CSC 215
Artificial Intelligence
Fall 2017

IMPLEMENATION PROJECT:
TRAINING OF LSTM RNN MODEL WITH HELP OF GENETIC ALGORITHMS

NAME: RAVALI GAMPA

PROFESSOR: Dr. Scott Gordon

Table of Contents

1.ABSTRACT:

The project proposes a methodology of training the recurrent neural networks with genetic algorithm. The recurrent neural networks are built using LSTM cells rather than the backpropagation technique. These LSTM cells are used for storing the error and adjusting the weights accordingly.

Genetic algorithms are used for training the recurrent neural networks. This algorithm helps in obtaining a suitable architecture for the deep networks. In this project, genetic algorithm helps to adjust the appropriate number of LSTM units and the window size of the unit.

The network built is used to test on the weather conditions of the Szeged area. The dataset is taken from Kaggle [1]. First the model was tested against a portion of the dataset. Results obtained were recorded and tabulated. Then entire dataset was experimented with the model built. Different values are tested for parameters like crossover, activation function, number of generations, generation length, population size. Even the various selection methods like roulette wheel selection and tournament selection were used. All the results obtained were recorded and tabulated.

2.INTRODUCTION:

An Artificial Neural Networks is an information processing model that is inspired by the way human biological nervous systems works such as brain and their methodology of processing the information. The main element of this model is the structure of information processing systems. Neural Networks like people learns by an example. It is configured for a specific application, such as pattern recognition or data classification, through a learning process.

Recurrent Neural Network is a class of Neural Networks that involves directed cycles in memory. This allows it to exhibit dynamic temporal behavior. They are useful with sequential data because each unit uses its internal memory to maintain information about previous input. The below diagram shows a rolled recurrent neural network [2].

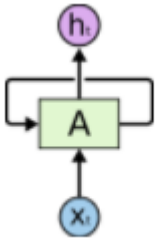

Figure1: Showing a rolled Recurrent Neural Networks with loop

The above diagram shows block of neural networks. **A** takes an input x(t) and gives h(t) as the output. Information is passed from one step to other step with the help of a loop. The below diagram [2] unrolls the above functionality.

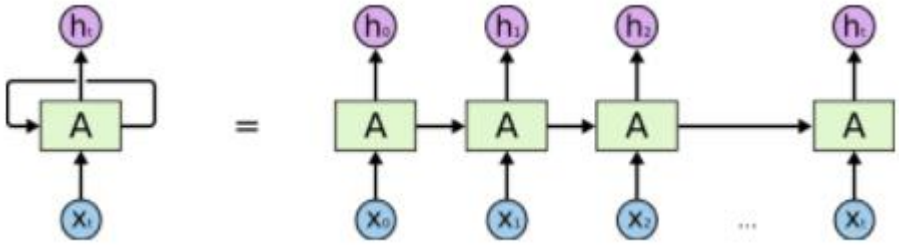

Figure 2: Showing an unrolled Recurrent Neural Networks with loops

It is understood that Recurrent networks take as their input not just the current input but also previously perceived output. The decision a recurrent net reached at time step t-1 affects the decision it will reach one moment later at time step t. Here is a diagram of a simple recurrent neural network proposed by Elman [3].

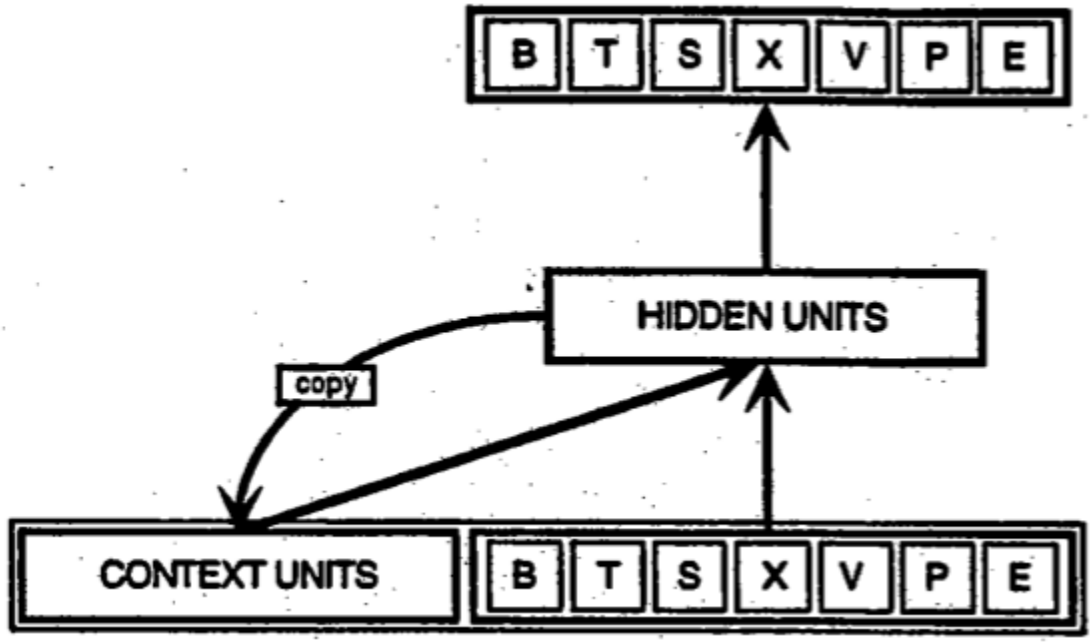

Figure 3: RNN model proposed by Elman

In the above diagram, the *BTSXPE* at the bottom of the drawing represents the input example in the current moment, and *CONTEXT UNIT* represents the output of the previous moment.

