Earthquake Forecasting Using Recurrent Neuron Networks

Susan Jiao
jjiao25@wisc.edu

Lynette Gao
lynette.gao@wisc.edu

Han Cao
hcao29@wisc.edu

William Han
whan29@wisc.edu

Abstract

Earthquake forecasting based on time-series analysis of seismic wave is one of the most challenging problems for a variety of learning methodologies. In this project, we utilize recurrent neural network (RNN) to predict the time until the next earthquake using laboratory data from the los Alamos National Laboratory. We evaluate and compare a number of variants of RNN based on LSTM and GRU. Both bidirectional and unidirectional architectures were employed to make forecasts. Unidirectional LSTM demonstrated the highest forecasting performance.

1. Introduction

An earthquake is a destructive natural disaster.[5]. It is the result of a sudden release of stored energy in the Earth's crust that creates seismic waves. Accurate earthquake prediction can save many lives and avoid economic losses. However, no major earthquake has been successfully predicted yet. Earth's crust response to changing seismic waves is not linear and is dependent on the crust's complex and highly variable geology. This complexity makes it difficult for researchers to create a replicable model for earthquakes. Scientists can only calculate the probability that a significant earthquake will occur in a specific area within a certain period. To put in the terms from the U.S. Geological Survey (USGS), earthquakes cannot be predicted but can be forecast.

There have been many attempts in the past to forecast when and where an earthquake would take place and its magnitude. Most have come up fruitless, and the few successful predictions have been fraught with controversy. If a successful model were to be discovered, millions of lives could be saved and governments could better prepare for its devastating effects. For example, since the beginning of 2000, there have been 32 major earthquakes, with each causing at least 100 deaths. The 2010 Haiti earthquake caused an estimated 310,000 deaths and seriously stunted the economy and overall well-being of the country.

In this project, we utilize recurrent neural network (RNN) to make earthquake forecasts. We forecast when an earthquake occurs based on data from the Los Alamos National Laboratory. Laboratory experiments provide meaningful insights into the physical and mathematical properties of seismic waves. While previous studies use recurrent neural network to predict the magnitude of the earthquake, we use RNN to predict the time remaining before laboratory earthquakes occur from real-time seismic data by building a robust model that compares results from Long Short-Term Memory and Gated Recurrent Network.

2. Related Work

There is some existing literature building various models to make earthquake forecasts. Previous studies can be divided into two categories: studies that focus on building mathematical models and studies that utilize artificial intelligence.

Researchers from USGS provided a comprehensive overview of models that are time-independent and time-dependent [6]. Time-independent models assume that the probability of the occurrence of an earthquake follows a Poisson distribution, whereas time-dependent models are based on log-normal and Brownian passage time assumptions.

Models that utilize artificial intelligence can be divided into unsupervised learning frameworks and supervised learning frameworks. In the unsupervised learning framework, clustering techniques and association rules have been applied. Mirrashid (2014) adopted a neuro-fuzzy inference system based on a C-means algorithm to predict earthquakes in Iran[5].

In recent years, regression and classification in supervised learning framework are the most widely used methods to forecast earthquakes. Our project lies in this category. Adeli (2009) proposed a probabilistic neural network to predict the magnitude of earthquakes in a predefined future period in a seismic region using eight mathematically computed seismicity indicators[1]. The model yields good prediction accuracy for earthquakes of magnitude between 4.5 and 6.0. Adeli along with Panakkat (2009) also developed a recurrent neural network model with good prediction accuracy for earthquakes with magnitude greater than 6.0[1]. In our project, we predict the time until the next earthquake instead of magnitude.

3. Proposed Method

Recurrent Neural Network (RNN) is the common deep learning model used for time series analysis. The seismic wave fits well in this model cotext. However, RNN is difficult to train due to the exploding and vanishing gradient problem. Long Short-Term Memory (LSTM) and Gated Recurrent Network (GRU) are two different improved architectures that are built from RNN. In this project, we use seismic waves to make predictions by building some robust models so that we can compare results between two types of RNN. We are also interested in seeing which model performs better for our data set.

