Assignment 3

Shubham Singla, Rupesh Kashyap, and Harsh Kumar

1 Introduction

Following report contains observation for the 3 data sets namely, MNIST, Signal data and Amazon Helpfulness ratio dataset. The observations contain the performance of different machine learning algorithms on the above mentioned data sets. Different model architecture were tried on these datasets to achieve good results which have been summarized below in the form of graphs, accuracy, precision, recall, f-score.

2 MNIST DataSet

This is a multi-class classification problem where the feature vectors are sparse. Constrained by the memory and computational power, the number of experiments for this data set were reduced. Standardizing the data set with each feature having zero mean and unit variance resulted in unsatisfactory results. Therefore, another normalization technique was used where each example was scaled to have unit norm. Here, assume number of components after PCA to be 80 unless stated.

2.1 Observations

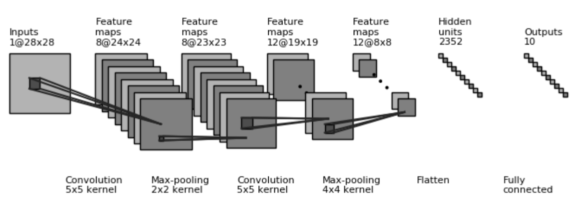

Fig. 1 Visualization of CNN architecture

The test result on logistic model were poor while the training error was small, indicating the distribution of test being different from the training data and a need of model which is more generalized. The multi-layer perceptron with two hidden layers (50,20) performed the best among all the models we tried where we did not take specialty of image data.

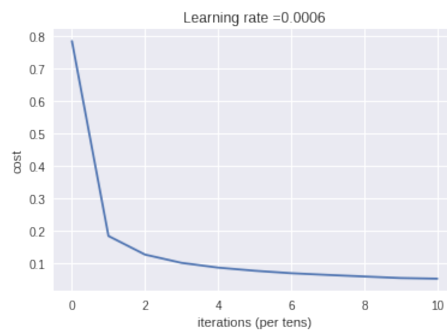

Fig. 2 Loss Curve CNN:

Learning Rate = 0.0006 ; Iterations = 10

Random Forest Model

Model	Accuracy
Logistic Regression	0.44
SVM (rbf Kernel)	0.456
MLP (50x20)	0.54
CNN	0.835

2.2 Conclusion

Maximum accuracy 83.5 was obtained for the CNN architecture (shown in fig 1.). To avoid over fitting the training data we constructed relatively simple architecture as compared to the architecture which performs the best on original mnist test set. Optimum result was concluded using early stopping.

3 Signal Data

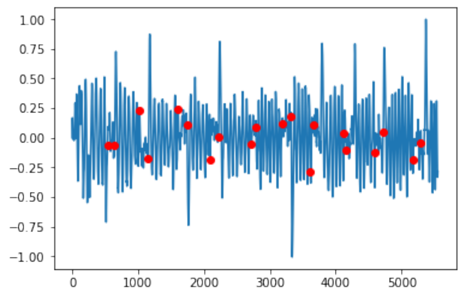

Fig. 3 Visualization of Signal and target points

The signal data problem consists of 30 signals of varying lengths with signal values given for each time-steps in each signal. The time-steps in each signal vary. Along with that, special points on the time-step scale are given for each of the signal. There are 30 given sequences(signals) with 20 special points in each signal.

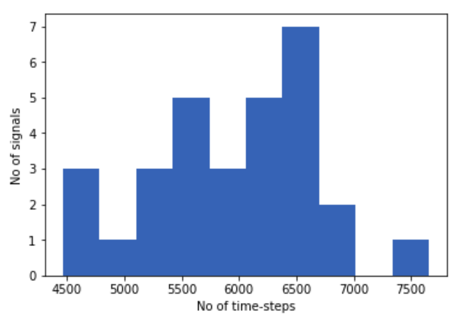

Fig. 4 No of signals vs time-steps

In the following problem, to identify special points, Recurrent Neural Networks seem to be the obvious choice as the data is sequential and has strong dependence across time. The analysis was done using Keras python module. There are two major problems encountered in the analysis of the following problem:

1) The different no of time-steps in each signal or sequence makes it hard to train the model and two methodologies are used to tackle this issue.
2) The no of time-steps in each sequence lie in range from 4500 to 7500(seen from the figure). These large sequences are hard to train in a simple RNN model.(Due to vanishing gradient issue) and takes a lot of computational power to train using LSTM. Again this is tackled using convolution nets.