Recurrent Neural networks rely on extension of back propagation called as **"Backpropagation through time(BPTT)"** to move backward from final error through the outputs, weights and inputs of each hidden layer. In this case, time is simply expressed as series of calculations linking one-time step to the next.

Like most other neural networks, recurrent neural networks faced a major obstacle with vanishing gradient problem to its performance. The layers and time steps of deep neural networks relate to each other through multiplication, so the derivatives are susceptible to diminishing or vanishing. Figure 4 [3] depicts the effects of applying the sigmoid function again and again. There is no detectable slope and is similar to a gradient vanishing as it passes through many layers.

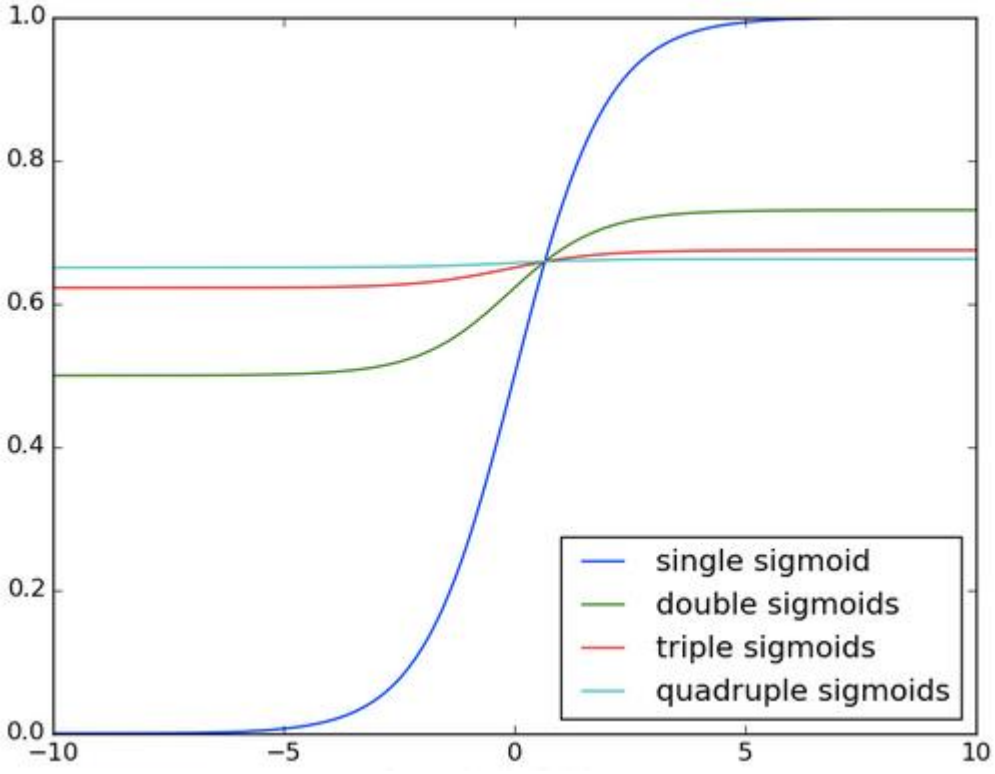

Figure 4: Showing the effect of applying sigmoid function again and again

Then came the **Long Short-Term Memory Units (LSTMs).** Adding a LSTM to the networks is like adding a memory unit that can remember context from the very beginning of the input. LSTM helps to preserve the error that can be backpropagated through time and layers.