LSTM is the most widely used way of dealing with the vanishing gradient problem by reparametrize the RNN. It consists of three gates: input, output, and forget. The gates facilitate the cell to remember values for an arbitrary amount of time and control the flow of information in and out the LSTM cell. However, the complex structure makes researchers question whether all components with it are all necessary. Jozefowicz et al.(2015)showed that the input gate and forget gate are important, whereas the output gate is relatively unimportant[4]. On the other hand, the gated recurrent unit (GRU) proposed by Cho el al.(2014) uses less training parameters by combining the forget gate and input gate[3].The way in which GRU controls the flow of information closely resembles the LSTM, but the GRU does not use memory cells. It exposes the full hidden content without any control. As a result, less memory is used. Thus, the GRU is computationally more efficient, whereas the LSTM gives higher accuracy on datasets that use longer sequences. Althelaya et al.(2018) conducted a similar time series analysis on stock market forecasting by using bidirectional and stacked LSTM and GRU[2]. They found that stacked LSTM produced the highest performance on both short- and long-term forecasting.

The seismic wave sequence in our data set is quite large, so utilizing the GRU could help with higher training efficiency. On the other hand, the prediction accuracy of the time till earthquake occurrence is our main object of interest, so we expect LSTM to give overall better performance than the GRU in terms of results.

In addition, we also introduced the idea of bidirectional RNNs to try to further overcome the limitations of the original regular RNN. Schuster(1997) proposed "a bidirectional recurrent neural network (BRNN) that can be trained using all available input information in the past and future of a specific time frame"[7]. By splitting the state neurons of a regular RNN into two parts - the positive time direction (forward states) and negative time direction (backward states), bidirectional network is able to step through the input sequence in both directions at the same time. Comparing to unidirectional LSTM which can only learn from the information of the past, bidirectional LSTM can use the two hidden states combined at any point to preserve information from both past and future. In our case, we expect the implementation of the BRNN will improve our model because the context of the whole seismic wave data is used to interpret the earthquake's pattern rather than a linear interpretation.

4. Experiments

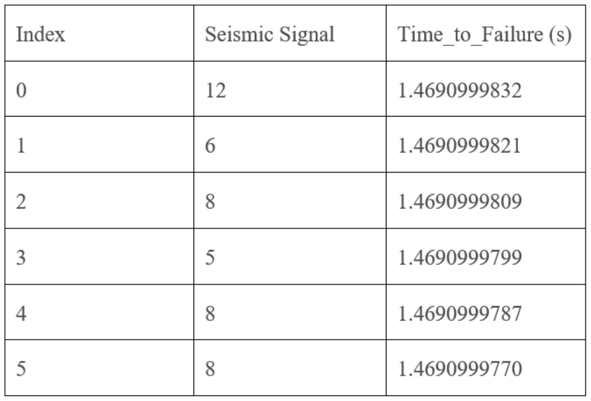
4.1. Data Overview

The total size of our train data is 9GB with 629 million rows. The training data is a single, continuous segment of experimental data consisting of two variables: seismic signal and time to failure. The seismic signal is the measure of seismic waves caused by the sudden breaking of rock within the earth. A specific way of measure is not given, so the unit of the seismic signal is unknown as to whether it is in terms of wavelength, amplitude, or frequency. However, the unit is not as important in this analysis, as our object of interest is the relationship between seismic signal and time till the next laboratory earthquake occurrence as represented by the variable time to failure. Time is measured in seconds. Table 1 gives an example overall of the training dataset. The difference between two adjacent rows is in terms of nanoseconds (10-9s).

