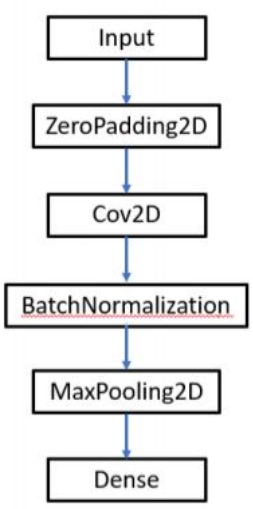
# ML-CSE-543 – Machine Learning

## Assignment 6 Analysis

**By: Rajat Agarwal and Abhinav Saurabh** Date: 4th Dec 2020

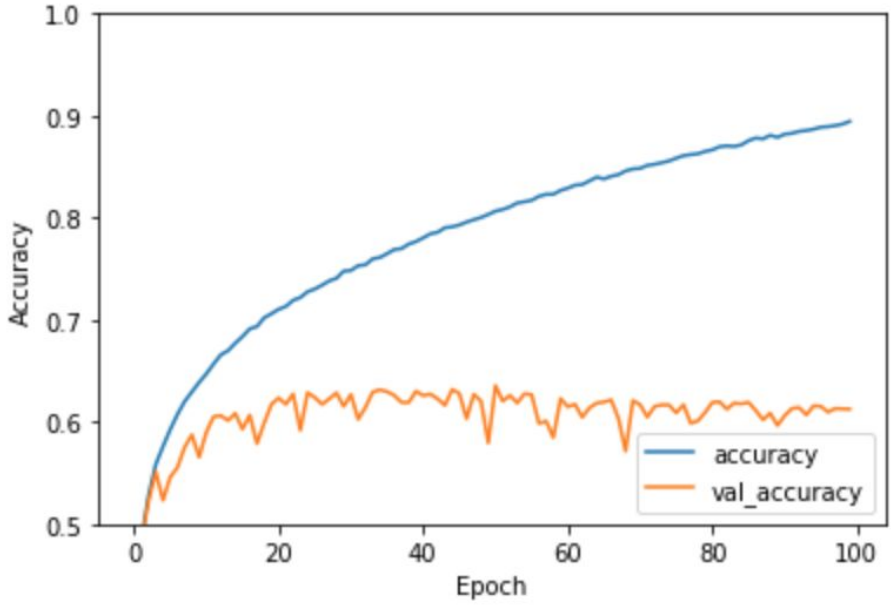
# Q1



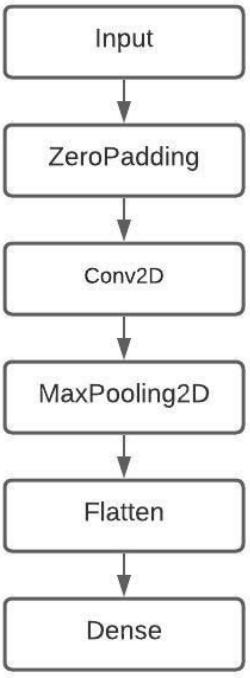

- This is the baseline architecture of the model given. Here we obtained the training accuracy of 89.44% for 100 epochs.
- The testing accuracy obtained here was 61.22% for 100 epochs.
- Model appears to be slightly overfit.
- We have used the below configuration of layers.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d (ZeroPadding2 (None, 34, 34, 3)         0
_________________________________________________________________
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization (BatchNo (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 10)                81930
=================================================================
Total params: 82,954
Trainable params: 82,890
Non-trainable params: 64
_________________________________________________________________
```

- Below is the training and testing accuracy graph.




**3 a)**

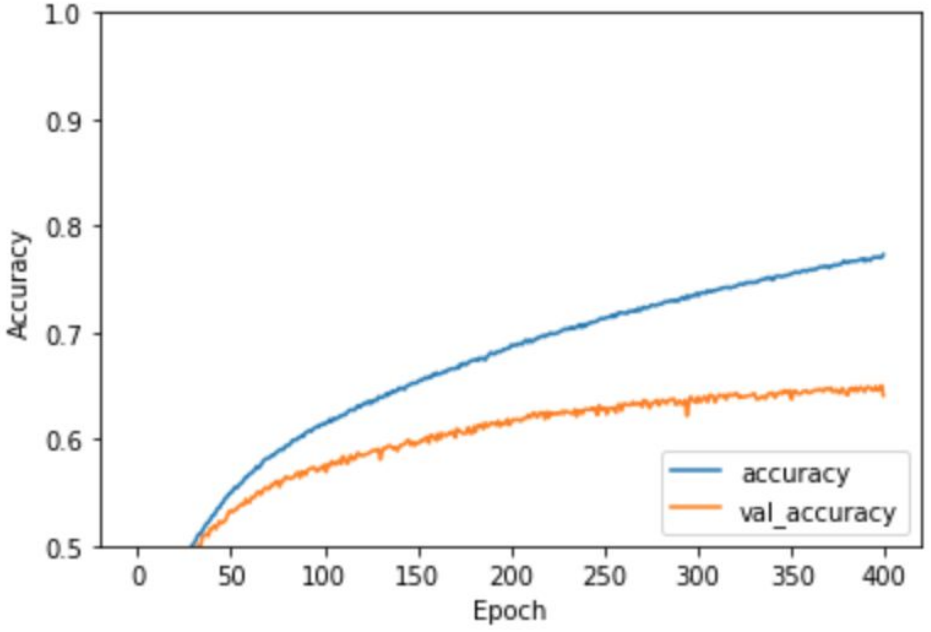



- This is the first variation of the architecture. Here we remove the batch normalization from the baseline architecture.
- The training accuracy obtained  at 100 epochs is 61.40%.
- The testing accuracy obtained at 100 epochs is 57.40%
- On further training for 400 epochs accuracy increases.

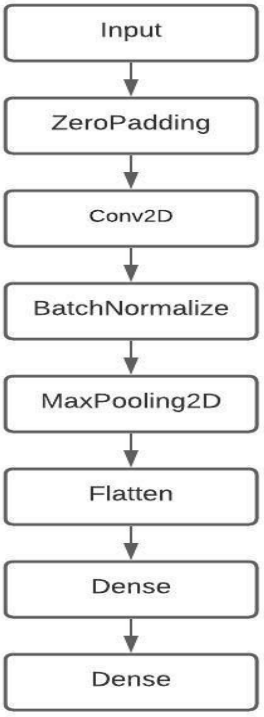
- The training accuracy at 400 epochs obtained is 77.29%.
- The testing accuracy at 400 epochs obtained is 64.08%.
- We have used the below configuration for layers.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d (ZeroPadding2 (None, 34, 34, 3)         0
_________________________________________________________________
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 10)                81930
=================================================================
Total params: 82,826
Trainable params: 82,826
Non-trainable params: 0
_________________________________________________________________
```

- Below is the training and testing accuracy graph for (a)



