Earthquake Prediction: Dictionary-Based Classifiers

Kyle Masters
Department of Computer Science
Utah State University
Logan, UT 84322
the.kyle.masters@gmail.com

***Abstract*—Dictionary based classifiers are widely used within time series classification. They work by finding the frequency of generalized pattern occurrences within time series and classifying based on that. Due to finding patterns that may be able to be seen by plotting a given time series, these classifiers can be considered as fairly interpretable. Earthquakes, while a common occurrence, are currently difficult to predict in a meaningful way. This issue is more easily seen among earthquakes with high magnitudes, which happen much less frequently. In this paper, I will explore different univariate applications of dictionary based classifiers on predicting if an earthquake of a certain magnitude will occur within a time period.**

***Index Terms*—Univariate Time Series classification, Dictionary based classifiers**

I. Introduction

Damage from earthquakes is a prevalent problem in many parts of the world, with an estimated 500,000 detectable earthquakes occurring each year [1]. Many of these earthquakes can't be felt and only a fraction of them cause damage. Even so, losses from earthquakes are estimated at $4.4B annually, with a significant portion of these damages being localized to California [2]. Different responses in preventing damages from earthquakes have been attempted in the past, with limited success. These responses may benefit from better prediction of when and where major seismic events will happen, but unfortunately with current methods it isn't possible to predict these events in any meaningful way [3].

Over the past decades, a large amount of data has been gathered and compiled regarding seismic activity around the world by different groups. One of these such groups compiled the STEAD Dataset, which comprises data on over 500,000 earthquakes [4]. With such a large amount of data becoming readily available, hopefully it will allow for advances in this area. Motivated by this amount of data, I utilized dictionary based classifiers for time series on this data after performing necessary transformations.

The rest of this paper is organized as follows: Sec 2. covers previous applications of time series analysis to the prediction of seismic events; Sec 3 dataset used and the preprocessing needed to analyze this data with time series methods; Sec. 4 provides a background on each of the methods used; Sec. 5 reports the results of these methods with different parameters; Sec. 6 contains a summary of my findings and discusses potential future work.

II. Related Work

While there is a large amount of information regarding earthquake prediction that has been published and studies that have been done, I will limit the related work I cover with 2 articles that inspired this work.

The first uses a hybrid clustering-LSTM approach [5]. While this approach differs from my own approaches, this article discussed important information necessary for any time series analysis on earthquake data. This article gave a background of the issue and discussed that major earthquakes can happen without consistent precursor events, which complicates the issue of prediction. It also discussed gathering the data and that the data is time-stamped and has irregular intervals, which further complicates using traditional time series methods to analyze this data.

The second uses an autoregressive integrated moving average model [6], which again is an approach that differs from my own. Within this article, it gave some information on dealing with the issues that were previously discussed about earthquake data being collected at irregular intervals. The authors of this paper created time series data by gathering information on the amount of earthquakes that occur between pre-set time steps leading up to major events. This paper also concluded with the fairly successful prediction of earthquakes of a magnitude of 8 or greater within a year.

The main inspirations I got from these sources are:

- Limit the area of analysis to major earthquake zones.
- Time series can be constructed based on events occurring in pre-set time intervals.
- Selection of time range for event prediction is important.

III. Data Used

A. STEAD Dataset

The dataset that I conducted my experiments on is the STanford EArthquake Dataset (STEAD), which is a publicly available dataset hosted on Github [7]. This data contains information about events gathered from different sensors around

the globe. For my purposes, I wasn't concerned about the sensors used so the relevant information from this dataset was the timestamp of an event, the latitude and longitude coordinates of an event, and the magnitude of an event. Within this dataset, times were in a universal timezone and all other pieces of data used the same scale for each datapoint.

B. Preprocessing Steps

To prepare this dataset, I used some techniques inspired by the second article discussed in Sec. 2.

