# SPATIO-TEMPORAL EARTHQUAKE PREDICTION WITH STRUCTURAL RECURRENT NEURAL NETWORKS

# YAPISAL TEKRARLAYAN SİNİR AĞLARI İLE ZAMAN-MEKANSAL DEPREM TAHMİNİ

**AYDIN DOĞAN**

**ASST. PROF. DR. ENGİN DEMİR**

**Supervisor**

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering.

2021

# ÖZET

## YAPISAL TEKRARLAYAN SİNİR AĞLARI İLE ZAMAN-MEKANSAL DEPREM TAHMİNİ

**Aydın DOĞAN**

**Yüksek Lisans, Bilgisayar Mühendisliği**
**Danışman: Dr. Öğretim Üyesi Engin DEMİR**
**Ocak 2021, 56 sayfa**

Deprem tahmin problemi, belirli bir bölgede, minimum Richter büyüklüğünde ve bir zaman aralığında depremin meydana gelme olasılığının tahmini olarak tanımlanabilir. Bu uzun süredir üzerinde çalışılan bir araştırma problemidir, ancak son on yıla kadar çok fazla ilerleme kaydedilememiştir. Hesaplama sistemlerinde ve derin öğrenme modellerindeki gelişmeler ile birlikte önemli sonuçlar elde edilmeye başlanmıştır. Bu tezde, mekansal yakınlığı ve bölgelerdeki fay hatlarının varlığı gibi ön bilgileri dikkate alan, bu bilgileri sistematik olarak işleyen yapısal tekrarlayan sinir ağını (SRNN) kullanan modeller sunulmuştur. Türkiye ve Çin gibi ölçek ve deprem bölgelerinin büyük ölçüde farklı olduğu iki ayrı bölgede deneyler gerçekleştirilmiştir. SRNN modelleri, kıyaslama yapılan baz model ve bilinen en iyi modellere göre daha iyi performans sonuçları elde etmiştir. Özellikle birinci dereceden mekansal komşuluğa ve fay hatlarına dayalı yapısal sınıflandırmaya göre kurgulanan $SRNNClass_{near}$ modeli, en yüksek $F_1$ skoruna ulaşmıştır.

**Anahtar Kelimeler:** Zaman-Mekansal Tahmin, Yapay Sinir Ağları, Deprem Tahmini, Yapısal Tekrarlayan Sinir Ağları

# ABSTRACT

## SPATIO-TEMPORAL EARTHQUAKE PREDICTION WITH STRUCTURAL RECURRENT NEURAL NETWORKS

**Aydın DOĞAN**

**Master of Science**, **Computer Engineering Department**
**Supervisor: Asst. Prof. Dr. Engin DEMİR**
**January 2021, 56 pages**

The earthquake prediction problem can be defined as a given a minimum Richter magnitude scale and a specified geographic region, predicting the possibility of an earthquake in that region within a time interval. This is a long time studied research problem, but not much progress is achieved until the last decade. With the advancement of computational systems and deep learning models, significant results are achieved. In this thesis, we introduce novel models using the structural recurrent neural network (SRNN) that capture the spatial proximity and structural properties such as the existence of faults in regions. Experimental results are carried out in two distinct regions such as Turkey and China, where the scale and earthquake zones differ greatly. SRNN models achieve better performance results compared with the baseline and the state of the art models. Especially $SRNNClass_{near}$ model, which captures first-order spatial neighborhood and structural classification based on fault lines, results in the highest $F_1$ score.

**Keywords:** Spatio-Temporal Prediction, Artifical Neural Networks, Earthquake Prediction, Structural Recurrent Neural Networks

## *ACKNOWLEDGEMENTS*

Firstly, I would like to express my sincere gratitude to my advisor, Asst. Prof. Dr. Engin DEMİR for all his support, guidance, and mentorship all stages of my thesis work period.

I would also like to thank the members of the thesis committee for their insightful comments about this thesis.

Finally, for all of her encouragement and patience, I thank my dear wife. Thank you to my beloved daughter for her beautiful presence among us. I would like to thank my sincere feelings to my family who supported me during my education life.

Aydın DOĞAN is supported by the TUBITAK 2210-A National Scholarship Program for MSc Students.

The experiments reported in this thesis were partially conducted at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

Aydın DOĞAN
January 2021, Ankara

# CONTENTS

# TABLES

# FIGURES

# 1. INTRODUCTION

## 1.1. Overview

Earthquakes are natural disasters. They can occur suddenly anywhere in the world and cause serious damage depending on their magnitude. This damage ranges from the destruction of buildings to the loss of people's lives. Earthquake prediction is a challenging task. It is considered that the internal dynamics of earthquakes are too complex to be understood. Not much progress has been made in scientific studies for long time [1]. However, with the development of neural networks, it has made great progress in learning to catch deep internal dynamics of complex data from many different sources [2]. Especially if the data is non-stationary and existing linear models are limited to model such complex data as in earthquake prediction problem, deep learning approaches might enlighten us to achieve accurate predictions.