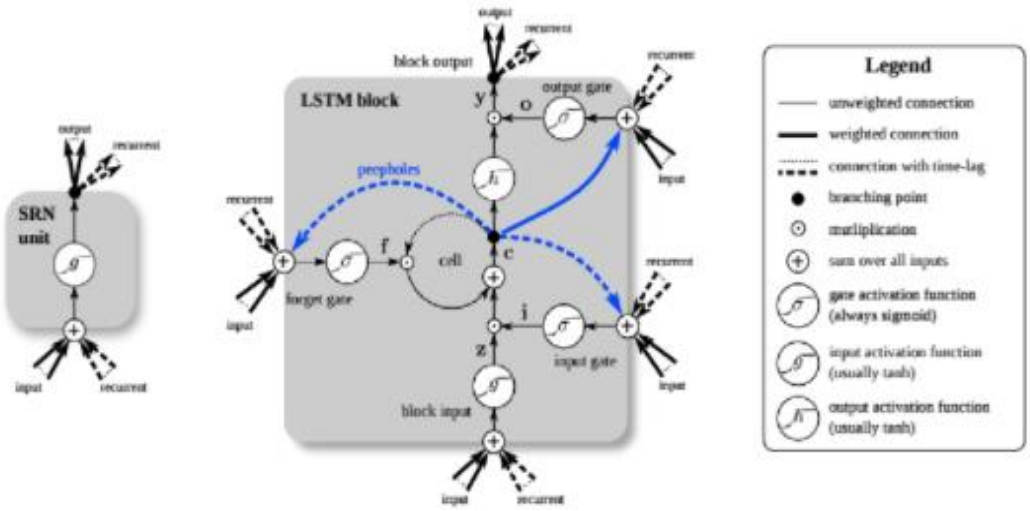

Figure 5: Schematic of simple RNN unit on left and LSTM meory block on right.

Note the central plus sign in the above diagrm [3] is the essential secret of LSTMs. Intsead of multipying its current state with the new input, it adds both of them to obtain the next subsequent cell.

3.MOTIVATION :

Recurrent neural netowrks has greatly optimsed its efficiency by using LSTM unts inside the Neural network. And LSTM have been used successfully in various applications such as Language Modelling, Image Captioning, Hand writing generation, Image generation and lot more. [4]

However like other deep neural networks, their applications to some real worls problem has been hindered by the relative time needed to train the model and obtain the desired results.

One of the expermient done by Matt Harvey [5] to apply deep learning technique to classify the CIFAR10 datset on a CPU computer took **63** hours to obtain the results. Then he began optmizing his technique with the help of Genetic algorithms to find the optimal weights. This optimization has reduced his implementation time to **7 hours.** This means he got the same results in **1/9th of time** which is a very big deal.

Due to the increase in applications of Neural networks and the success ratio of their applications there has been lot of research going on to employ Reinforcement Learning [6] and Evolutionary algorithms [7] to automatically search for optimal neural architectures which reduces the time taken to train the neural network.

4. PROBLEM STATEMENT

In this project, I have applied Genetic Algorithms to train the Long Short-term Memory based Recurrent Neural Networks. This kind off training enhances the performance of the network by

choosing a better architecture in shorter time. The model built is tested for a textual data. The data contains the parameters to obtain the weather conditions of Szeged area.

5.LITERATURE REVIEW AND EXISTING METHODOLOGY:

People have incorporated the hybrid optimization algorithm that combines the effect of both evolutionary algorithms and neural networks. Lamos-Sweeney, Joshua D [8] did a similar experiment involving genetic programming to train the deep learning. This paper covers the implementation of a multilayer deep learning network using a genetic algorithm, including tuning the genetic algorithm, as well as results of experiments involving data compression and object classification. This paper aims to show that a genetic algorithm can be used to train a non-trivial deep learning network in place of existing methodologies for network training, and that the features extracted can be used for a variety of real world computational problems.

There was even research to introduce how to incorporate the genetic algorithm to obtain the optimal seize of recurrent neural networks. M.Delgado, M.C Pegalajar [9] experimented to build a multi objective genetic algorithm to obtain the optimal size of the network for grammatical interface. This paper aims to show the reason why new training algorithms should be studied to train the network and it comes up with an approach to learn the positive and negative examples of certain language.

6.PROPOSED METHODOLOGY:

Long Short-term based Recurrent Neural Networks model has been built to understand the weather trends in Szeged, Hungary area in between 2006 and 2016 [1]. Genetic Algorithm is used to enhance the training of the RNN model.

6.1: ALGORITHM USED: Genetic Algorithm

- Genetic Algorithm have been applied to find an optimal window size and number of units in Long Short-term Memory of RNN model.
- Genetic Algorithm are adaptive heuristic search algorithms based on ideas of natural selection and genetics. They propose an intelligent search to solve an optimization problem. They do not break even though there are slight changes in inputs.
- Each individual in the population are represented as population of character strings.
- Genetic Algorithms simulate the survival of the fittest among the individuals to the next generation for solving a problem.
- Before moving on to the next consecutive generation individuals undergo crossover and mutation to preserve the diversity in the population.

6.2: TOOLS USED: Keras, DEAP

Keras API [10] helped in the execution of the Recurrent Neural Networks and DEAP framework [11] helped in incorporating Genetic Algorithms with this network.

- "DEAP (Distributed Evolutionary Algorithms in Python)" is a computation framework for rapid prototyping and testing of ideas. It works perfect with parallelization mechanism such as multiprocessing.
- One of its feature is to support genetic algorithm using representation like List, Array, Numpy Array and many more.
- "Keras" is a high level neural network API written in python to support fast and easy prototyping of deep networks.
- It supports both convolutional neural networks and recurrent neural networks.

6.3: DATA COLLECTION:

The Dataset used for this work is collected from Kaggle [1]. It is a platform for predictive modelling and analysis. This dataset has rich source about weather trends in Szeged, Hungary area in between 2006 and 2016. It includes hourly/daily summary for Szeged. Each attribute as the column attribute in the dataset. It has 12 columns and 95k rows.