Figure 1 shows a snapshot of how trends of seismic signal and time to failure overlap within one occurrence of a laboratory earthquake. The green line denotes the time to failure where it slowly decreases until reaching the point when an earthquake takes place. Then time to failure resets after an earthquake occurs, as shown by the peak, and such pattern repeats. The seismic signal, our predictor, is represented by the blue line. It shows a tight cyclical trend around zero until a heavy peak of the signal occurs. The peak is followed by a smaller scale fluctuation, and the earthquake occurs shortly after. This pattern stably repeats itself for each earthquake occurrence as shown in a longer lagged series in figure 2.

4.2. Feature Selection

We attempted two methods for our feature extraction. We first tried to manually select features based on commonly used descriptive statistics. Since the raw training set has a huge number of time slices of 629 million, we split the data set into a smaller number of time slides of 150,000 which is the number of time slides in the test data sets provided by Kaggle. Then, we manually generate smaller chunks by dividing the original sequence into 1,000 chunks where each consists of 150-time slices. Within each chunk, we extract 11 descriptive statistics: minimum, maximum, standard deviation, average, sum, median, the difference with mean, and quartiles.



Index	Seismic Signal	Time_to_Failure (s)
0	12	1.4690999832
1	6	1.4690999821
2	8	1.4690999809
3	5	1.4690999799
4	8	1.4690999787
5	8	1.4690999770

Table 1. data overview

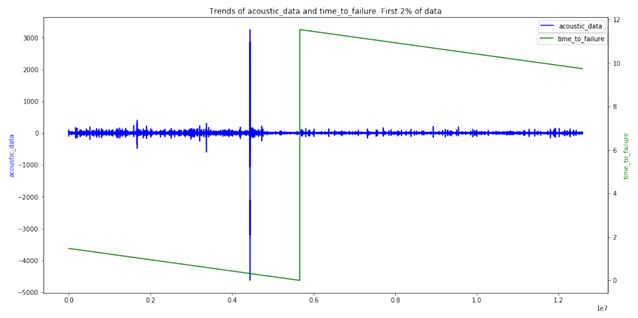


Figure 1. trends of seismic signal and time to failure within one occurrence of laboratory earthquake

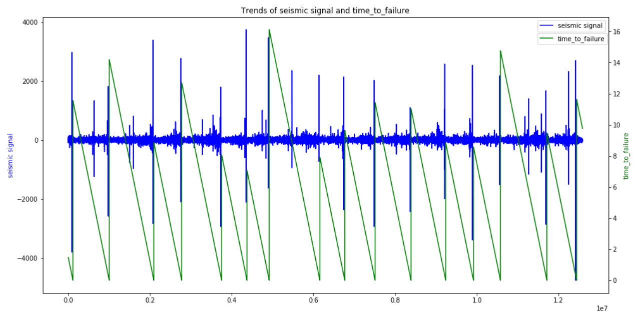


Figure 2. trends of seismic signal and time to failure within multiple occurrence of laboratory earthquake

We also considered using a one-dimensional convolution neural network (CNN) to automatically extract features for us. We chose to use three layers where each has a stride of 10 to reduce the dimension of the input data from 150,000 to 15,000. After the three layers, we now have input data with a sequence length of 150 with 16 features selected by CNN. CNN could potentially remove less important inputs. However, we encountered some convergence problems, and we will complete this part as future work.

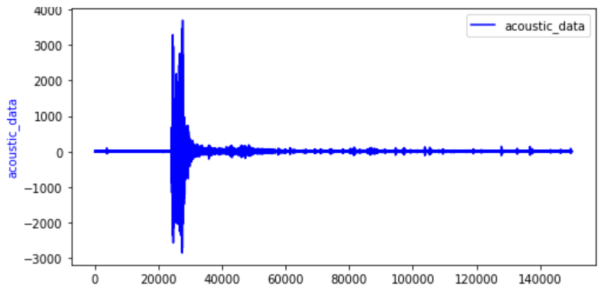


Figure 3. test segment 1

4.3. Performance Evaluation

We use mean absolute error (MAE) as our way of evaluation against a naive benchmark linear regression model. The linear regression model gives an MAE score of 2.804 for the training data and an MAE score of 3.1054 for the validation set: We expect both of our proposed methods to achieve a lower mean absolute error. We also compare the MAE between the LSTM and the GRU. The mathematical equations of the performance measures are defined as follows:

$$MAE = 1/n \sum_{i=1}^{n} |y_i - \hat{y}_i|.$$

The test data set provided by the Los Alamos National Laboratory consists of many small segments of experimental data instead of a single segment as in the training set. The data within each segment is continuous, but the test set does not represent a continuous segment of the experiment. The predictions cannot be assumed to follow the same regular pattern seen in the training data, as more potential volatility exists in the smaller segments. Figure 3, 4, 5 and 6 show three example segments within the test data. We can see that test segment in figure 3 follow the general seismic signal pattern in the training, whereas test segment in figure 5 and 6 shows a much more volatile pattern with several peaks and more frequent cyclical fluctuation around mean zero. The test segment in figure 4 shows a similar pattern with the general seismic signal, but there are two peaks instead of one. A small peak occurred before the second large peak where the earthquake actually occurred. The first small peak could be a false signal. Also, we tell see that the test set contains more noise than the training set.

Since our data is from Kaggle, Kaggle conveniently computes the MAE of our models for the test data. The private leader board is calculated with approximately 87 percent of the test data., while the public leader board is calculated with approximately 13 percent of the test data. MAE score in the private leader board provides a more accurate evaluation of our models.

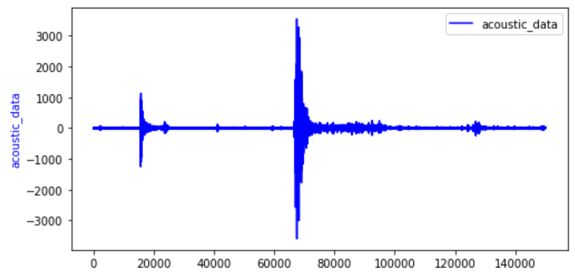


Figure 4. test segment 2

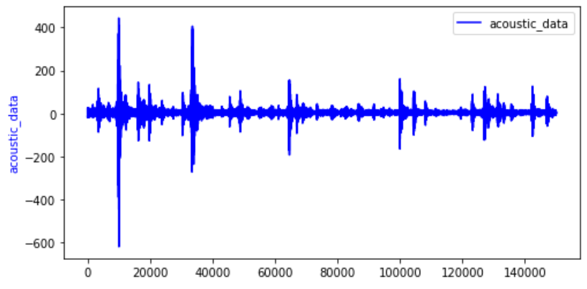


Figure 5. test segment 3

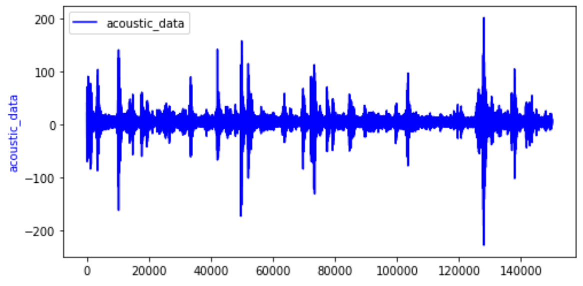


Figure 6. test segment 4

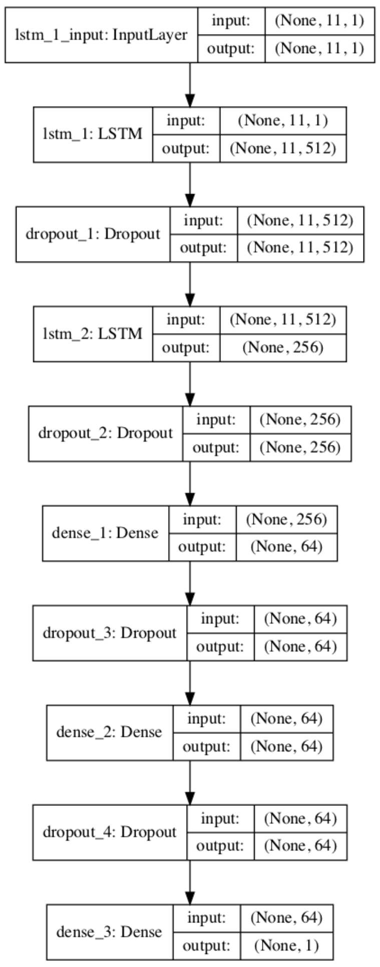


Figure 7. LSTM implementation

4.4. Prediction Model and Parameter Tuning

As discussed before, we decide to use the LSTM and the GRU as our two main models for analysis. We used Keras to implement both, and the weights are initialized to be uniformly distributed. In the first LSTM model, we use bidirectional LSTM with 512 units as our input layer, and we add a second LSTM layer with 256 units. The activation function we use is