Time window, space window, and magnitude window are suggested as a first three attributes for earthquake prediction [3]. The time and space attributes form the earthquake prediction as a spatio-temporal problem. Throughout the thesis, the earthquake prediction problem is defined as given a minimum Richter magnitude scale and a specified geographic region, predicting the possibility of an earthquake in that region within a time interval i.e., each month.

## 1.2. Motivation

Based on the non-stationary characteristic of earthquake prediction, artificial neural networks capturing structural spatio-temporal models may result in a better model to learn the latent solution space. Structural Recurrent Neural Network (SRNN) [4] model achieved successful results in different problems such as in traffic speed prediction [5], group activity prediction [6], and human trajectory prediction [7]. Motivated by the success of SRNN, it is adapted to learn spatio-temporal and structural relations using the earthquake catalog data including the Richter magnitude scale, timestamp, and geographic location.

## 1.3. Major Contributions of the Thesis

The objective of this thesis is to show that via modeling earthquake data with the SRNN, the structural and spatio-temporal relationships within the data can be discovered and accurate predictions are possible. The performance results of our SRNN models are compared with the SMA, recently proposed simple neural network One Neuron [8] and LSTM [9] models. Performance results of these distinct models are compared with the evaluation measures of $F_1$ score, precision, and recall since the numbers of positive and negative instances are imbalanced.

The contributions are three fold:

- Structural recurrent neural network models are introduced to capture spatio-temporal and structural characteristics of the earthquakes.
- The structural recurrent neural network architecture results in a robust model compared with the state-of-the-art LSTM model.
- Extensive experiments are carried out on two distinct geographic regions that have different characteristics due to the scale and zones.

## 1.4. Thesis Outline

The thesis is structured as follows:

**Chapter 2** describes background information about spatio-temporal prediction of earthquakes. Basic information about earthquakes is briefly mentioned. Then, Recurrent Neural Networks (RNN) and LSTM are discussed.

**Chapter 3** provides an overview of related studies. It is examined in two categories in terms of data and prediction models.

**Chapter 4** presents the problem definition and datasets. The formal notation of the problem is provided. Then, Turkey and China datasets are detailed.

**Chapter 5** describes baseline models such as Simple Moving Average (SMA), One Neuron, and Long-Short Term Memory (LSTM) and the proposed Structural Recurrent Neural Networks (SRNN).

**Chapter 6** presents the experiments and results in detail.

**Chapter 7** concludes the thesis and offers possible directions for future studies.

# 2. BACKGROUND

In this chapter, preliminary information is provided for spatio-temporal earthquake prediction. First of all, basic issues related to earthquakes are mentioned. Then, brief information will be given about the RNN and LSTM models that form the basis of the solution models we use for the earthquake prediction problem.

## 2.1. Earthquake

Earthquake can be defined as the vibrations on the Earth's surface caused by a sudden release of energy, and from a broader perspective, it can be defined as a complex long-term stress accumulation and the oscillation process [10]. In many studies [11–13], it has been suggested that earthquakes have chaotic process characteristics.

The Richter scale is developed by Charles F. Richter [14] and shows the strength of the earthquake. The representation of Ricther scale formula is given in Equation (1).

$$M_L = \log_{10}(\frac{A}{A_0(\delta)}) \tag{1}$$

In Equation (1), $M_L$ denotes the magnitude scale of the earthquake, $A$ is maximum deviation and $A_0$ is a function that changes depending on the distance from the epicenter of the earthquake. Since the magnitude calculation formula includes a base 10 logarithm, an increase in the magnitude of the earthquake by 1 unit on the Richter scale means its actual magnitude will increase ten times. Earthquakes larger than 8 in magnitude can not be measured with the Richter scale. There are different magnitude types for elevating earthquake strengths [15] such as $Mb$ (Short-Period Body Wave), $Mwr$ (Regional), $Md$ (Duration). For simplicity, all the magnitudes are called Richter scale in this thesis.

It is clear that the movement and fracture of the ground affect other ground movements. The results of this interaction can be observed in terms of time and space [16, 17]. A sequence of earthquakes can be observed after the first earthquake. The first earthquake is called the main shock, followed by smaller or larger earthquakes called aftershocks.