1) The first step was eliminating any erroneous data; with many events being captured by multiple sensors, there were occurrences when different times and magnitudes were recorded. In all cases, this was one sensor having different information recorded than others and this difference was normally small ($<$ 1 sec, $<$ magnitude 0.1). This was corrected by collecting the data for each event as recorded by a majority of the sensors.
2) After this, events were selected to be included by their location. These locations were divided based on the whole number latitude and longitude coordinates. Locations within the western United States were included if they had 10,000 events recorded during the years 2011-2018.
3) Time series were then created with the included earthquakes. This was done by selecting an interval to predict for, which I selected as 1 week. I also selected multiple time lengths and weekly time steps to include in each time series leading up to prediction. I selected 3 combinations of these to test on: (9 weeks, 14 time steps per week); (9 weeks, 168 time steps per week); and (17 weeks, 14 time steps per week). For each of these combinations, 3 time series were created for each week from 2011-2018. The first time series had data points as the average magnitude between time steps, the second looked at amount of earthquakes between time steps, and the last recorded the maximum magnitude occurring between time steps. A sample time set of time series is shown in Fig. 1.
4) These time series were then separated into two classes. A positive class had an earthquake with a magnitude 3 or higher occur during the prediction week while a negative class did not.

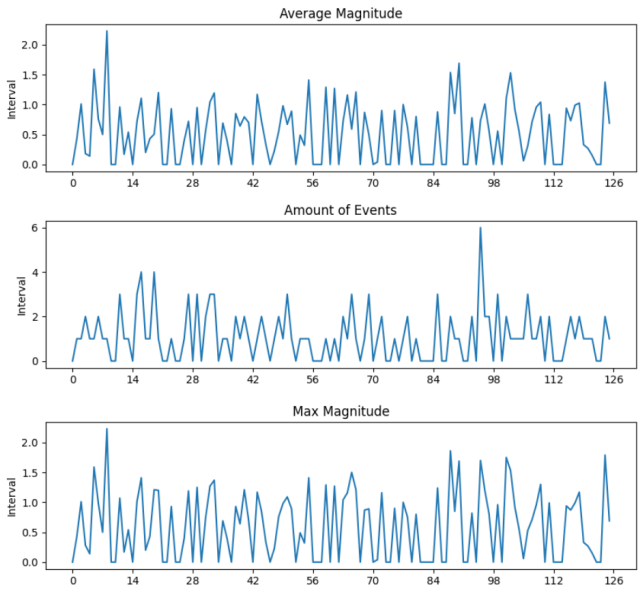

Fig. 1: Example time series after preprocessing

IV. Methodology

A. Bag of Patterns

Bag of patterns is a method that extracts subsequences from the time series and transforms each subsequence into a word representation using Piecewise Aggregate Approximation. This is a method that calculates the means over different windows to reduce noise. It then calculates the frequency of each word for each series. This is a common method that's seen applications in natural language processing [8] as well as medical classification [9]. This method is based on the fundamental idea that events of similar classes will, in general, have similar features that appear at similar frequencies. For earthquake prediction, this method would be able to determine if intervals with major earthquakes are identified by a change in frequency, average magnitude, or higher magnitude events in the intervals leading up to that event. This method is a feature generation method that can then be fed into other classifiers. In my methodology, I used a Random Forest Classifier to classify each time series after transforming it with a bag of patterns. The bag of patterns implementation used was through the pyts python package [10].

B. Bag-of-SFA Symbols in Vector Space

Similar to the Bag of Patterns method, the Bag-of-SFA Symbols in Vector Space (BOSSVS) algorithm transforms each time series into a word representation. As opposed to using Piecewise Aggregate Approximation, this method instead uses Symbolic Fourier Approximation, an approximation algorithm based on fourier transforms over different windows [11]. After each time series is represented using word frequency, all of these histograms are used to calculate a tf-idf vector and predict classes based on this vector [12]. This method provides some noise reduction through the use of the Symbolic Fourier Approximation, which may be helpful in a subject with such high variance as seismology. Because of the use of the tf-idf vector to classify, this method provides some speed improvements over other methods [12]. In a real world setting, this could prove important for earthquake prediction in a time-sensitive environment. The implementation used was through the pyts python package [10].

