Neural Network Stacking of Time Series Forecasting Models

Cindy Le [1 *] **Alice Chen** [1 *] **Chunlin Zhu** [1 *] **Darshan Thaker** [1] **Iddo Drori** [1]

Abstract

This work focuses on predicting rare events from time series data. We use a neural network to stack diverse and strong predictors, which include recent sequence models. Stacking allows us to easily combine many models, that together result in excellent performance. We demonstrate the applicability of our approach in forecasting when earthquakes will occur from real-time seismic data, ranking within the top 2% out of 3,495 teams in the Kaggle LANL earthquake prediction competition.

1. Introduction

An important challenge in dealing with time series data is forecasting the time of rare events such as earthquakes, for which prediction is crucial, but data is limited. Earthquake prediction has been an important component of seismology for many decades. In this work, we focus on the earthquake prediction task from laboratory produced real-time acoustic data simulated by two fault gouge layers with a certain load and a shear velocity (Leeman et al., 2016). While the simulation model is simplified in the lab setting, it shares many traits with real-world data. The goal of the task is to predict the time to an earthquake from the end time of each segment of the test seismic data.

In this work, we explore several prediction models for this task, ranging from neural network sequence models, genetic programming, and gradient boosting. While each model performs well on the prediction task, we find that these models have different limitations that hinder their performance. As expected, combining these models using a stacked neural network is crucial in obtaining an overall low error. Specifically, we explore the effectiveness of stacked generalization performed using neural networks (Ghorbani & Owrangh, 2001). Given a set of diverse predictors, we combine their results using a fully connected neural network, which can be viewed as a generalization of the weighted ensemble approach. We focus on stacking diverse strong predictors including recent sequence models as described in Section 2.

1.1. Related Work

Work in earthquake prediction dates back to the 1960s when researchers first realized that with sufficient data, it would be possible to predict earthquakes (Rikitake, 1968). In 1973, Scholz et al. discovered that there are precursory effects before almost all shallow earthquakes, which makes earthquake prediction possible (Scholz et al., 1973). Despite an initial positive outlook, subsequent work debated whether earthquakes are predictable. Geller et al. suggested that it is not possible to predict earthquakes since earthquakes result from sudden slips on a geological fault, which are notoriously intractable (Geller et al., 1997). Yet, Wyss et al. pushed back that the ruptures are not as sudden as Geller claimed them to be. In 10% to 30% of large earthquakes, there are strong foreshocks as precursors (Wyss, 1997). Kagan et al. claimed that while it is quite hard to predict earthquakes using solely precursors, real-time seismology can be used to predict the shaking after big earthquakes several seconds before the physical event (Kagan, 1997).

There have been many works that present models to tackle the earthquake prediction task. Gerstenberger et al. use a short-term clustering model to predict the probability of strong ground shaking anywhere in California within the next 24 hours during major earthquake sequences (Gerstenberger et al., 2005). Alves et al. use neural networks to integrate different physical precursors to narrow the time window of the predicted earthquake (Alves, 2006). Helmstetter et al. develop a time-independent forecast that includes smaller earthquakes in magnitude and outperforms other time-independent models (Helmstetter et al., 2006). Schorlemmer et al. present a method for estimating earthquake detection probabilities that avoid assumptions about earthquake occurrence and uses only empirical data, which could be used in regions with sparse data (Schorlemmer & Woessner, 2008).

More recently, Wang et al. (Wang et al., 2017) employ long short-term memory (LSTM) networks to learn the

[1]Columbia University. Correspondence to: Cindy Le <xl2738@columbia.edu>, Alice Chen <mc3197@columbia.edu>, Chunlin Zhu <cz2487@columbia.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

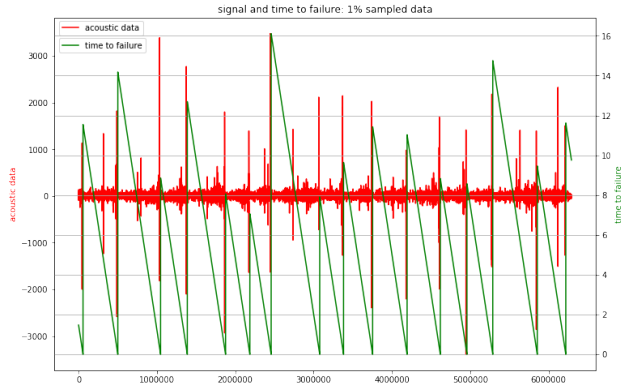

Figure 1. Input acoustic signal (red) and ground truth time to failure (green). The horizontal axis denotes samples out of 600,000. The vertical axis for acoustic signal is the amplitude of the seismic signal, and the vertical axis for the ground truth is time in seconds.