## 2.2. Recurrent Neural Networks (RNN)

Recurrent neural networks, RNNs, are designed to process sequential data [18]. The outputs of RNNs depend on the previous elements of the sequence. The previous information is passed by the hidden states [19]. This process of RNNs is shown in Figure 2.1..

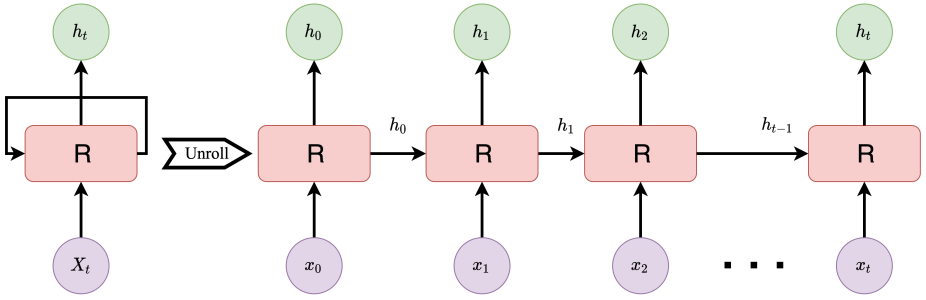


**Figure 2.1.** Recurrent Neural Networks

$$h_t = RNN(x_t, h_{t-1}; W^{RNN}) \tag{2}$$

The RNN functional definition is given in Equation (2) where $W^{RNN}$ is learnable weight matrix, $x_t$ is the input at time $t$ and $h_{t-1}$ is the hidden state of the previous time. The drawback of that architecture is vanishing gradients [20] especially in long time sequences.

## 2.3. Long-Short Term Memory (LSTM)

Learning to store information over long time intervals with the back propagation of RNNs causes the gradient effect to decrease in series for the first time and learning to take longer. LSTM model has proposed in [21] to overcome this issue. The architecture of the LSTM model is seen in the Figure 2.2.[1].

$$f_t = \sigma([h_{t-1}, x_t]W^f + b^f) \tag{3}$$

$$i_t = \sigma([h_{t-1}, x_t]W^i + b^i) \tag{4}$$

[1]Figure is adapted from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`

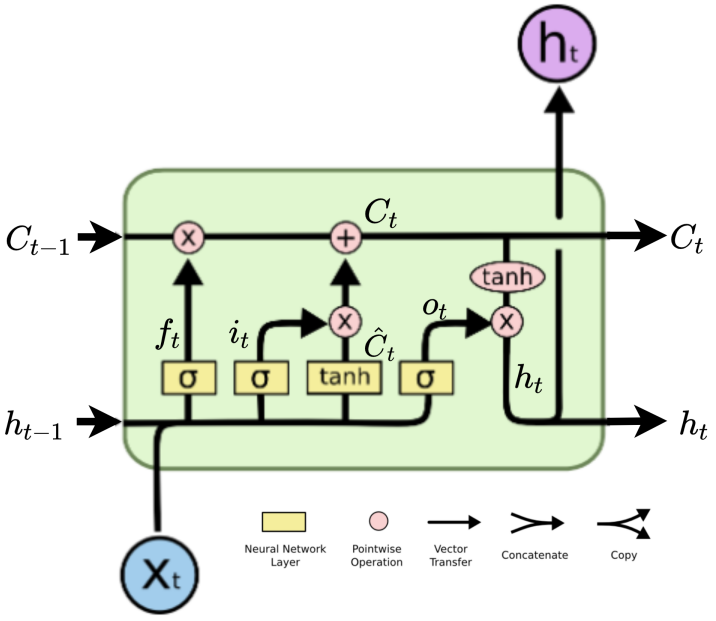


**Figure 2.2.** Long-Short Term Memory

$$\hat{C}_t = tanh([h_{t-1}, x_t]W^C + b^C) \tag{5}$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \tag{6}$$

$$o_t = \sigma([h_{t-1}, x_t]W^{out} + b^{out}) \tag{7}$$

$$h_t = o_t * tanh(C_t) \tag{8}$$

Typical formulations are given in between Equations (3) and (8). In these equations, $[h_{t-1}, x_t]$ refers the concatenation of input at time t, $x_t$ and previous time hidden state, $h_{t-1}$, $b$ is bias, $W$ is learnable weight matrix, $\sigma$ stands for sigmoid function, $tanh$ is hyperbolic tangent function. Input gate, forget gate, output gate, memory cell, candidate values are represented by $i_t$, $f_t$, $o_t$, $C_t$, $\hat{C}_t$, respectively. Sigmoid ($\sigma$) and the hyperbolic tangent ($tanh$) activation functions are used in LSTM formulas. The graph of sigmoid ($\sigma$) is shown in Figure 2.3., and

its formula is shown in Equation 9. The graph of hyperbolic tangent ($tanh$) is shown in in Figure 2.4., and its formula is shown in Equation 10.