	Formatted Date	Summary	Precip Type	Temperature	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)	Daily Summary
0	2012-02-28 13:00:00.000 +0100	Mostly Cloudy	rain	1.077778	-2.672222	0.60	12.9122	251	11.4471	0	1022.41	Foggy starting in the evening.
1	2008-12-29 15:00:00.000 +0100	Overcast	snow	-6.038889	-10.233333	0.92	9.1609	1	3.9123	0	1040.13	Foggy starting in the morning continuing until...
2	2013-08-08 00:00:00.000 +0200	Clear	rain	20.605556	20.605556	0.70	3.2200	50	16.1000	0	1014.80	Partly cloudy starting in the afternoon contin...

Figure 6: Sample of the collected dataset

6.4: INSIGHTS ABOUT THE DATASET:

I tried experimenting the dataset with the help of Jupyter Notebook [12] and the followings insights were obtained.

1. Complete details of the dataset were reflected at a single glance. Figure 7 [12] depicts all the details of the dataset.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1699 entries, 0 to 1698
Data columns (total 13 columns):
Formatted Date                1699 non-null object
Summary                       1699 non-null object
Precip Type                   1699 non-null object
Temperature                   1699 non-null float64
Apparent Temperature (C)      1699 non-null float64
Humidity                      1699 non-null float64
Wind Speed (km/h)             1699 non-null float64
Wind Bearing (degrees)        1699 non-null int64
Visibility (km)               1699 non-null float64
Loud Cover                    1699 non-null int64
Pressure (millibars)          1699 non-null float64
Daily Summary                 1699 non-null object
0.315298306                   1699 non-null float64
dtypes: float64(7), int64(2), object(4)
memory usage: 172.6+ KB
```

Figure 7: Depicting the details about the dataset.

2. It has 4 categorial data items and 8 quantitative data items. The range about each entity in both categories is visualized. Details like count, minimum value, maximum value, mean value etc. of each attribute in quantitative section is understood. Figure 8 [12] talks about these details. Count factor in these details help us in concluding that there are no NULL values in the dataset. Figure 9[12] talks about the insights of the summary attribute.

```
]: df[quantitative].describe()
```

]:

	Temperature	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover	Pressure (millibars)
count	1699.000000	1699.000000	1699.000000	1699.000000	1699.000000	1699.000000	1699.0	1699.000000
mean	12.107671	11.104002	0.741701	10.523269	186.539729	10.397899	0.0	1004.517587
std	9.424189	10.503965	0.194283	6.809845	106.144672	4.198100	0.0	109.937223
min	-19.444444	-24.638889	0.000000	0.000000	0.000000	0.000000	0.0	0.000000
25%	4.905556	2.705556	0.625000	5.248600	119.000000	8.267350	0.0	1011.450000
50%	12.172222	12.172222	0.790000	9.708300	174.000000	10.046400	0.0	1016.170000
75%	18.788889	18.788889	0.900000	13.918450	290.000000	14.908600	0.0	1020.560000
max	37.938889	37.188889	1.000000	41.393100	359.000000	16.100000	0.0	1041.630000

Figure 8: Insights about quantitative attributes.

3. Visualization of categorial data.
   The Dataset is visualized as a horizontal histogram for a better understanding of the data. Figure 9 [12] talks about the insights of the summary attribute.