3.1 Observations

3.1.1 *Training in a single batch with equal time-steps*

As a starting point of the model, all the sequences were cut down to length of 5410 time-steps and various models were tested on these.

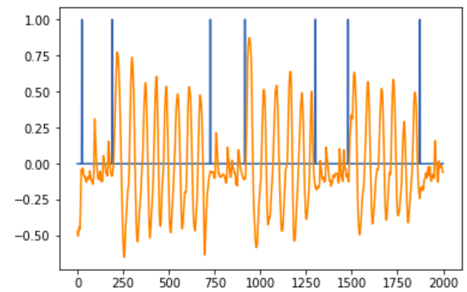

Fig.5 Signal with special points marked as vertical lines

From the figure it is clear that the special points in the signal are the starting and ending points of a sound in a speech signal.

$1st$ model: Single LSTM layer with sigmoid activation — Result - No inference

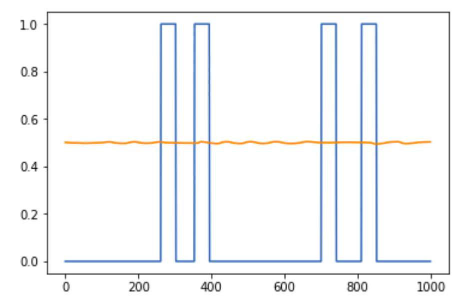

Fig.6 No inference from single LSTM layer

Similarly no inference was achieved on increasing the number of LSTM layers.

3.1.2 *Generalizing training sequence length*

Training was done on the actual sequence lengths and model was developed for general inference.
Using model.fit-generator Training of the model was done using a single sequence as a complete batch and feeding the models with no of batches = no of sequences.

$2nd$ Model: LSTM(single layer, 2units) with sigmoid activation

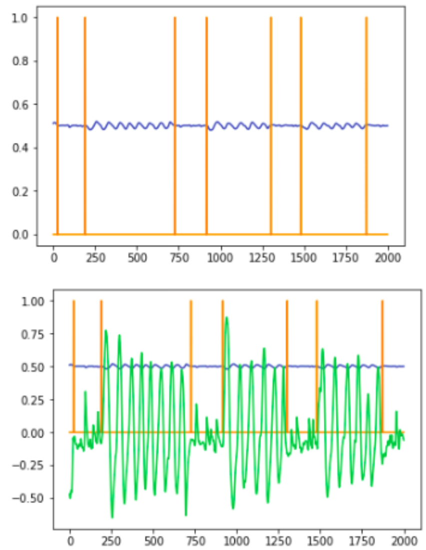

Fig. 7 Probability smooths around 0.5 between special points

Using masking

To include different lengths within a single batch, masking is used. In-band or out-of-band are signalled. In-band would be a special value that indicates that the step should be ignored. Out-of-band would be a separate input with 1 for included steps and 0 for excluded steps. Losses are multiplied by the mask, so losses for excluded steps are 0.