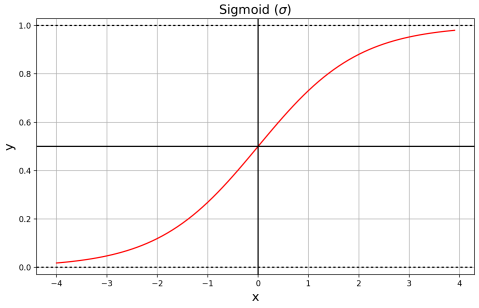


**Figure 2.3.** Sigmoid

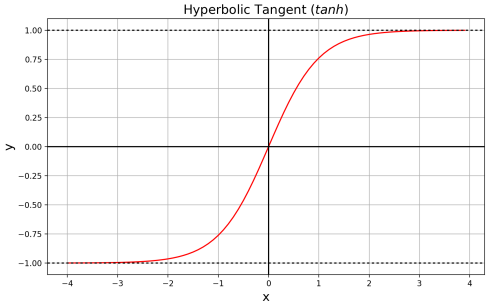


**Figure 2.4.** Hyperbolic tangent

$$sigmoid(x) = \frac{1}{1 + exp^{-x}} \tag{9}$$

$$tanh(x) = \frac{exp^{x} - exp^{-x}}{exp^{x} + exp^{-x}} \tag{10}$$

# 3. RELATED WORK

Earthquake prediction studies differ greatly in terms of both the data and models used. Data types are classified into three categories, such as seismic waves, earthquake catalogs, and potential earthquake precursors. Methods are also classified into two categories such as statistical methods and deep learning models. These are presented in the following sections.

## 3.1. Data

Most of the earlier studies focus on distinct local regions and hence it is not easy to compare the results of the previous studies. Types of data utilized in earlier works can typically be grouped into three categories.

First type of data is the seismic wave data. It is a signal data of earth surface vibration accumulated from one or more stations. Using this type of data, the location, depth, and magnitude of earthquakes can be calculated. It has been used as inputs to the deep learning model [22] for simultaneous earthquake detection and selection of the phases of P and S waves, which are the important signs of earthquake time and magnitude. The Convolutional Neural Network (CNN) model [23] was applied on induced seismicity to improve seismic hazard assessment in Oklahoma, USA, and detected 17 times more than earthquakes previously cataloged by the Oklahoma Geological Survey. In [24], RNN model has been adapted for performing the seismic phase relationship.

Second type of data is the earthquake catalog in tabular format. Time, magnitude and location data are mainly used in these tabular catalogs. In 2007 [25] and 2009 [26], Panakkat and Adeli carried out comprehensive studies using tabular catalogs. In [25], Levenberg-Marquart backpropagation (LMBP) neural network, an RNN model, and a Radial Basis Function (RBF) was proposed to predict magnitude of the largest seismic activity in the following month in Southern California and San Francisco. It has been reported that the RNN model achieves better results than LMBP and RBF. In [26], RNN model has been adopted to predict the time, location, and magnitude of the future earthquakes via using 8 seismic indicators which are derived from the historical earthquake data. Experiments conducted on equally divided 6 rectangular subregions in the Southern California area show that subregions with high earthquake intensity achieve better results than lower-intensity subregions.

In [27], a CNN model was proposed to predict whether an earthquake with a magnitude larger than six would occur in Taiwan. Taiwan catalog data was converted into a 256x256 pixel binary map with a 30-day time window. It is stated that the data for the last 120 days obtained the best predictive results to estimate the next 30 days. Another recent study [9] divided the China region into 9 equal rectangles and introduced an LSTM model that predicts earthquakes within one-month time intervals. It is emphasized that, for the LSTM model [9], the two-dimensional input had better results in earthquake prediction in terms of true positive accuracy and true negative accuracy compared with a one-dimensional input representation. Spatial and temporal relationships are generally investigated with different architectures and then their interactions are combined to achieve spatio-temporal integrity [28]. However, in the proposed LSTM model, the data of the regions is fed to the model without any explicit relation among the regions. Expecting to discover the interaction between regions is not likely without feeding the structural interconnection relations of these regions to the model. In [29], LSTM is used to determine time series dependencies in 10x10 square kilometer areas in the Japan region, and CNN is used to capture spatial dependencies. The prediction results regarding whether there will be an earthquake with magnitude larger than 3 and larger than 5 in the next 10 to 60 days were produced by processing the tabular catalogs. It is suggested that the combination of CNN and LSTM model achieved better results on the basis of both the Precision-Recall Area Under Curve (PR AUC) and the Receiver Operating Characteristics Area Under Curve (ROC AUC) scores when compared with the general gradient boosting approach and the simple historical mean.