```
rcParams['figure.figsize'] = 8, 5
sns.countplot(y=df['Summary'])
plt.show()
```

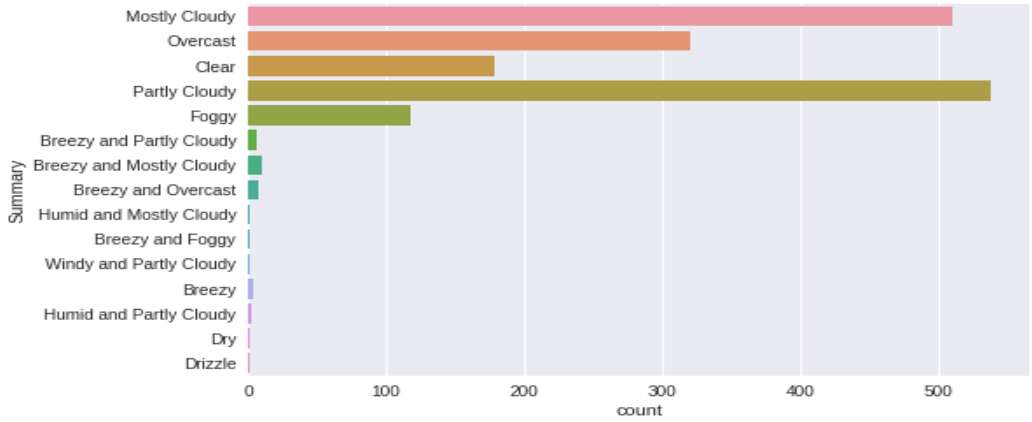

Figure 9: Insights about Summary attribute

4. Visualization of the quantitative attributes

   The Dataset is visualized as set of histograms for a better understanding of the data. A total of eight attributes was displayed. The below diagram depicts that.

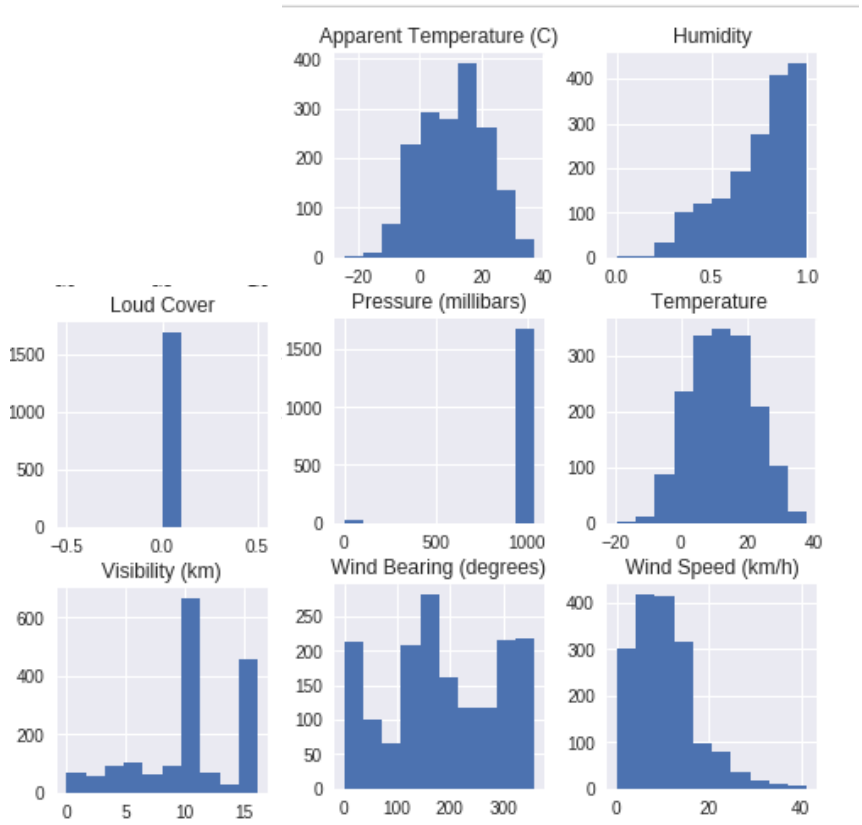

Figure 10: Histograms depicting the trends in quantitative attributes

6.5: PARAMETRS SELECTION

In the categorical attributes Summary, Daily summary and the Precipitation type are in turn dependent on the quantitative attributes. Considering only the quantitative attributes will solve the purpose of understanding the weather trends. And as the dataset is collected hourly/daily basis, the Date attribute is not an important factor.

From the visualization it is evident that Loud Cover, Pressure and Visibility effect the least. But Temperature, Apparent Temperature, Humidity, Wind Bearing, Wind Speed are the dominant factors.

So, I have chosen Humidity and Apparent temperature to shape the model.

7.IMPLEMENATION DETAILS

7.1 Using the Dataset

- Initially when complete dataset was used to start experimenting. It was taking a longer time. So, I thought why not I narrow the dataset and compute the results in shorter time. With this idea I narrowed the dataset from 95k rows to 1700 rows.
- Then rand() function was added as the last column and shuffled the data entries with respect to this column and then top 1700 rows were been selected.
- This shuffling helped to preserve the diversity of variations in the dataset.

Formatted	Summary	Precip Typ	Temperat	Apparent	Humidity	Wind Spe	Wind Bea	Visibility (	Loud Cove	Pressure (	Daily Sum	rand=()
2012-02-2	Mostly Clo	rain	1.077778	-2.67222	0.6	12.9122	251	11.4471	0	1022.41	Foggy star	0.842389
2008-12-2	Overcast	snow	-6.03889	-10.2333	0.92	9.1609	1	3.9123	0	1040.13	Foggy star	0.065563
2013-08-0	Clear	rain	20.60556	20.60556	0.7	3.22	50	16.1	0	1014.8	Partly clou	0.429549
2006-09-0	Partly Clo	rain	20.33333	20.33333	0.71	8.3559	117	10.3523	0	1017.19	Partly clou	0.129459
2006-01-1	Overcast	snow	-4.38889	-6.87778	0.88	5.6511	160	7.9051	0	1028.22	Overcast t	0.0651
2016-11-1	Mostly Clo	rain	6.766667	4.283333	0.8	12.7834	159	9.1931	0	1019.66	Partly clou	0.239026
2007-04-2	Clear	rain	14.59444	14.59444	0.37	14.1519	33	15.3111	0	1025.43	Partly clou	0.073061
2012-07-2	Overcast	rain	16.63889	16.63889	0.9	10.4811	329	14.3451	0	1017.18	Mostly clo	0.41186
2015-04-0	Partly Clo	rain	7.2	5.438889	0.49	9.3219	331	16.1	0	1029.33	Partly clou	0.320697
2015-08-3	Partly Clo	rain	35	34.71111	0.3	9.4346	170	16.1	0	1020.49	Partly clou	0.290901
2009-11-1	Overcast	rain	11.14444	11.14444	0.93	9.7727	76	3.8318	0	1004.01	Foggy star	0.180444
2013-11-2	Mostly Clo	rain	12.47222	12.47222	0.74	12.5419	88	7.8407	0	1003	Foggy star	0.085132
2007-08-0	Mostly Clo	rain	30.82778	29.86667	0.33	7.9695	326	11.2056	0	1008.74	Partly clou	0.106787
2014-09-2	Clear	rain	7.855556	5.961111	0.89	10.6743	282	15.1823	0	1019.98	Partly clou	0.141044
2007-02-2	Partly Clo	rain	0.788889	0.788889	0.88	3.1395	318	11.9784	0	1014.26	Mostly clo	0.796934
2012-02-0	Partly Clo	snow	-7.22222	-10.5222	0.67	6.44	100	8.05	0	1029.1	Partly clou	0.72202
2016-06-0	Mostly Clo	rain	15.02222	15.02222	0.99	5.7799	296	9.982	0	1009.51	Mostly clo	0.348298
2016-11-2	Mostly Clo	rain	4.127778	1.183333	0.85	12.1716	160	5.957	0	1019.66	Mostly clo	0.825297
2013-09-1	Mostly Clo	rain	19.75556	19.75556	0.47	4.025	27	10.5777	0	1014.29	Mostly clo	0.556127
2014-07-2	Clear	rain	27.2	27.9	0.54	0.1449	80	16.1	0	1015.8	Partly clou	0.470705