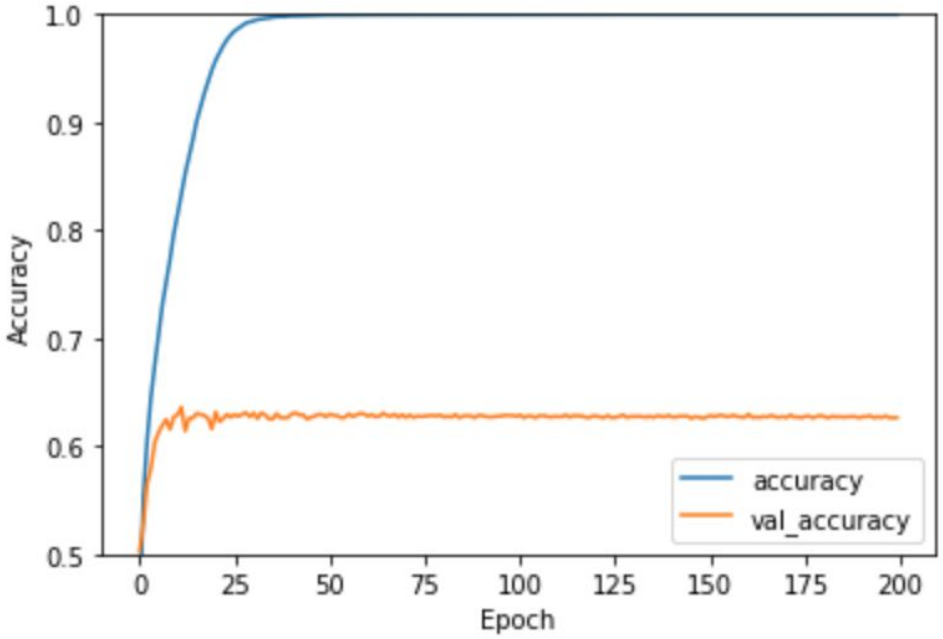
**b)**



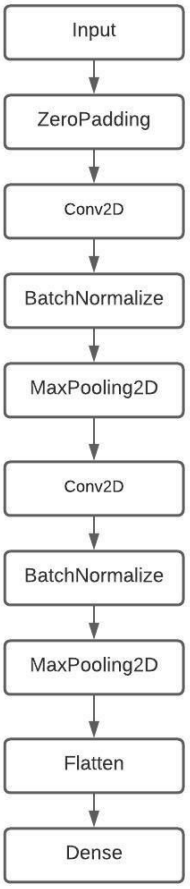

- This is the second variation of the architecture. Here we add an extra dense layer in the baseline architecture.
- The training accuracy obtained  at 100 epochs is 99.98%.
- The testing accuracy obtained at 100 epochs is 62.76%
- On further training for 200 epochs training accuracy and model becomes more overfit.
- The training accuracy at 200 epochs obtained is 99.99%.
- The testing accuracy at 200 epochs obtained is 62.65%.
- We have used the below configuration for layers.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d (ZeroPadding2 (None, 34, 34, 3)         0
_________________________________________________________________
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization (BatchNo (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 8192)              0
_________________________________________________________________
dense (Dense)                (None, 64)                524352
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 526,026
Trainable params: 525,962
Non-trainable params: 64
```

- Below is the training and testing accuracy graph for (b)




**c)**

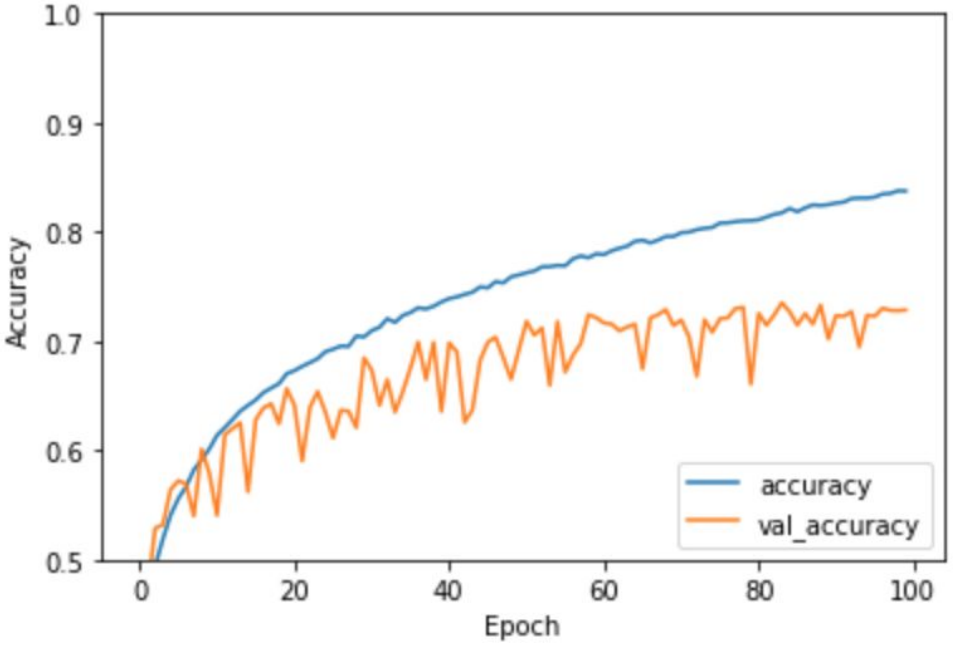



- This is the Third variation of the architecture. Here we add an extra **1 blocks of (Cov2D -> BatchNorm2D->MaxPooling2D) in the baseline architecture**.