The third data type is a compilation of different sources apart from seismic waves or earthquake catalogs. This type of data can be referred as potential earthquake precursors. Electromagnetic signals that are considered to be earthquake precursor signals [30, 31], cloud images before earthquakes [32, 33], and animal abnormal behavior [34] have been studied to increase the sensitivity in earthquake prediction.

The type of data utilized in the problem formulation is an important aspect as new features and latent relations can be discovered. On the other hand, widely accessible data might provide some latent insights if appropriate models are utilized. The data is typically captured as temporal or spatio-temporal data, hence methods in related work include time series analysis and spatio-temporal analysis.

### 3.2. Methods

The methods used in earthquake prediction can be divided into two categories such as classical methods and deep learning models. The first one is classical methods consisting of statistical and rule-based methods. Long Term Average (LTA) / Short Term Average (STA) algorithms [35–37], and heuristic pattern matching methods [38, 39] from previous earthquake waveforms are among the classical methods. While heuristic pattern matching methods that are applied to detect similar patterns in the past earthquakes achieve good sensitivity scores, these models are prone to noise and take a lot of time with large-scale data [40].

The second collection of methods composed of machine learning and deep learning based models. Machine learning models such as K-nearest neighborhoods (kNNs), and Support Vector Machines (SVM) [41, 42] have been used to explore time series relations in earthquake prediction. In the recent years, different neural networks such as Long-Short Term Memory (LSTM) [9], Convolutional Neural Network (CNN) [29], and Deep Neural Networks (DNN) [43] have been proposed to overcome the limitations of classical models. While deep learning networks achieve better results than simple neural networks in many fields, on the contrary, it has been suggested that simple neural networks can be as successful as deep networks in predicting aftershocks in earthquake prediction [44]. The ROC value obtained via the deep network [43] has also been obtained using only one neuron in [8].

Research on spatio-temporal problems using neural networks increased in volume significantly in recent years [45]. We also regard spatio-temporal deep learning models as key solution candidates for solving earthquake prediction. Therefore, it makes sense to briefly examine the deep learning models suggested in different spatio-temporal problem solutions. To solve this kind of spatio-temporal problems, different neural network architectures are proposed, such as Restricted Boltzmann Machines (RBM), RNN, LSTM, CNN, and Structural Recurrent Neural Networks (SRNN). RBM models [46] are conducted for traffic prediction with taxi gps trace in [47]. CNN architecture is adopted different type of problem sets such as future location prediction in trajectories [48, 49], and traffic congestion prediction [50]. The combination of CNN and LSTM is proposed for traffic accident prediction [51] and precipitation nowcasting [52]. SRNN is proposed for human activity prediction [4], traffic speed prediction [5] and group activity prediction. Spatio-Temporal Graph Neural Network is proposed for Covid-19 forecasting [53], and Spatio-Temporal Graph Convolution Network is used for traffic forecasting in [54]. Although the use of deep learning in many

spatial-temporal problems has increased, there are a few earthquake prediction studies using spatio-temporal deep learning [9, 29].

# 4. PROBLEM DEFINITION AND DATASET

This chapter presents formal definition of the problem. It also provides information about Turkey and China datasets.

## 4.1. Problem Definition

The historical data is composed of time windows $T = \{t_1, t_2, ..., t_L\}$ where each time window $t_i$ refers to a time interval of one month. The region of interest is defined in terms of a range of latitude and longitude values. This region of interest is partitioned into rectangular subregions where the range of latitude and longitude values are divided by $J$ and $K$, respectively. $x_t = \{x_t^1, x_t^2, ..., x_t^N\}$ denotes the data of earthquakes in all subregions $N = J \times K$ within time window $t$. For a specific region $i$, at time $t$, $x_t^i \in \{0, 1\}$ is a binary data such that if there is an earthquake above a given threshold magnitude then it is set to 1, otherwise 0. For the $L$ sequence of time windows, earthquake data in all subregions is represented as $X = \{x_1, x_2, ..., x_L\}$, $X \in \mathbb{R}^{L \times J \times K}$.