spatiotemporal relationship among earthquakes in different locations and improve predictions. Asim et al. (Asim et al., 2018) coin GP-AdaBoost which combines seismic indicators along with Genetic Programming (GP) and the AdaBoost ensemble method to provide predictions for earthquakes of magnitude 5.0 and above fifteen days prior to the earthquake. Earthquake prediction is also becoming an ethical issue. Scientists discuss how earthquake scientists conceptualize earthquake prediction given that the predictions may not be accurate and have severe consequences (Joffe et al., 2018). Ross et al. (Ross et al., 2018) train recurrent neural networks to link phases together that share a common origin. The method is simple to implement for any tectonic regime, suitable for real-time processing, and can naturally incorporate errors in arrival times.

2. Methods

In the following sections, we first describe the earthquake prediction task along with the dataset and feature extraction process. Next, we describe a general method to stack a set of diverse models trained for the task. Finally, we discuss the specific models stacked.

2.1. Dataset and Feature Extraction

The training data is a single sequence of the seismic signal that contains around 600 Million data points of signal value and their corresponding time intervals. We observe that each test data is a chunk of $150,000$ single data points so during the training process, we also consider one chunk of $150,000$ data points as a single training sample. Figure 1 shows the acoustic signals and time to failure. The horizontal axis denotes samples out of 600,000. The vertical axis for the acoustic signal (in red) is the amplitude of the seismic signal, and the vertical axis for the ground truth (green) is time in seconds.

A key challenge is avoiding overfitting since the number of data chunks in the earthquake prediction task is low. Instead of dividing the 600 Million datapoints into 4,000 data chunks, we consider each data point as a possible end point of a training sample. We generate training samples on the fly during training by randomly choosing a batch size of end points within each step. Thus, there are more than 600 million possible training samples available, many of which overlap significantly. Since there are so many possible training samples, we cannot extract the features beforehand, but need to extract features on the fly. We extract features from sliding windows of size 1000. We first standardize the data to a mean of 5 and a standard deviation of 3. We then extract the mean, standard deviation, minimum, maximum, sum, minimum of absolute values, maximum of absolute values, $1\%, 5\%, 95\%, 99\%$ quantiles, and the mean differences between a data point and its previous one (12 features in total). We then calculate the same 12 features for the last 100 data points of each sliding window; we also calculate the same 12 features for the last 10 data points of each sliding window. This process gives us 36 features in total for each sliding window. One training sample consists of $150,000$ data points and is divided into 150 sliding windows. We stitch these features together to form a matrix of dimensions 150×32. Each training data point is thus such a matrix with features extracted from the original acoustic data.

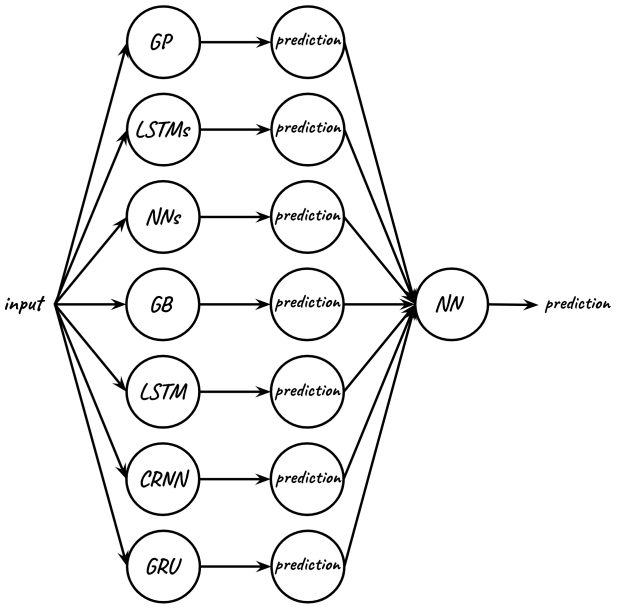

Figure 2. Neural network stacking architecture.

2.2. Stacking

Next, we describe our procedure for stacking a set of diverse models. The key insight is that each model trained for a task has its specific limitations, which can vary from model to model. Thus, in order to boost accuracy as well as average out noisy predictions across models, we adopt a ensemble models that combines the predictions of the various models. We experiment with three ensemble models.