- The training accuracy obtained  at 100 epochs is 84.23%.

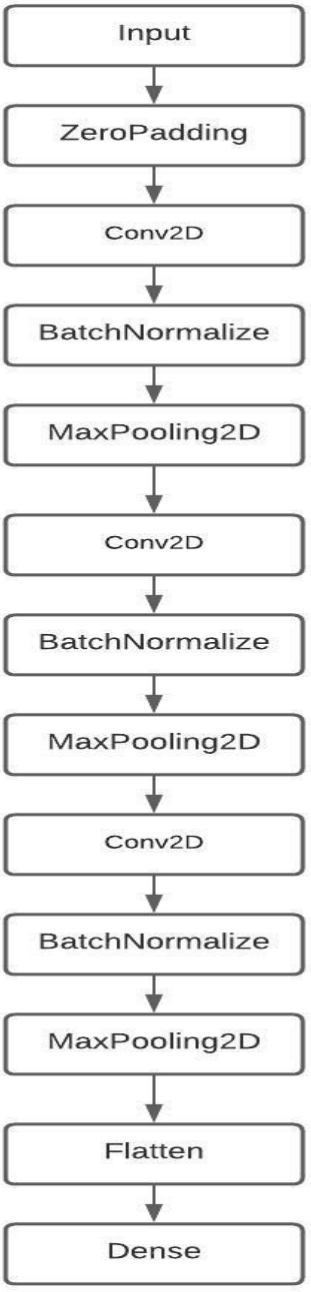
- The testing accuracy obtained at 100 epochs is 70.11%
- On further training for 200 epochs the model becomes more overfit.
- The training accuracy at 200 epochs obtained is 89.67%.
- The testing accuracy at 200 epochs obtained is 69.12%.
- We have used dropout 1 time after block of (Cov2D -> BatchNorm2D->MaxPooling2D) to reduce the overfitting.
- We have used the below configuration for layers.

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d_4 (ZeroPaddin (None, 34, 34, 3)         0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization_8 (Batch (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 16, 16, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 14, 14, 64)        18496
_________________________________________________________________
batch_normalization_9 (Batch (None, 14, 14, 64)        256
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
flatten_4 (Flatten)          (None, 3136)              0
_________________________________________________________________
dense_4 (Dense)              (None, 10)                31370
=================================================================
Total params: 51,146
Trainable params: 50,954
Non-trainable params: 192
_________________________________________________________________
```

