although this result is certainly not the best possible result as more fine tuning could be done to better improve the model.

Here are some suggestions that I can explore more on this food recognition problem for further improvement. Firstly, the time we spent on training the model is long due the complexity of the model and the nature of deep learning. In the future, I can use early stopping to halt the training when overfitting occurs so that I do not have to wait, for example, 500 epochs when overfitting occurs at about 200 epochs. This will save a lot more time and more models could be trained during this time. Secondly, the result we got from the Vgg16 model might not necessarily be the best model. Things like unfreezing more or lesser vgg16 blocks can significantly change the accuracy of the model. However, due to time constraints, I only unfreeze the 5th block of the vgg16 model. Lastly, we could further improve the model performance by adding bounding boxes to the images. Since some images may contain other food items that is not supposed to be in the food category (e.g. there are sausages beside the garlic bread image), the model might misinterpret the image, resulting in lower accuracy. This noisy element could be eliminated by the use of bounding boxes to the images and this could potentially be another possibility of increasing the accuracy of the images.

6 References

DanB. (2017). *Food 101*. Retrieved from Kaggel: https://www.kaggle.com/dansbecker/food-101

Giovanni Maria Farinella, D. A. (2016, July 08). Computers in Biology and Medicine. *Retrieval and classification of food images*, 17. Retrieved from https://iplab.dmi.unict.it/UNICT-FD1200/FinalOnline.pdf