Relu after each LSTM layer. We have compared the results among tanh, sigmoid, and relu, and relu is the best performed one. In addition, we add a dropout with probability of 0.2 after each layer to reduce over-fitting. Unidirectional LSTM is built in the same way with the bidirectional layer replaced by unidirectional one. Figure 7 shows the detailed implementation.

The GRU model resembles a similar structure with LSTM. We also use two layers of GRU where the first layer is bidirectional with 512 units, and the second layer is one-directional with 256 units. Relu is used as the activation function. Different from the LSTM model, we add batch normalization to normalize hidden layer input and to increase training stability and convergence rate. We also implemented unidirectional GRU using the same structure. Detailed implementation of it is shown in figure 8.

4.5. Software

For tuning purposes, we use Google Colab(colab.research.google.com/) to adjust the code. Google Colab provides a Nvidia K40 GPU. We also activate Google Colab Pro to use better GPUs for training. Google Colab Pro gives us access to Nvidia V100 or TPU which are all very powerful GPUs. Due to the highly interactive features of Google Colab, it allows us to fix bugs and tune our model more efficiently.

- Python:

-DL: Kera

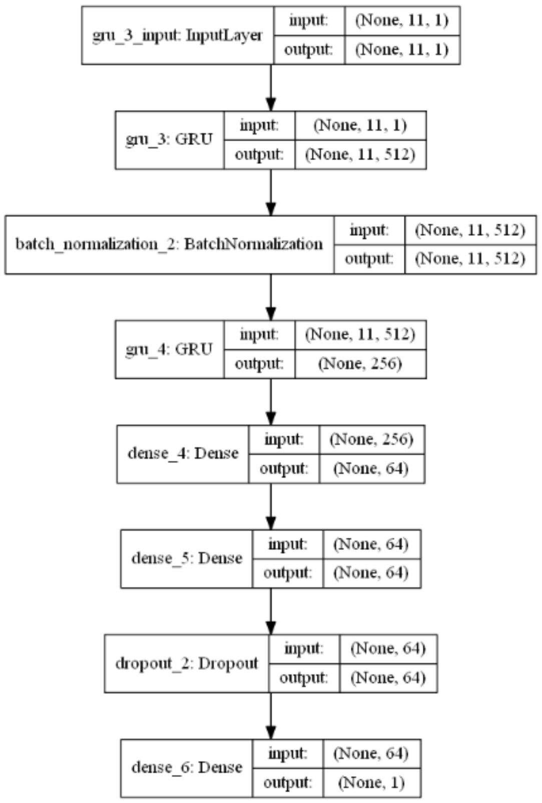


Figure 8. GRU implementation

-Visualization: Matplotlib

-Others: NumPy, Pandas

-Environment: Jupyter notebook, google colab

The source code is available at: `https://github.com/r0b3rt24/Earthquake_Prediction_RNN`

5. Results and Discussion

Table 2 summarizes our results. All four variations of the LSTM and GRU model outperform the benchmark linear regression. LSTM with unidirectional implementation performed the best among all models. It achieves an MAE score of 2.098, which is 1.007 lower than the MAE score achieved by the benchmark linear regression model. Unidirectional GRU performed relatively poor among all four models with an MAE of 2.3167, which is 0.7887 lower than the benchmark score. This result aligns with the result achieved by Althelaya et al (2018) on stock market forecasting as discussed in the proposed method section[2].