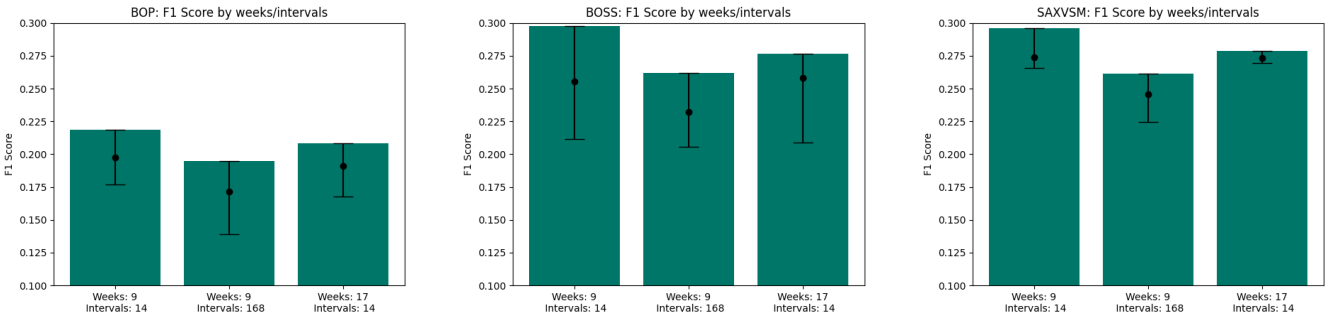

Fig. 2: F1 Scores by model. Error bars indicate max, mean, and minimum F1 scores with different hyperparameters used.

C. SAXVSM

In the SAXVSM method, each series is transformed into a word-frequency representation using the Symbolic Aggregate Approximation algorithm. It is similar to the BOSSVS method in that it uses a tf-idf vector to classify each series after its word-frequency representation. This method has shown that it is not subject to noise as other methods [13]. The implementation used was through the pyts python package [10].

V. Results

A. Experimental Setup

As mentioned previously, there were 3 sets of time series created with the variables of included preceding weeks and weekly intervals changed. In addition, due to the nature of this dataset, these time series originally had a class imbalance because not every week has a major earthquake. This class imbalance was about 12% positive, 88% negative. This class imbalance was left intact for these experiments.

For each of these tests, each dictionary-based classifier was used to predict earthquakes for each week over the years 2011-2018. The hyper-parameters changed to evaluate prediction in each of these situations included changing the word sizes in the set [10, 30, 50], the window size for the transformation function used in the set [0.2, 0.5], the amounts of bins for each algorithm to produce within the set [10, 14], and the functions to evaluate being the average magnitude of events, amount of events, and maximum magnitude of events in each weekly interval. The results reported with predictions were accuracy, recall, and F1 Score, with F1 score being the most focused.

B. Results Overview

After results for each of the tests were calculated, the best performance (based on F1 score) of each method for each week-interval combination was compared. Among these, it was found that the combination of 9 weeks and 168 intervals yielded the worst results and the combination of 9 weeks and 14 intervals yielded the best results for all methods used. Between the methods, Bag of Patterns performed notably worse than the others, with the others performing very similarly at least in the best cases. BOSSVS had lower mean and minimum scores based on variations in hyperparameters than SAXVSM did. The bar plots in Fig. 2 shows the F1 scores achieved by each model, along with error bars indicating the mean and minimum scores of each method.

C. Bag of Patterns

The Bag of Patterns method yielded the worst results, but it's still interesting to look the specifics of its results. The results of the two best performing models within the Bag of Patterns method are shown in Table 1. These models both share the hyperparameters word_size = 10, window_size = 0.5, function = average magnitude. The best model had 14 bins and the second best had 10 bins. Of all the hyperparameters for this method, the most impactful on the success of this model was the window_size. The best model with a 0.5 window_size had an F1 score of 6.5% higher than the best model with a 0.2 window_size. The best values for hyperparameters and the amount higher than the next best value for that hyperparameter are shown in Table 2.