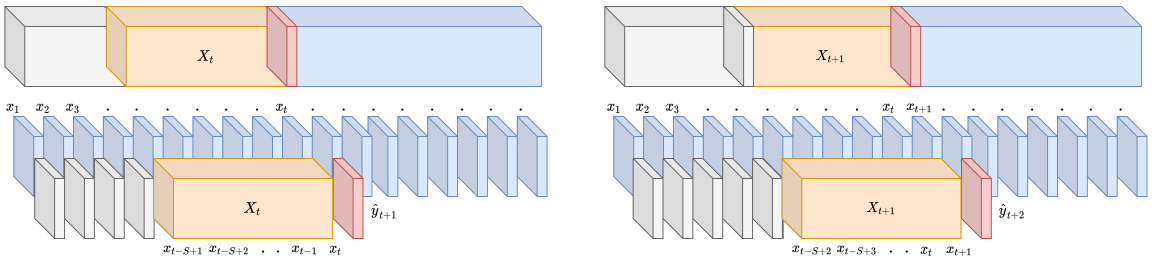


**Figure 4.1.** Sliding windows for prediction

During the prediction, the last $S$ time windows are taken into account. For a given time window $t$, $X_t = \{x_{t-S+1}, x_{t-S+2}, ..., x_{t-1}, x_t\}$ data to be used with a sequence of length $S$. Sliding windows are used to predict all consecutive time windows. For example, at time $t + 1$, input data is constructed as $X_{t+1} = \{x_{t-S+2}, x_{t-S+3}, ..., x_t, x_{t+1}\}$. Therefore, prediction at $t + 1$ is $\hat{y}_{t+1} = \{x_{t+1}^1, x_{t+1}^2, ..., x_{t+1}^N\}$ . This progress is displayed in Figure 4.1.. The problem is defined as predicting earthquakes at the time $t + 1$, $\hat{y}_{t+1}$, by feeding the historical earthquakes data $X_t$ to the prediction model as shown in Equation (11).

$$\hat{y}_{t+1} = f(X_t) \tag{11}$$

## 4.2. Dataset

Earthquake catalog data is collected from USGS [1]. In this work, the geographic regions of Turkey and China are utilized due to their different characteristics and zones. The datasets include tabular data such as time, magnitude, and location information. The location information includes latitude, longitude, respectively, according to the geographic coordinate system.

All the interested area $A$ is divided into subregions with two different approaches. First one is regular grid which is basically dividing all region into equal $(J \times K)$ subregions. Second one is a quadtree which divides the total area with respect to the density of earthquakes within a subregion. These approaches are detailed in the Section 6.1.3..

### 4.2.1. Turkey

Turkey dataset is the one used in our work [55]. Earthquake catalog data was collected between 1970 and 2019. Earthquakes with a magnitude 4 and above are selected. Detailed information about the data is presented in Table 4.1.. The geographic distribution of earthquakes together with the scale of magnitudes is shown in Figure 4.2.. The Richter scale histogram is shown in Figure 4.3.. 5425 of the 5482 earthquakes occurred between 4 and 5 magnitudes.

**Table 4.1.** The details of the data in Turkey

| Parameter | TURKEY |
|---|---|
| Minimum Latitude | 34 |
| Maximum Latitude | 43 |
| Minimum Longitude | 25 |
| Maximum Longitude | 46 |
| Magnitude Threshold | 4 |
| Magnitude Mean | 4.39 |
| Magnitude Standard Deviation | 0.41 |
| Start time | 1970-03-28 |
| End time | 2019-12-29 |
| Number of Earthquakes | 5482 |