It is unclear whether implementing a bidirectional model would lead to better performance. The unidirectional model performed better in the validation set for LSTM, whereas the bidirectional model performed better in the validation set for GRU. An argument in both ways could be made. One possible reason that bidirectional GRU performed better than unidirectional GRU could be the seismic sequence pattern is captured in more detail. The model now captures the sequence pattern not only in the forward direction where the spike of the seismic wave goes before earthquake occurrence but also captures how seismic waves behave as the earthquake is getting further. On the other hand, one possible explanation that unidirectional LSTM performed better than bidirectional GRU could be that the cyclical component of the seismic waves is too chaotic that analyzing the trend in both directions so more noise is added to the pattern.

Figure 9 shows the loss function from unidirectional LSTM and figure 10 shows loss function from bidirectional LSTM. There is a large drop in MAE in the first 20 epoch. Then, the model gradually learns and begins to converge after the 120 epoch. Bidirectional LSTM seems to converge a little faster than unidirectional LSTM. In addition, we observe that the validation set achieves a lower MAE than training set in both models. This could because we use dropout with probability 0.2 after each layer, thus overfitting is reduced substantially.

The loss function for bidirectional GRU and unidirectional GRU is presented in figure 11 and figure 12. There is a large drop in MAE in the first 40 epoch, and the model learns gradually during epoch 40 and 100. The model then converges at the end. Different from the LSTM model, the loss function for the validation set is higher than in the training set. The difference between the validation set and training set is very small, which suggests that our implementation of dropout and batch normalization to reduce overfitting is effective in both models.

The results for test set are presented in table 3. As discussed in the evaluation section, we use MAE for the private test as it is a more accurate evaluation of our models. LSTM with unidirectional implementation performed the best, whereas unidirectional GRU performed relatively poorly. Such results are aligned with results from the validation set.

	LSTM (Bidirectional)	LSTM (Unidirectional)	GRU (Bidirectional)	GRU (Unidirectional)	Linear Regression
MSE on training data set	2.2599	2.2704	2.1462	2.1775	2.8061
MSE on validation data set	2.1339	2.09792	2.2842	2.3167	3.1054

Table 2. Result table

	LSTM (Bidirectional)	LSTM (Unidirectional)	GRU (Bidirectional)	GRU (Unidirectional)
Private testing	2.68	2.63	2.79	3.17
Public testing	1.58	1.62	1.65	1.78