- Below is the training and testing accuracy graph for (c)



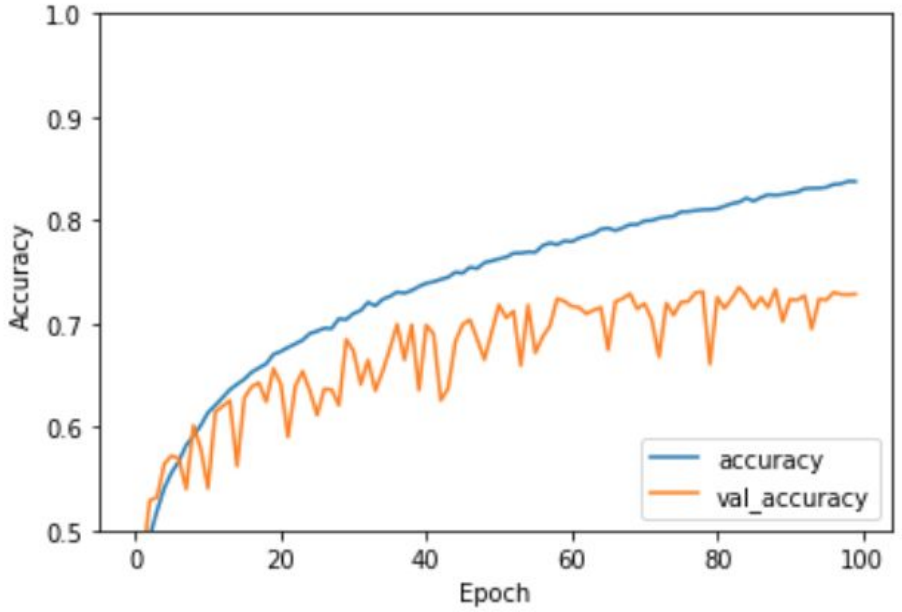
**d)**




- This is the Third variation of the architecture. Here we add an extra **2 blocks of (Cov2D -> BatchNorm2D->MaxPooling2D) in the baseline architecture**.
- The training accuracy obtained at 50 epochs is 76.07%.
- The testing accuracy obtained at 50 epochs is 69.20%
- On further training for 100 epochs accuracy increases.
- The training accuracy at 100 epochs obtained is 83.75%.
- The testing accuracy at 100 epochs obtained is 72.85 %.