[1] https://earthquake.usgs.gov/

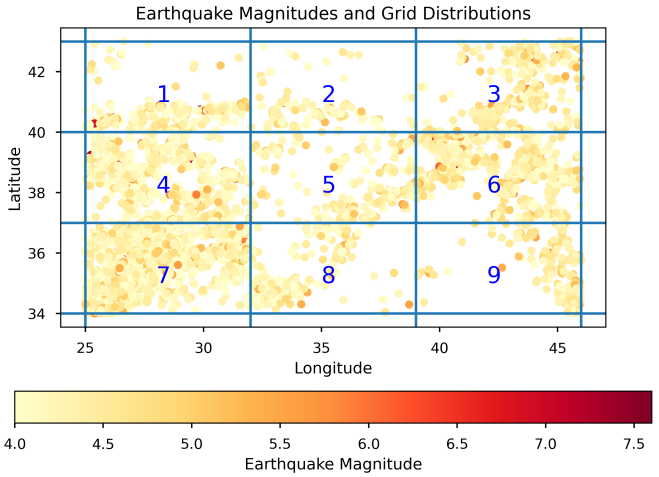


**Figure 4.2.** Geographic distributions of earthquakes with their scale of magnitude in 9 subregions of Turkey

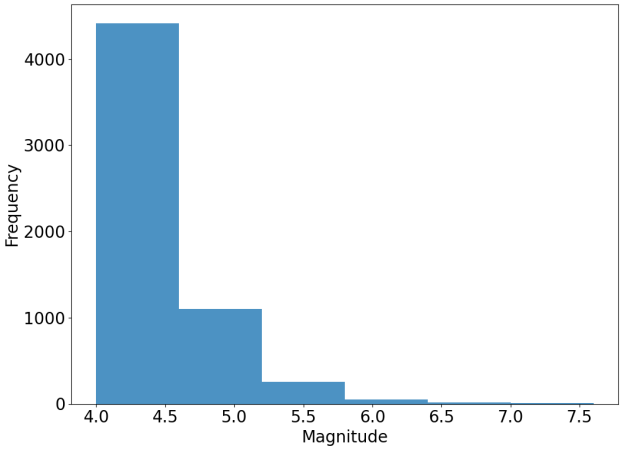


**Figure 4.3.** The histogram of earthquake magnitudes in Turkey

### 4.2.2. China

China dataset is collected as in the study of Wang et al. [9]. Details of the dataset for China are provided in Table 4.2.. Earthquakes larger than 4.5 Richter scale are selected. The histogram of earthquakes are shown in Figure 4.5..

**Table 4.2.** The details of the data in China

| Parameter | CHINA |
|---|---|
| Minimum Latitude | 23 |
| Maximum Latitude | 45 |
| Minimum Longitude | 75 |
| Maximum Longitude | 120 |
| Magnitude Threshold | 4.5 |
| Magnitude Mean | 4.859 |
| Magnitude Standard Deviation | 0.396 |
| Start time | 1966-12-16 |
| End time | 2016-09-28 |
| Number of Earthquakes | 5577 |

The earthquake density over 9 subregions in China is shown in Figure 4.4.. While subregions 3, 6, 9 have low intensity earthquakes, other subregions contain relatively high intensity earthquakes. In addition, the earthquakes with the highest magnitudes are seen in subregion 7 and subregion 8.

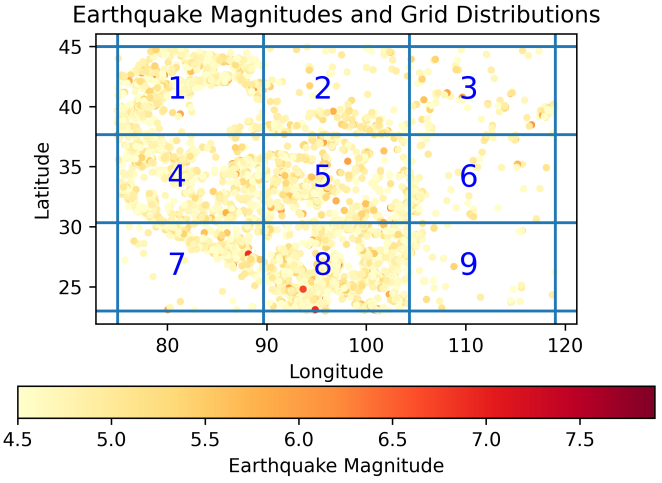


**Figure 4.4.** Geographic distributions of earthquakes with their scale of magnitude in 9 subregions of China

Turkey and China greatly differ from each other in terms of location, size, plates, and fault lines. Turkey is located in a complex collision between the Eurasian Plate and both the African and Arabian Plates. Turkey has two major strike-slip fault zones. These are the North Anatolian Fault and East Anatolian Fault [56]. China is located in a unique tectonic environment where the Paleo-Asian Ocean, Tethys and Western Pacific areas converge. China's

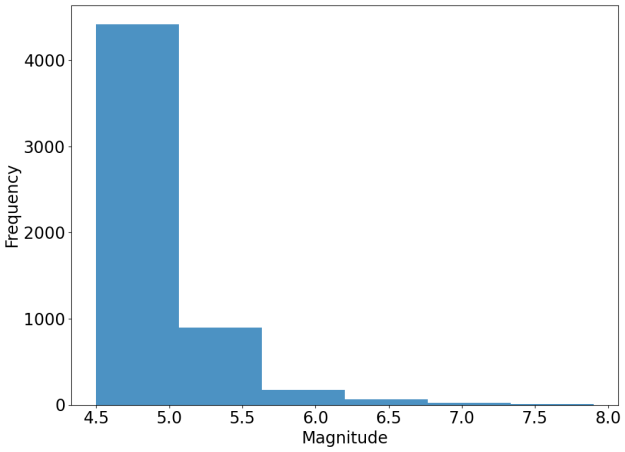


**Figure 4.5.** The histogram of earthquake magnitudes in China

tectonic framework can be divided into eastern and western regions according to the assembly of cratonic blocks [57].