Results: Very similar to that by fit-generator. $3rd$ Model: 3 layers of lstm (5 units each) with sigmoid activation

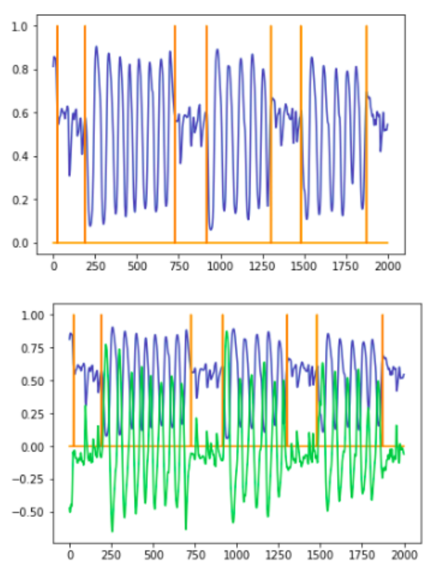

Fig. 8 Probability is shown in blue

Further increasing LSTM layers and units results in lose of gradient value as a result of large sequence lengths. This causes the model to return an image of the original signal.

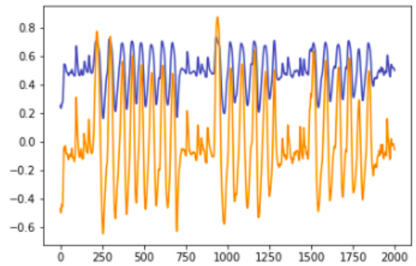

Fig. 9 Probability is shown in blue. Signal in yellow

Final model: The sequences are too large for LSTM. A layer of 1 dimensional convolution net is used followed by two layers of LSTM (5 units each) followed by sigmoid activation.

Filter(kernel) size = 15
stride = 4
no. of filters = 10
lstm units = 5

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	(None, 5410, 1)	0
conv1d_4 (Conv1D)	(None, 1349, 10)	160
batch_normalization_7 (Batch	(None, 1349, 10)	40
activation_4 (Activation)	(None, 1349, 10)	0
dropout_10 (Dropout)	(None, 1349, 10)	0
lstm_4 (LSTM)	(None, 1349, 5)	320
dropout_11 (Dropout)	(None, 1349, 5)	0
batch_normalization_8 (Batch	(None, 1349, 5)	20
dropout_12 (Dropout)	(None, 1349, 5)	0
dense_4 (Dense)	(None, 1349, 1)	6

```
Total params: 546
Trainable params: 516
Non-trainable params: 30
```

Fig. 10 Final Model summary

This model resulted in meaningful inferences.

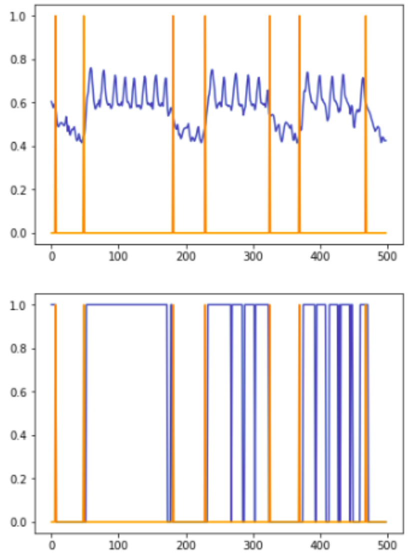

Fig. 11 Probability (blue) is lower than the threshold between special points

4 Amazon Review Helpfulness

We used the review dataset of healthcare products provided by Mcauley et al. [1]. The dataset includes reviews with their corresponding ratings, text, helpfulness votes and product metadata. Amazon has a voting system whereby community members provide helpful votes to rate the reviews of other community members. Helpfulness is available in the form of ratio of number of people who found the review helpful to the total number of people who voted. We learnt a neural network architecture which labels the review as class 1 having helpfulness > 0.5. and class 0 otherwise. Two main features used are readability index and subjectivity of the review.