Classical Ensemble: A standard approach to taking an ensemble of models is to make final predictions by averaging over the class probabilities for every model. Specifically, assuming each model i outputs a vector of class probabilities p_i over the class set $\mathcal{C}$, the final prediction y is computed as

$$y = \arg\max_{j \in \mathcal{C}} \frac{1}{m} (\sum_{i=1}^{m} p_i^{(j)}) \quad (1)$$

Weighted Average Ensemble: We can augment the classical ensemble approach by taking a weighted average of models. As a heuristic for rewarding better models, we set these weights to be inversely proportional to the MAE of each model. However, the problem with this model is the manual setting of weights, which may be sub-optimal.

Neural Network Stacking: In this approach, we use a neural network to combine the predictions of several models. More formally, denote $m_1, \ldots, m_n$ as n different models. Using these models, we train a stacking neural network, with dense layers $f_1, \ldots, f_L$, whose input is the concatenated outputs of $m_1, \ldots, m_n$. The final prediction $F(x)$ for an input x is then

$$F(x) = f_1(f_2(f_3 ... f_L([m_1(x), ..., m_n(x)]))) \quad (2)$$

This architecture is shown pictorially in Figure 2 for the sub-models defined for the earthquake prediction task. In this case, the number of models is 7 and the number of stacked layers is 3. Note that when training the stacking neural network, the models $m_1, \ldots, m_n$ are kept constant, and only the parameters of the dense layers are trained. For a one-layer neural network, this is equivalent to learning the weights of a weighted ensemble. Adding more dense layers to the stacking neural network corresponds to a nonlinear generalization of the weighted ensemble approach, allowing the model to discover underlying correlations between model outputs more effectively.

2.3. Models

In this section, we describe the various models that we use for the earthquake prediction task. The goal of using many different models, varying from simple fully connected neural networks to complex convolutional recurrent neural networks, is to encourage diversity in the types of models employed.

Multilayer LSTMs (Lambert, 2016): Long short-term memory (LSTM) networks are commonly used for prediction over time series data, capturing long-term dependencies across sequences. We use a model consisting of four stacked LSTM layers, each with hidden units of dimensions 64, 48, 48, and 32, followed by two fully connected layers.

Neural Network Ensemble: Fully connected neural networks historically do not perform as well as more complex recurrent models when modelling time series data; however a classical ensemble of dense neural networks serves as a simple baseline model.

LSTMs (Hochreiter & Schmidhuber, 1997): A one-layer LSTM model is used as opposed to multi-layer LSTMs to obtain a simple sequence model.

CRNN (Zuo et al., 2015): This model comprises of two convolution layers, each of which is followed by a max pooling layer. We feed the output of this simple convolution network to an LSTM layer followed by two fully connected layers.

GRU (Cho et al., 2014): Finally, we experiment with Gated Recurrent Units (GRUs), a model commonly used in sequence modelling tasks.

Genetic programming (Asim et al., 2018): We use a pretrained genetic programming model.

Gradient boosting (Ke et al., 2017): We use a model based on lightGBM, which is a gradient boosting method based on one-side sampling and exclusive feature bundling to accelerate the training process, while maintaining accuracy.

3. Results

Our prediction results are evaluated in the Kaggle competition using the mean absolute error (MAE) of the difference between the predicted earthquake time and ground truth time. Table 1 shows the performance of our neural network stacking and of each individual model. As shown, neural network stacking outperforms each individual model, achieving the lowest MAE of 1.419.

Training each individual model takes a few hours, and training the neural network stacking takes under an hour. Prediction is performed under a minute per sample. All models are trained using the same features for 10 epochs with batch size 32.

We also compared the performance only for the sequence models, excluding genetic programming and gradient boosting. Neural network stacking of the sequence models only results in MAE of 1.443 whereas a weighted average ensem-

Model	MAE
Neural network stacking	1.419
Genetic programming	1.435
Multi-layer LSTMs	1.495
Neural network ensemble	1.501
Gradient Boosting	1.507
LSTM	1.509
CRNN	1.510
GRU	1.536

Table 1. Comparison of mean absolution error (MAE) of neural network stacking and individual models.

ble of sequence models only results in MAE of 1.477. The best result is obtained by the neural network stacking of all individual models for an MAE of 1.419, which currently ranks 77 out of 3,495 teams in the Kaggle LANL earthquake prediction competition (Los Alamos National Laboratory, 2019).