- We have used dropout 2 times after blocks of (Cov2D -> BatchNorm2D->MaxPooling2D) to reduce the overfitting.

- Below is the training and testing accuracy graph for (d)




- We have used the below configuration for layers.

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d_6 (ZeroPaddin (None, 34, 34, 3)         0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization_18 (Batc (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d_18 (MaxPooling (None, 16, 16, 32)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_19 (Conv2D)           (None, 14, 14, 64)        18496
_________________________________________________________________
batch_normalization_19 (Batc (None, 14, 14, 64)        256
_________________________________________________________________
max_pooling2d_19 (MaxPooling (None, 7, 7, 64)          0
_________________________________________________________________
dropout_3 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
conv2d_20 (Conv2D)           (None, 5, 5, 128)         73856
_________________________________________________________________
batch_normalization_20 (Batc (None, 5, 5, 128)         512
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 2, 2, 128)         0
_________________________________________________________________
flatten_6 (Flatten)          (None, 512)               0
_________________________________________________________________
dense_6 (Dense)              (None, 10)                5130
=================================================================
Total params: 99,274
Trainable params: 98,826
Non-trainable params: 448
```

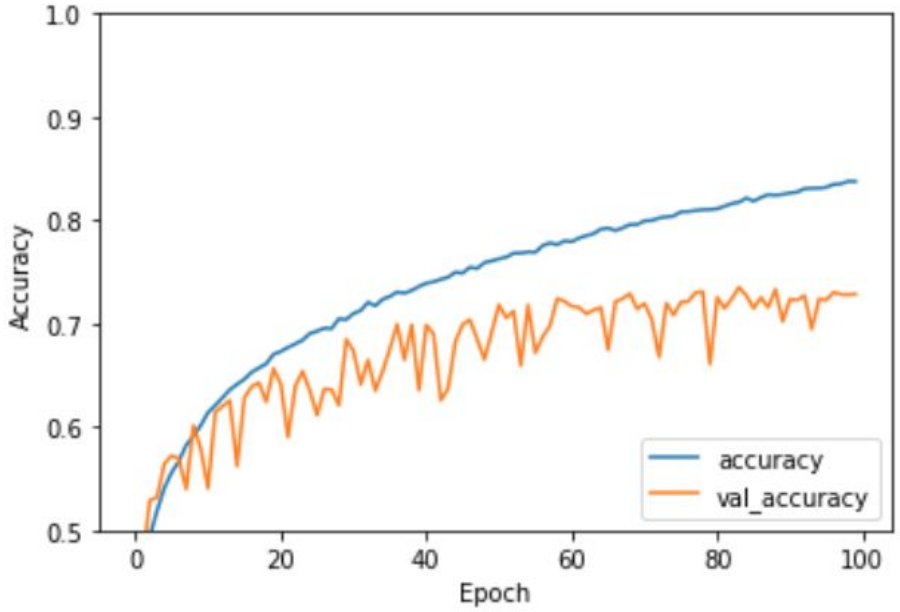
| Model | Train Accuracy | Test Accuracy |
|---|---|---|
| Base Model | 89.44% | 61.22%  for 100 epochs |
| Without BatchNormalization | 77.29%. | 64.08% for 400 epochs |
| 2 Dense Layers | 99.99%. | 62.65% for 200 epochs |
| With 2 blocks 2Covd | 89.67% | 69.12% for 200 epochs |
| With 3 blocks 2Covd | 83.75% | 72.85 % for 100 epochs |

## 4.

- The best model we have obtained is in the 4th architecture where **3 blocks of (Cov2D -> BatchNorm2D->MaxPooling2D) is present.**
- We have used the below configuration for layers.

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d_6 (ZeroPaddin (None, 34, 34, 3)         0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization_18 (Batc (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d_18 (MaxPooling (None, 16, 16, 32)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_19 (Conv2D)           (None, 14, 14, 64)        18496
_________________________________________________________________
batch_normalization_19 (Batc (None, 14, 14, 64)        256
_________________________________________________________________
max_pooling2d_19 (MaxPooling (None, 7, 7, 64)          0
_________________________________________________________________
dropout_3 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
conv2d_20 (Conv2D)           (None, 5, 5, 128)         73856
_________________________________________________________________
batch_normalization_20 (Batc (None, 5, 5, 128)         512
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 2, 2, 128)         0
_________________________________________________________________
flatten_6 (Flatten)          (None, 512)               0
_________________________________________________________________
dense_6 (Dense)              (None, 10)                5130
=================================================================
Total params: 99,274
Trainable params: 98,826
Non-trainable params: 448
```