Figure 11: Dataset depicting the rand() with other attributes

- Then as the results were not good enough again the original dataset has been considered. In that 90000 rows are considered for training the model and rest 6k rows

are considered for testing the model. 20% of training dataset is used as a validation dataset.

7.2 Working with Genetic Algorithm

- Each individual in the population is represented as strings of binary integers. The length of each individual is 10 bits long.
- Fitness is the criterion used to decide if any individual moves to the next generation or not. The fittest among the individuals is moved to the next generation for solving a problem. In this case Root-Mean-Square Error (RMSE) on validation set acts as the fitness function.
- Multiple options are being tested for selection and crossover with the help of pre-defined functions in the DEAP package. [13]
- Roulette wheel and Tournament selection are experimented for selecting the off springs.
- Mutation is applied through the concept called Elitism.
- One point, two-point crossover and Ordered crossover are tried with the population.
- Then values in different range is chosen to find the best set value set (Population size, Number of generations and Generation length) for building the model.

7.3 Incorporating Genetic Algorithm with LSTM

- The first 6 bits of each individual is represented as window size and bits after that to end are represented by number of units in LSTM model
- The genetic algorithm helps to obtain the best possible value to build the LSTM model.

8.RESULTS

8.1 Considering a smaller Dataset

- The Results were not good enough considering a small dataset.
- Population Size, Generation length and Number of generations are tested for their higher and lower values respectively.
- Two-point crossover, Roulette wheel selection and Linear activation function is used.
- The below tabular column shows the set of results.
- It shows the range of values for Train RMSE and Loss.
- It even shows the test RMSE on testing the model with the test cases.

Table 1: Showing the results for a smaller dataset.

	POPULATION SIZE		NUMBER OF GENERATIONS		GENERATION LENTH	
	Higher value (400)	**Lower value (4)**	**Higher value (1000)**	**Lower value (10)**	**Higher value (500)**	**Lower value (5)**
Train RMSE	11-23	12-25	5-15	10-14	10-20	9-20
Test RMSE	11.14	7.89	7.14	12	10.379	9
Loss	120-230	120-240	140-280	130-287	135-302	120-280

8.2 Considering complete Dataset

a. Roulette Wheel selection

- The Results were much better than considering a small dataset.
- Population Size, Generation length and Number of generations are tested for their higher and lower values respectively.
- The below tabular column shows the set of results.
- Two-point crossover and Linear activation function is used.
- It shows the range of values for Train RMSE and Loss.
- It even shows the test RMSE on testing the model with the test cases.

Table 2: Showing the results for a complete dataset using Roulette wheel selection

	POPULATION SIZE		NUMBER OF GENERATIONS		GENERATION LENTH	
	Higher value (500)	**Lower value (10)**	**Higher value (1000)**	**Lower value (10)**	**Moderate value (25)**	**Lower value (7)**
Train RMSE	1.5-1.75	1.7-1.85	1.3-1.9	1.7-1.85	1.4-1.7	1.7-1.85
Test RMSE	1.55	1.59	1.44	1.59	1.4	1.59
Loss	10-20	15-30	12-20	15-30	15-30	15-30

b. Tournament Selection

- The Results were much better than considering a small dataset.
- Population Size, Generation length and Number of generations are tested for their higher and lower values respectively.