Compared to the other two methods utilized, bag of patterns had the lowest rate of positive classification, with an average of 8.8% of time series being classed as positive. This is close to the 12% of actually series that are positive. Unfortunately, this closeness to the actual ratio of positive classification did not seem to contribute to this method outperforming the others.

TABLE I: Best Performing BOP Results

Model	Accuracy	Recall	**F1**
#1	0.835	0.190	**0.218**
#2	0.838	0.183	**0.214**

D. BOSSVS

The BOSSVS method yielded the best results, with an F1 score of 0.298 compared, which was 36.4% better than the Bag of Patterns best result and 0.6% better than the

TABLE II: BOP Hyperparameters Affect on Results

Hyperparameter	Best Value	**F1 Increase**
word_size	10	**5.4%**
window_size	0.5	**6.5%**
bins	14	**1.9%**
function	Average Magnitude	**5.4%**

SAXVSM best result. The results of the two best performing models within the BOSSVS method are shown in Table 3. These models both share the hyperparameters word_size = 10, window_size = 0.2, function = maximum magnitude. The best model has 14 bins and the second best has 10 bins. The most impactful hyperparameters on this model's success were word_size and window_size, each with a maximum F1 score increase between values of 11.6%. The best values for hyperparameters and their percentage increase for the BOSSVS models are shown in Table 4.

Though the BOSSVS method yielded the best results, it was only 0.6% better than the SAXVSM model with significantly higher variation in performance. Across all BOSSVS models, the average F1 score was 0.255, compared to the average among SAXVSM being 0.274. The BOSSVS scores also had about twice the standard deviation among them than the SAXVSM scores. These variations indicate that the BOSSVS models are significantly more susceptible to being affected by poor hyperparameter tuning than the similarly performing SAXVSM models.

TABLE III: Best Performing BOSSVS Results

Model	Accuracy	Recall	**F1**
#1	0.681	0.560	**0.298**
#2	0.680	0.544	**0.291**

TABLE IV: BOSSVS Hyperparameters Affect on Results

Hyperparameter	Best Value	**F1 Increase**
word_size	10	**11.6%**
window_size	0.2	**11.6%**
bins	14	**2.4%**
function	Max Magnitude	**2.7%**

E. SAXVSM

The SAXVSM method yielded similar results to the BOSSVS method with a best F1 score of 0.296. The results of the two best performing models within this model are shown in Table 5. These models share the hyperparameters word_size = 10 and window_size = 0.2. The best model has 10 bins and uses the function count, while the second best has 14 bins and uses the function average magnitude. This model's most impactful hyperparameters was the word_size, with a maximum F1 score of the among the best performing value for this hyperparameter being 10.9% higher than the next best. The best values for hyperparameters and their percentage increase for the SAXVSM models are shown in Table 6.

Unlike the other methods, which had a noticeable difference in their scores generated between time series comprised of the average magnitude, event count, or maximum magnitude, SAXVSM did not. The difference between F1 scores among this hyperparameter was the smallest of any hyperparameter for any model. This difference was only 0.1%.

TABLE V: Best Performing SAXVSM Results

Model	Accuracy	Recall	**F1**
#1	0.678	0.201	**0.296**
#2	0.678	0.201	**0.296**

TABLE VI: SAXVSM Hyperparameters Affect on Results

Hyperparameter	Best Value	**F1 Increase**
word_size	10	**10.9%**
window_size	0.2	**9.8%**
bins	14	**0.1%**
function	Event Count	**0.1%**

VI. Conclusion

A. Reflection

In this work, I set out to attempt to predict if an earthquake above a certain magnitude could be predicted using time series dictionary based methods. All of these results for the parameters (Weeks = 9, Intervals = 14) are recorded in Table 7. While results were far from perfect, there were differences shown between the performance of the different methods. The Bag of Patterns performed significantly worse as noted before compared to both of the other methods.