- Below is the training and testing accuracy graph for best model



# Q2

Steps:

1. Load the dataset into a dataframe and extract X and y

2. Split the dataset into 80:20 ratio

3. Data preprocessing:
    a. Convert text into lowercase
    b. Filter out punctuation marks and special characters using Tokenizer from Keras

4. Create sequences using tokenizer.texts_to_sequences method for both X_train and X_test data

5. Pad the sequences to make sentences to equal length using pad_sequences from Keras

6. Prepare Glove embedding using 6B.50d file
    a. Load glove embedding and prepare a dictionary
    b. Create embedding matrix for each word in the training dataset using embeddings_index

7. Embedding layer, weights are set to the embedding matrix created in last step, input_dim is set to training data vocabulary length, output_dim is set to 50 (hyperparameter), input_lenght is set to 50 because padding is done to 50

**Model 1 (Bidirectional RNN)**

8. Define the model:
    a. Model is build sequentially using keras.Sequential
    b. Text tokenizer
    c. Embedding layer
    d. Bidirectional RNN is used with 16 neurons. This propagates the input forward and backwards through the RNN layer and then concatenates the final output. Dropout of 0.3 is also added to these layers.
    e. After the RNN has converted the sequence to a single vector the two dense layers, do some final processing, and convert from this vector representation to a binary classification output.