- Two-point crossover and Linear activation function is used.
- The below tabular column shows the set of results.

Table 3: Showing the results for a complete dataset using Tournament selection

	POPULATION SIZE		NUMBER OF GENERATIONS		GENERATION LENTH	
	Higher value (500)	**Lower value (10)**	**Higher value (1000)**	**Lower value (10)**	**Moderate value (25)**	**Lower value (7)**
Train RMSE	1.2-1.75	1.3-1.75	1.3-1.9	1.3-1.75	1.3-1.8	1.3-1.75
Test RMSE	1.35	1.45	1.4	1.45	1.3	1.45
Loss	10-20	12-28	10-20	12-28	15-30	12-28

c. **Relu Activation function**

Linear Activation function is used in the above two cases. Now model is experimented with 'Relu' activation function.

- The Results were much better than considering a small dataset.
- Population Size, Generation length and Number of generations are tested for their higher and lower values respectively.
- The below tabular column shows the set of results.

Table 4: Showing the results for a complete dataset using Relu activation function.

	POPULATION SIZE		NUMBER OF GENERATIONS		GENERATION LENTH	
	Higher value (500)	**Lower value (10)**	**Higher value (1000)**	**Lower value (10)**	**Moderate value (25)**	**Lower value (7)**
Train RMSE	1.2-1.7	1.3-1.9	1.3-1.7	1.3-1.9	1.2-1.8	1.3-1.9
Test RMSE	1.30	1.4	1.3	1.4	1.34	1.4
Loss	10-20	15-25	12-20	15-25	15-30	15-25

All the above cases where even experimented with different crossover techniques like one point and ordered crossover. But there was not a huge difference in the results obtained. The values differed in the second or third decimal point.

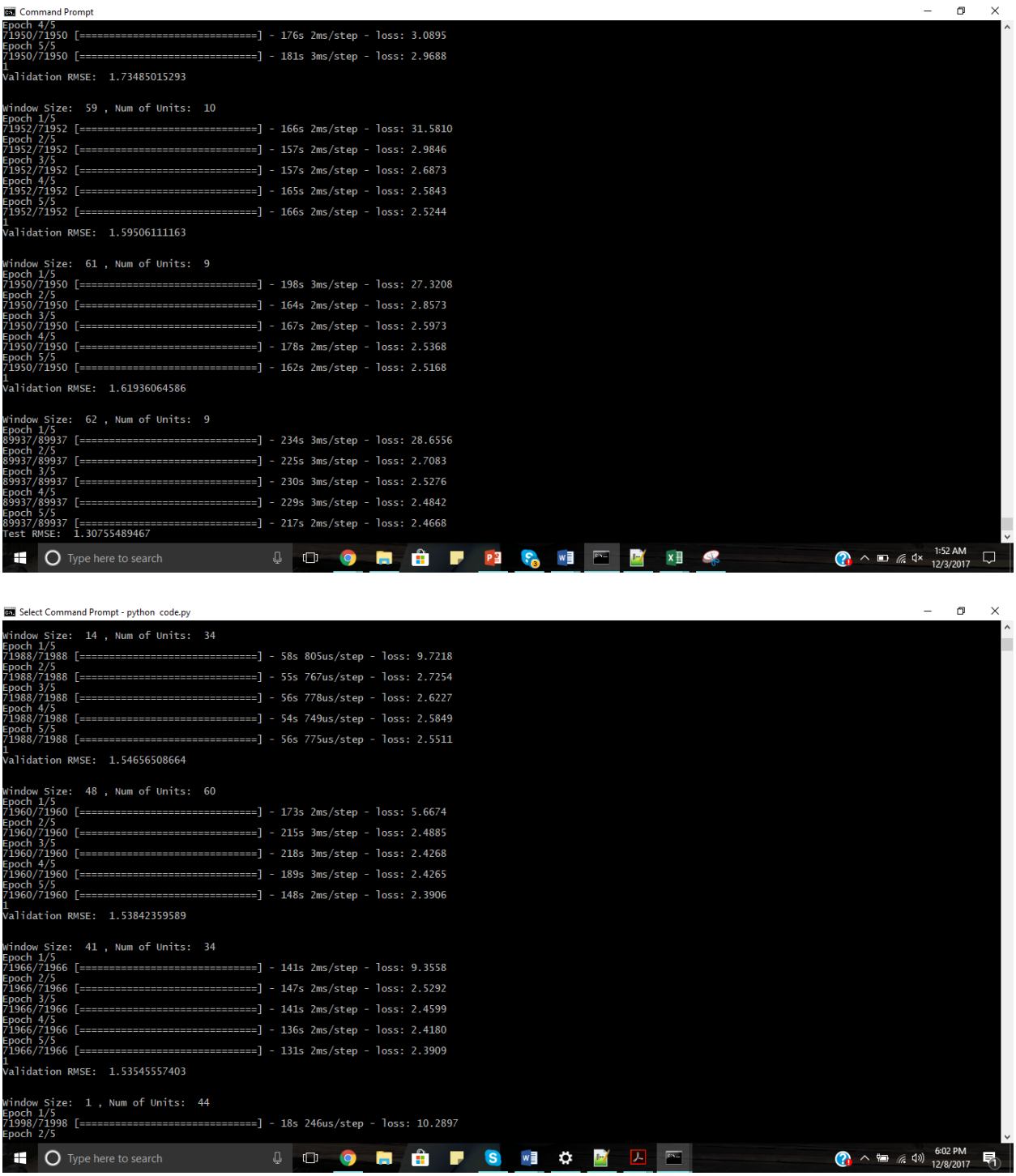

Figure 12: Test results for Relu Activation function and Tournement selection

9.ANALYSIS OF RESULTS

- Initially when the model was built using a partial dataset the results obtained were very bad. All three Train RMSE, Test RMSE and the loss have high values. The results did not improve on changing the values for number of generations, generation length and population size. But

when the complete dataset was used the performance of the model enhanced in a considerable way.

So, it can be concluded that these LSTM based RNN networks works best for the larger data.

- Then for the larger dataset, primarily roulette wheel selection was used as a selection technique to select the off-springs for the next generation. Then the results were much better than the smaller dataset. The Train RMSE, Test RMSE came down from a double-digit value to a single digit value. Loss decreased almost to one-tenth of its value.
- Then when the model was experimented with "Tournament Selection" there is a betterment in the results. Though there is not a great change in the loss value. There is an improvement in the Train and Test RMSE values.
- Then when the activation function was changed from 'Linear' to 'Relu', there is a slight improvement in the results obtained.
- In each case having larger generations, higher population size, moderate generation length showed better results.
- First epoch in every generation had a greater loss compared to other epoch's value. This probably means, it is when model is learning and training the data for the first time it is having little difficulty to understand the trends in data unlike 2^{nd} , 3^{rd} .. time it would have understood the trends little.
- The results were best working with higher values of population size, more number of generations, moderate generation length and Tournament selection used with Relu activation function.

10.FUTURE SCOPE AND CONCLUSION

10.1 CONCLUSION

The proposed LSTM based RNN model worked best for the larger dataset. This means that model can be well trained and tested with larger data.