Interestingly enough, this method also had the best accuracy with the accuracy of this method ranging around 0.830. The BOSSVS and SAXVSM both had accuracies with 0.7 being the high end. These methods had a higher F1 Score and so at first thought one might think that accuracy and F1 Score would have a negative correlation, but this isn't necessarily the case. For the Bag of Patterns and BOSSVS methods, these two metrics had a positive correlation as shown in Figs. 3-4. In the SAXVSM method there was a negative correlation, though it was noticed that all of the points were clustered very close around high accuracy-low F1 and low accuracy-high F1.

In terms of real-time application, I believe that the SAXVSM would be the best method. Though it did technically perform worse than the BOSSVS models, as discussed earlier, these models were much less prone to less accurate predictions by sub-optimal hyperparameters. Due to this, a SAXVSM based model may be more better utilized across a broad range of areas that may have differing frequency of high-magnitude events. I don't see a real-time application for the Bag of Patterns models, seeing that even the best BOP models

performed worse than the worst models from the other two methods.

B. Future Work

There were a few areas in which the work from these experiments could be expanded upon. The first is that this experiment could be expanded. A small amount of both hyperparameters and experiment parameters were tested, and better results may be gained from redoing these experiments with an increased amount of these. For the hyperparameters, it appeared that smaller window sizes, smaller word sizes, and more bins yielded the best results so it would be interesting to see what hyperparameters were the best within a small interval bounded closer to the best hyperparameter values from these experimenets. It would also be interesting to see what different paramaters on the experiments could be tested. Both smaller and larger amounts of weeks included in each time series could be tested, and testing different combinations of weekly intervals with those might show some more insightful results. Unfortunately, one area that limited this experiment was the processing power available.

The second is that further analysis could be done in analyzing these results. While the hyperparameter performance and performance in the experiment paramaters was measured, there was no analysis done into what features exactly each model is finding that may help to predict an earthquake. To this end, identifying if there are particular shapes (or words) that might determine if an earthquake is about to happen may help further the use of other methods as well.

The last is that all of these methods were univariate methods that were testing different features of these time series. Each of these features (average magnitude, event count, and maximum magnitude) were used to gain some successful prediction results, and because of this, they may tell a more complete story together. In the spirit of using dictionary based methods for these experiments, using a multi-variate based classifier such as WEASEL-MUSE or performing functions on the features to estimate a single feature from all of them would be a logical next step.

References

[1] "Cool earthquake facts," Cool Earthquake Facts — U.S. Geological Survey. [Online]. Available: https://www.usgs.gov/programs/earthquake-hazards/cool-earthquake-facts.

[2] B. Arguero, Annual U.S. earthquake losses estimated at $4.4B. [Online]. Available: https://www.govcon.com/doc/annual-us-earthquake-losses-estimated-at-44b-0001.

[3] "Can you predict earthquakes?," Can you predict earthquakes? — U.S. Geological Survey. [Online]. Available: https://www.usgs.gov/faqs/can-you-predict-earthquakes.

[4] S. M. Mousavi, Y. Sheng, W. Zhu and G. C. Beroza, "STanford EArthquake Dataset (STEAD): A Global Data Set of Seismic Signals for AI," in IEEE Access, vol. 7, pp. 179464-179476, 2019, doi: 10.1109/ACCESS.2019.2947848.