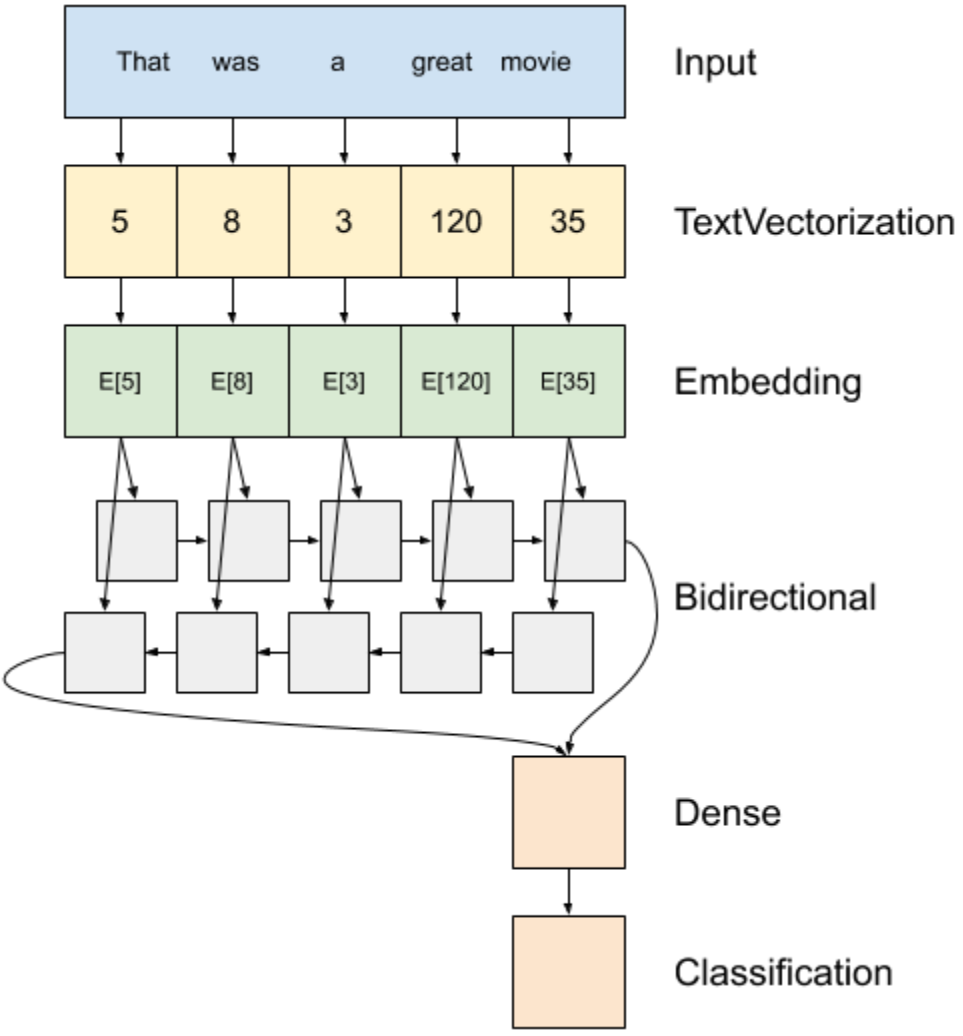



Fig 2.i Diagram for the model used

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 50, 50)            138550
_________________________________________________________________
bidirectional (Bidirectional (None, 50, 32)            2144
_________________________________________________________________
dense (Dense)                (None, 50, 64)            2112
_________________________________________________________________
dense_1 (Dense)              (None, 50, 1)             65
=================================================================
Total params: 142,871
Trainable params: 4,321
Non-trainable params: 138,550
_________________________________________________________________
```

Fig 2.ii Model summary

| | Accuracy | Loss |
|---|---|---|
| **Training** | 85.76 | 0.34 |
| **Testing** | 71.81 | 0.57 |

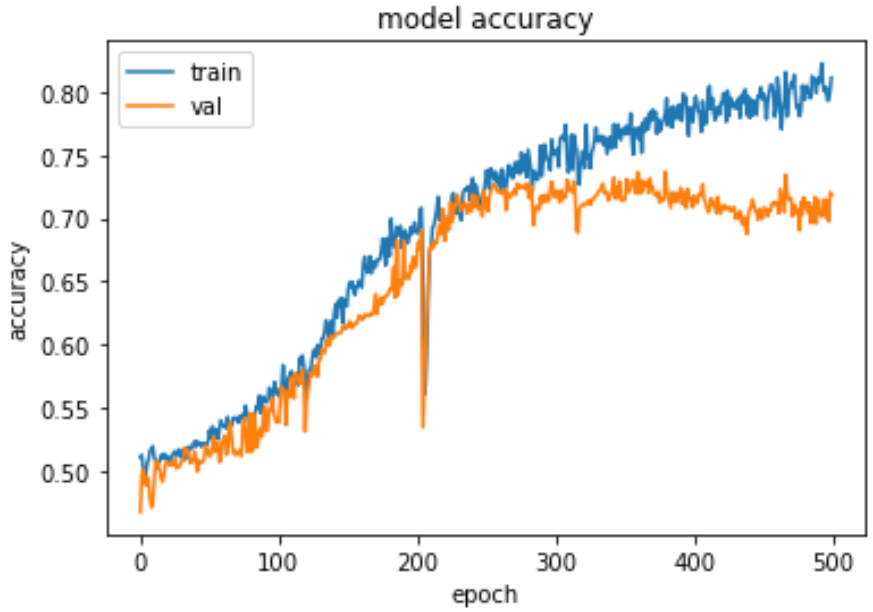



Fig 2.iii) Model Accuracy vs Epochs

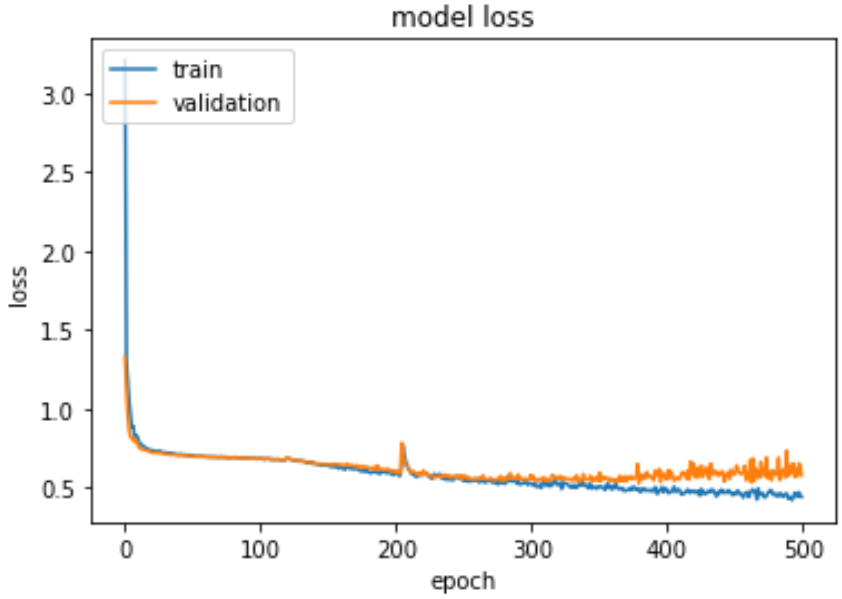


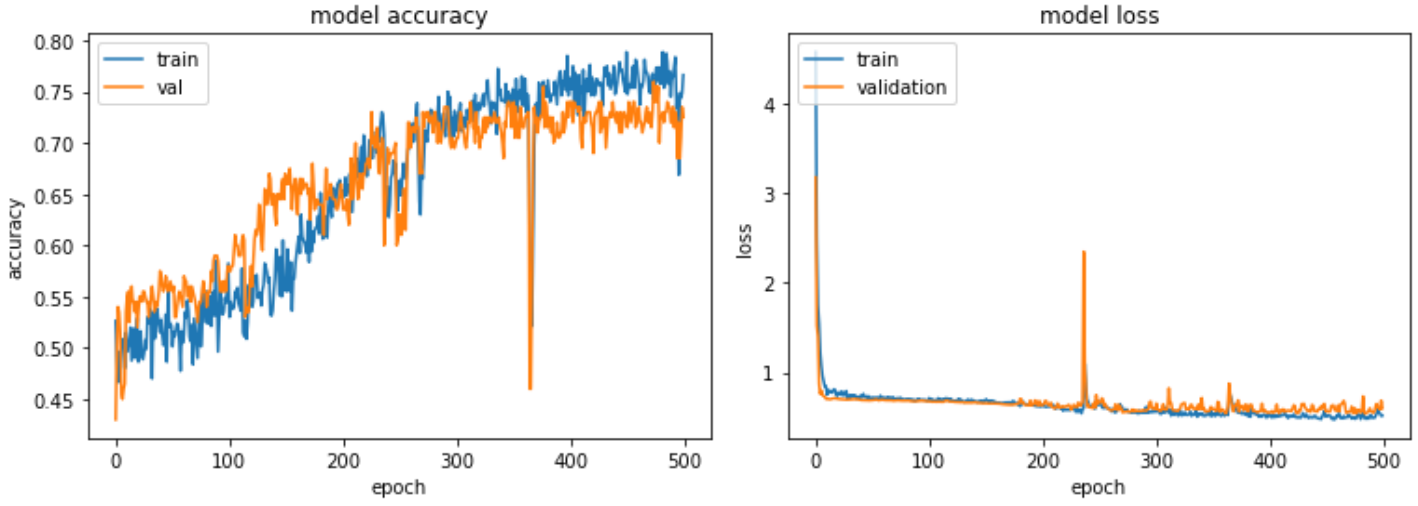

Fig 2.iv) Model Loss vs Epochs

***The main advantage to a bidirectional RNN is that the signal from the beginning of the input doesn't need to be processed all the way through every timestep to affect the output.***

**Model 2 (Stacking two layer of Bidirectional RNNs)**

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 50, 50)            138550
_________________________________________________________________
bidirectional_1 (Bidirection (None, 50, 32)            2144
_________________________________________________________________
bidirectional_2 (Bidirection (None, 32)                1568
_________________________________________________________________
dense_2 (Dense)              (None, 64)                2112
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
=================================================================
Total params: 144,439
Trainable params: 5,889
Non-trainable params: 138,550
_________________________________________________________________
```

| | Accuracy | Loss |
|---|---|---|
| **Training** | 80.87 | 0.42 |
| **Testing** | 72.56 | 0.60 |




We don't see any significant improvement with stacking therefore Model 1 is finalized.

**Model Baseline (Forward RNN)**

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 50, 50)            138550
_________________________________________________________________
simple_rnn_3 (SimpleRNN)     (None, 16)                1072
_________________________________________________________________
dense_4 (Dense)              (None, 64)                1088
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 65
=================================================================
Total params: 140,775
Trainable params: 2,225
Non-trainable params: 138,550
```

| | Accuracy | Loss |
|---|---|---|
| **Training** | 51.02 | 0.69 |
| **Testing** | 46.23 | 0.68 |