Among the two-selection methods Tournament Selection was better than compared to Roulette Wheel selection for the chosen dataset.

There was not a significant difference in the performance of the model with change in type of crossover technique being used.

Model with 'Relu' Activation function performed slightly better than 'linear' Activation function.

Model predicted well for higher population size, higher number of generations and a moderate generation length than lower values.

10.2 FUTURE SCOPE

The project proposed the a RNN-LSTM model to predict the weather conditions of Szeged area. The training of this RNN-LSTM model was experimented through genetic algorithm. Two of the four dominant attributes were considered at once during building the model. I look forward to build a model considering all the four dominant features at once.

Later on, this can be compared with the traditional LSTM RNN model to understand how well did the genetic algorithms help in the training process.

The project was all concerned about the textual data. This concept of training an RNN-LSTM model with help of genetic algorithms can be applied to the image dataset as well to see how it performs on a 3D data.

In addition to genetic algorithms there are other evolutionary algorithms like PSO that can be applied to train the network to obtain the better architecture and check for possible results.

11. REFERENCES

[1] NorbertBudincsevity, "Weather in Szeged 2006-2016", gathered and deposited in Kaggle about a year ago.

[2] Magenta webpage: https://magenta.tensorflow.org/2016/06/10/recurrent-neural-network-generation-tutorial "A Recurrent Neural Networks Music Generation Tutorial" (last accessed on 8th December).

[3] DeepLearning4j webpage: https://deeplearning4j.org/lstm.html "A Beginner's Guide to Recurrent Networks and LSTMs", (last accessed on 8th December).

[4] Quora webpage : https://www.quora.com/What-are-the-various-applications-where-LSTM-networks-have-been-successfully-used "What are the various applications where LSTM networks have been successfully used? "(last accessed on 8th December).

[5] Blog coast webpage: https://blog.coast.ai/lets-evolve-a-neural-network-with-a-genetic-algorithm-code-included-8809bece164 " by Matt Harvey. "Let's evolve a neural network with genetic algorithm-code included" (last accessed on 8th December).

[6] Barret Zoph, Quoc V. Le in "Neural Architecture Search with Reinforcement Learning" a conference paper submitted at ICLR in 2017.

[7] Kenneth O. Stanley, Risto Mikkulainen in "Evolving Neural Networks through Augmenting Topologies "a MIT press Journal.

[8] Joshua D. Lamos-Sweeney in "Deep Learning Using Genetic Algorithms" a Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Computer Science at B.

Thomas Golisano College of Computing and Information Sciences Rochester Institute of Technology.

[9] M. Delgado, M.C Pegalajar in " A multipurpose genetic algorithm for obtaining the optimal size of recurrent neural network for grammatical interference" in September 2015.

[10] Keras webpage: https://keras.io/ "Keras: The python Deep learning library" (last accessed on 8th December).

[11] Github webpage: https://github.com/DEAP/deap "Distributed Evolutionary Algorithms in Python"(last accessed on 8th December).

[12] Kaggle kernels: https://www.kaggle.com/saraseguraquerol/exploratory-analysis-and-weather-calendar "Exploratory Analysis and Weather Calendar" (last accessed on 8th December).

[13] DEAP webpage : http://deap.readthedocs.io/en/master/api/tools.html "Evolutionary tools" (last accessed on 8th December).