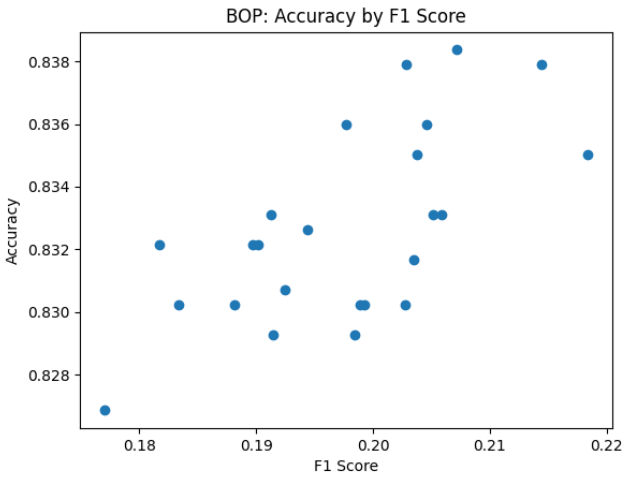

Fig. 3: BOP Accuracy by F1 Score: Positive correlation

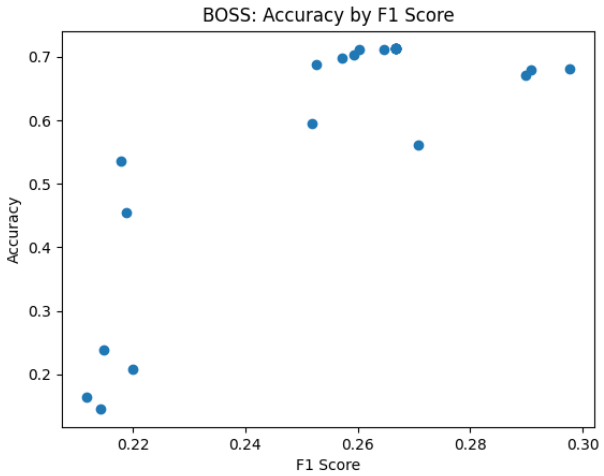

Fig. 4: BOSSVS Accuracy by F1 Score: Positive correlation

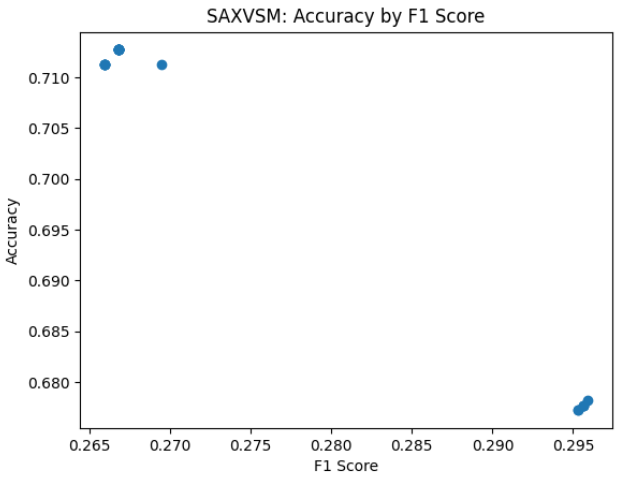

Fig. 5: SAXVSM Accuracy by F1 Score: Negative correlation

TABLE VII: Results for Weeks=9 and Intervals=14

Word Size	Window Size	Bins	Function	BOP Accuracy	BOP Recall	BOP F1	BOSSVS Accuracy	BOSSVS Recall	BOSSVS F1	SAXVSM Accuracy	SAXVSM Recall	SAXVSM F1
10	0.2	10	avg	0.83	0.18	0.21	0.56	0.67	0.27	0.68	0.56	0.30
10	0.2	10	count	0.84	0.17	0.20	0.14	0.96	0.21	0.68	0.56	0.30
10	0.2	10	max	0.83	0.15	0.18	0.68	0.54	0.29	0.68	0.56	0.30
10	0.2	14	avg	0.83	0.17	0.19	0.67	0.56	0.29	0.68	0.56	0.30
10	0.2	14	count	0.83	0.16	0.18	0.21	0.92	0.22	0.68	0.56	0.30
10	0.2	14	max	0.83	0.17	0.20	0.68	0.56	0.30	0.68	0.56	0.30
10	0.5	10	avg	0.84	0.18	0.21	0.60	0.56	0.25	0.71	0.43	0.27
10	0.5	10	count	0.83	0.18	0.21	0.16	0.93	0.21	0.71	0.44	0.27
10	0.5	10	max	0.83	0.15	0.18	0.71	0.43	0.26	0.71	0.43	0.27
10	0.5	14	avg	0.84	0.19	0.22	0.70	0.43	0.26	0.71	0.43	0.27
10	0.5	14	count	0.83	0.16	0.19	0.24	0.86	0.21	0.71	0.43	0.27
10	0.5	14	max	0.83	0.16	0.19	0.71	0.42	0.26	0.71	0.43	0.27
30	0.5	10	avg	0.84	0.17	0.20	0.71	0.43	0.27	0.71	0.43	0.27
30	0.5	10	count	0.83	0.16	0.19	0.46	0.63	0.22	0.71	0.43	0.27
30	0.5	10	max	0.83	0.17	0.20	0.71	0.43	0.27	0.71	0.43	0.27
30	0.5	14	avg	0.83	0.17	0.19	0.71	0.43	0.27	0.71	0.43	0.27
30	0.5	14	count	0.83	0.16	0.19	0.54	0.54	0.22	0.71	0.43	0.27
30	0.5	14	max	0.83	0.17	0.19	0.71	0.43	0.27	0.71	0.43	0.27
50	0.5	10	avg	0.84	0.17	0.20	0.71	0.43	0.27	0.71	0.43	0.27
50	0.5	10	count	0.84	0.17	0.21	0.69	0.44	0.25	0.71	0.43	0.27
50	0.5	10	max	0.83	0.18	0.20	0.71	0.43	0.27	0.71	0.43	0.27
50	0.5	14	avg	0.84	0.17	0.20	0.71	0.43	0.27	0.71	0.43	0.27