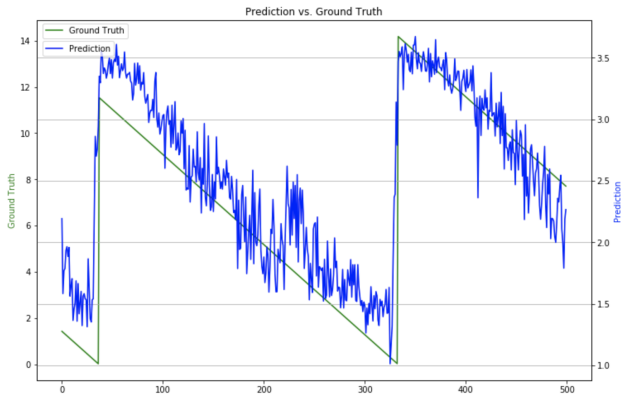

Figure 3. Prediction result: ground truth times are shown as spikes (green) compared with predicted results (blue). The horizontal axis represents individual samples zooming in on a segment of 500 samples, and the vertical axis represent time in seconds.

4. Conclusions

Our neural network stacking model for earthquake prediction achieves results which are better than each of the individual sub-level models we used. This model ranks within the top 2% in a competition consisting of 3,495 teams. Our ensemble of diverse models is effective in averaging out much of the noise within the training data, and the different strong models complement each other in different parts of the space. A limitation of our approach is that it requires a large training set for both the individual models and a validation set for training the stacking neural network. In future work, we would like to improve our feature extraction for the earthquake prediction problem as well as apply our method to other time series datasets.

References

Alves, E. I. Earthquake forecasting using neural networks: results and future work. *Nonlinear Dynamics*, 44(1-4): 341–349, 2006.

Asim, K. M., Idris, A., Iqbal, T., and Martínez-Álvarez, F. Seismic indicators based earthquake predictor system using genetic programming and AdaBoost classification. *Soil Dynamics and Earthquake Engineering*, 111:1–7, 2018.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734, 2014.

Geller, R. J., Jackson, D. D., Kagan, Y. Y., and Mulargia, F. Earthquakes cannot be predicted. *Science*, 275(5306): 1616–1616, 1997.

Gerstenberger, M. C., Wiemer, S., Jones, L. M., and Reasenberg, P. A. Real-time forecasts of tomorrow's earthquakes in california. *Nature*, 435(7040):328, 2005.

Ghorbani, A. A. and Owrangh, K. Stacked generalization in neural networks: generalization on statistically neutral problems. In *International Joint Conference on Neural Networks*, volume 3, pp. 1715–1720, 2001.

Helmstetter, A., Kagan, Y. Y., and Jackson, D. D. Comparison of short-term and time-independent earthquake forecast models for southern california. *Bulletin of the Seismological Society of America*, 96(1):90–106, 2006.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Joffe, H., Rossetto, T., Bradley, C., and O'Connor, C. Stigma in science: the case of earthquake prediction. *Disasters*, 42(1):81–100, 2018.

Kagan, Y. Y. Are earthquakes predictable? *Geophysical Journal International*, 131(3):505–525, 1997.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pp. 3146–3154, 2017.

Lambert, J. Stacked RNNs for encoder-decoder networks: Accurate machine understanding of images, 2016. URL `cs224d.stanford.edu/reports/Lambert.pdf`.

Leeman, J., Saffer, D., Scuderi, M., and Marone, C. Laboratory observations of slow earthquakes and the spectrum of tectonic fault slip modes. *Nature communications*, 7: 11104, 2016.

Los Alamos National Laboratory. Kaggle LANL earthquake prediction competition, 2019. URL www.kaggle.com/c/LANL-Earthquake-Prediction.

Rikitake, T. Earthquake prediction. *Earth-Science Reviews*, 4:245–282, 1968.

Ross, Z. E., Yue, Y., Meier, M., Hauksson, E., and Heaton, T. A deep learning approach to seismic phase association. In *AGU Fall Meeting Abstracts*, 2018.

Scholz, C. H., Sykes, L. R., and Aggarwal, Y. P. Earthquake prediction: a physical basis. *Science*, 181(4102):803–810, 1973.

Schorlemmer, D. and Woessner, J. Probability of detecting an earthquake. *Bulletin of the Seismological Society of America*, 98(5):2103–2117, 2008.

Wang, Q., Guo, Y., Yu, L., and Li, P. Earthquake prediction based on spatio-temporal data mining: An LSTM network approach. *IEEE Transactions on Emerging Topics in Computing*, 2017.

Wyss, M. Cannot earthquakes be predicted? *Science*, 278 (5337):487–490, 1997.

Zuo, Z., Bing, S., Gang, W., Xiao, L., Xingxing, W., Bing, W., and Yushi, C. Convolutional recurrent neural networks: Learning spatial dependencies for image representation. In *Conference on Computer Vision and Pattern Recognition*, pp. 18–26, 2015.