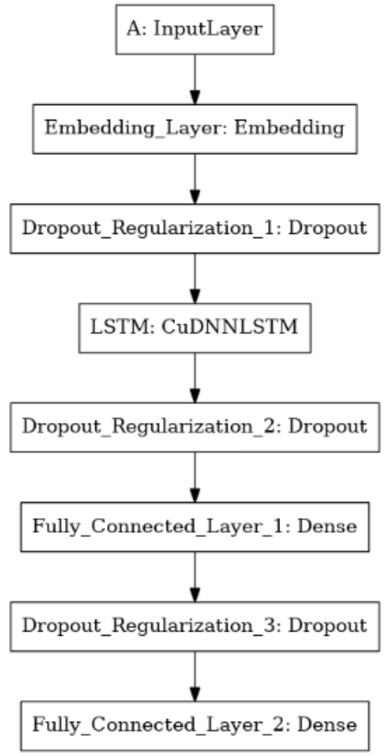

Fig 1. Neural Network Architecture

4.1 Observations

4.1.1 Data Distribution

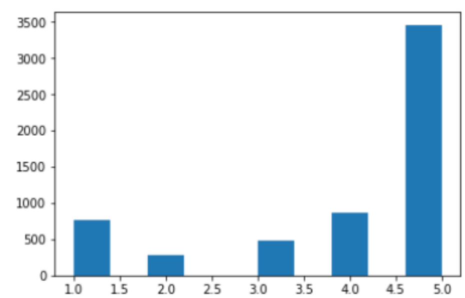

Fig 2. Ratings Histogram

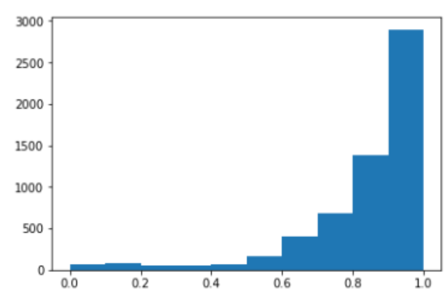

Fig 3. Helpfulness Ratio Histogram

Random Forest Model

Class	Precision	Recall	F-Score
1	0.52	0.48	0.5
0	0.45	0.36	0.4

Neural Network Model

Class	Precision	Recall	F-Score
1	0.69	0.65	0.67
0	0.587	0.632	0.608

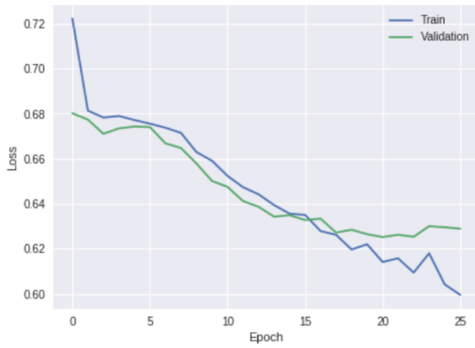

Fig 4. Neural Network Model Loss

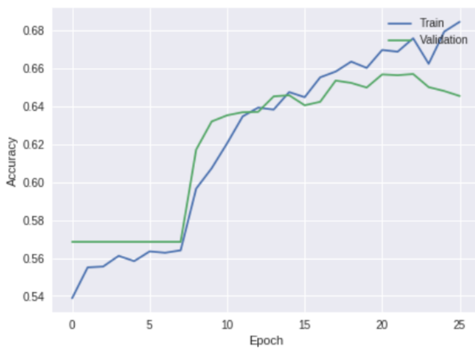

Fig 5. Neural Network Model Accuracy

4.2 Conclusion

Maximum F-score of 0.67 for class 1 is obtained using a complex neural network model architecture as described. The results are not good since we observed that the total number of unique words present were 80474 out of which 42552 were not present in the Word2Vec model due to spelling errors. These are assigned a zero vector in our model. This can be increased by either using a spell corrector or using Fasttext word embeddings provided by Facebook Research.

4.3 References

[1] Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, R. He, J. McAuley, WWW, 2016

[2] Estimating the Helpfulness and Economic Impact of Product Reviews: Mining Text and Reviewer Characteristics Anindya Ghose and Panagiotis G. Ipeirotis, IEEE, 2011

[3] http://yann.lecun.com/exdb/mnist/