50	0.5	14	count	0.83	0.17	0.20	0.70	0.43	0.26	0.71	0.43	0.27
50	0.5	14	max	0.83	0.18	0.20	0.71	0.43	0.27	0.71	0.43	0.27

[5] S. Mighani, "Earthquake time-series forecasts using a hybrid clustering-LSTM approach-PART I: EDA," Medium, 03-Nov-2020. [Online]. Available: https://towardsdatascience.com/earthquake-time-series-forecasts-using-a-hybrid-clustering-lstm-approach-part-i-eda-6797b22aed8c.

[6] A. Amei, W. Fu, and C.-H. Ho, "Time Series Analysis for Predicting the Occurrences of Large Scale Earthquakes," International Journal of Applied Science and Technology, vol. 2, no. 7, pp. 64–75, Aug. 2012.

[7] "Smousavi05/Stead: Stanford earthquake dataset (stead):a global data set of seismic signals for ai," GitHub. [Online]. Available: https://github.com/smousavi05/STEAD.

[8] J. Brownlee, "A gentle introduction to the bag-of-words model," MachineLearningMastery.com, 07-Aug-2019. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-bag-words-model/.

[9] J. Wang, P. Liu, M. F. H. She, S. Nahavandi, and A. Kouzani, "Bag-of-words representation for biomedical time series classification," Biomedical Signal Processing and Control, vol. 8, no. 6, pp. 634–644, 2013.

[10] Johann Faouzi and Hicham Janati. pyts: A python package for time series classification. Journal of Machine Learning Research, 21(46):1-6, 2020.

[11] P. Schäfer, "The Boss is concerned with time series classification in the presence of noise," Data Mining and Knowledge Discovery, vol. 29, no. 6, pp. 1505–1530, 2014.

[12] P. Schäfer, "Scalable time series classification," Data Mining and Knowledge Discovery, vol. 30, no. 5, pp. 1273–1298, 2015.

[13] P. Senin and S. Malinchik, "Sax-VSM: Interpretable time series classification using sax and Vector space model," 2013 IEEE 13th International Conference on Data Mining, 2013.