Table 3. Result table

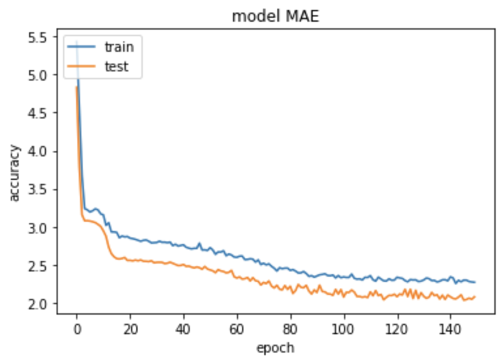


Figure 9. LSTM loss function: unidirectional

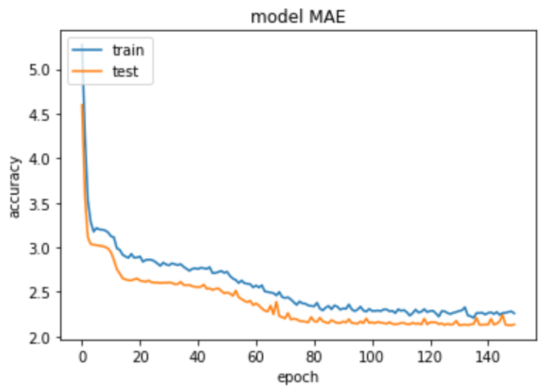


Figure 10. LSTM loss function: bidirectional

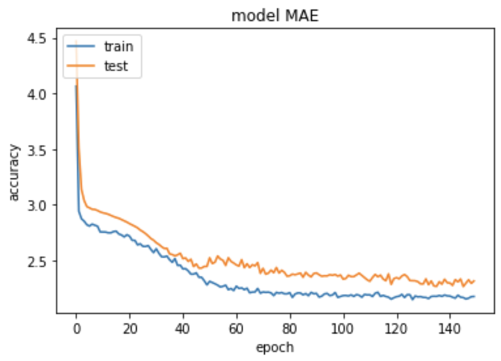


Figure 11. GRU loss function: unibidirectional

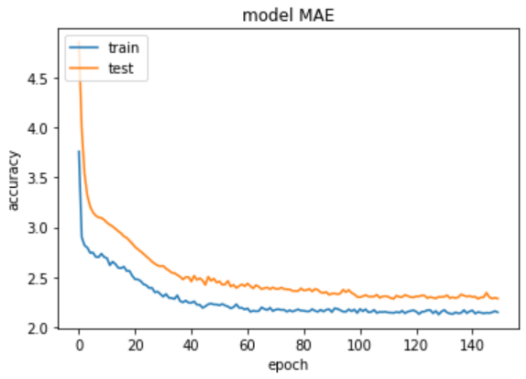


Figure 12. GRU loss function: bidirectional

6. Conclusions

In this project, we conducted experiments to forecast the time until the next laboratory earthquake, and we evaluated the performance of different types of RNN using bidirectional LSTM and bidirectional GRU. A simple benchmark linear regression model is used to evaluate the forecasting performance using mean absolute value. The results show that both types of RNN models improved forecasting performance. In addition, networks developed using bidirectional LSTM demonstrated better performance. Although all our models outperformed the baseline linear regression model, there are still many areas that we want to explore further.

For future steps, we would like to apply more deep learning techniques to tune both LSTM and GRU better. We are also interested in exploring more recurrent neural network models, such as a transformer model. Learning of sequential data continues to be a challenging task in machine learning. Applications involving sequential data may require prediction of new events, generation of new sequences, or decision making such as classification of sequences or subsequences. We would like to utilize RNN in other problems that involve time series analysis as well.

7. Acknowledgements

We thank Dr. Sebastian Raschka for his guidance and support.

8. Contributions

Susan Jiao proposed the topic and wrote the report. She built and tuned the GRU model and helped with unidirectional LSTM.

Lynette Gao built and tuned the LSTM model.She provided great help with tuning the GRU model, and she also contributed in writing and refining the report.

Han Cao contributed in building the LSTM and GRU. He gave meaningful feedback to the report refinement.

William Han helped with building and tuning the GRU model, and he helped refining the report and checked details within the report.

References

[1] H. Adeli and A. Panakkat. A probabilistic neural network for earthquake magnitude prediction. *Neural networks*, 22(7):1018–1024, 2009.

[2] K. A. Althelaya, E. M. El-Alfy, and S. Mohammed. Evaluation of bidirectional lstm for short-and long-term stock market prediction. In *2018 9th International Conference on Information and Communication Systems (ICICS)*, pages 151–156, 2018.

[3] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[4] R. Jozefowicz, W. Zaremba, and I. Sutskever. An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350, 2015.

[5] M. Mirrashid. Earthquake magnitude prediction by adaptive neuro-fuzzy inference system (anfis) based on fuzzy c-means algorithm. *Natural hazards*, 74(3):1577–1593, 2014.

[6] M. D. Petersen, T. Cao, K. W. Campbell, and A. D. Frankel. Time-independent and time-dependent seismic hazard assessment for the state of california: Uniform california earthquake rupture forecast model 1.0. *Seismological Research Letters*, 78(1):99–109, 2007.

[7] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, Nov 1997.