# 5. MODELS

In this chapter, three distinct baseline models namely Simple Moving Average (SMA), One Neuron, and LSTM are compared with the proposed Structural Recurrent Neural Networks (SRNN) model.

## 5.1. Simple Moving Average (SMA)

In SMA model, the data is considered only as time series and the spatial relationship is not taken into account. The simple average operation is applied as shown in (12). $S$ is the length of windows to be averaged and set to 12 in the experiments.

$$\hat{y}_{t+1} = \frac{x_{t-S+1} + x_{t-S+2} + ...x_{t-1} + x_t}{S} \tag{12}$$

## 5.2. One Neuron

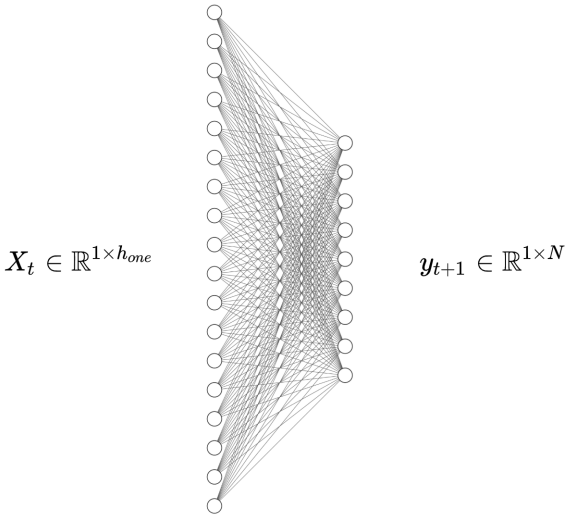


**Figure 5.1.** One Neuron model

Inspired by the work of Mignan and Broccardo [8], one neuron model is adapted. In this basic architecture, data $X_t$ has reshaped as a flatten form before forwarded to neuron, $X_t \in \mathbb{R}^{1\times h_{one}}$, $h_{one} = S \times N$. This is basically shown in (13) where $W^{one} \in \mathbb{R}^{h_{one}\times N}$, $b \in \mathbb{R}^N$ and

$\hat{y}_{t+1} \in \mathbb{R}^{1\times N}$. The architecture shown in Figure 5.1.[1] indicates that there is only one neuron for every region.

$$\hat{y}_{t+1} = X_t W^{one} + b \tag{13}$$

## 5.3. LSTM

An LSTM model is proposed to predict earthquakes from catalog data [9]. The proposed model is shown in Figure 5.2.. The input of the LSTM is $X_t \in \mathbb{R}^{S\times N}$ and the output of the model is $\hat{y}_{t+1} \in \mathbb{R}^{1\times N}$ which is composed of all node prediction scores for time $t+1$. The sequence length $S$ is set to 12 to consider only the last 12 months.

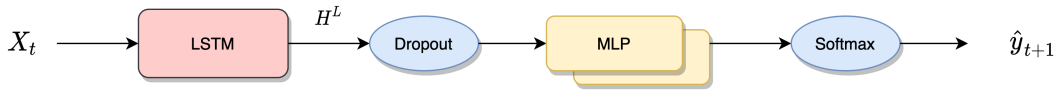


**Figure 5.2.** LSTM model

$$H_t^L = LSTM(X_t, H_{t-1}^L; W^L) \tag{14}$$

$$\hat{y}_{t+1} = Softmax(MLP_2(MLP_1(Dropout(H_t^L); W^{M1}); W^{M2}) \tag{15}$$

In Equation (14), the input data $X_t \in \mathbb{R}^{S\times N}$ is forwarded to the LSTM model in order. In Equation (15), the output of LSTM layer is processed by two fully connected layers sequentially. In this model, all the spatial and temporal relations are expected to be revealed by just a flattened form of nodes data with time sequences. The model parameters such as LSTM, $MLP_1$ and $MLP_2$ hidden layer sizes are set to 128, 256, and 64, respectively, as in [9].

## 5.4. SRNN

So far, SRNN model has been implemented in different domains to reveal spatio-temporal relations, such as video motion detection [4], traffic speed prediction [5], and crowd flow

[1] plot created with `https://alexlenail.me/NN-SVG/`

prediction [6]. The success of the spatio-temporal detection capability of SRNN motivated us to adapt this model in the earthquake prediction problem. In order to discretize the location information, we divide geographic region into regular grids. Our intention is to effectively expose intra and inter spatio-temporal relations of earthquakes in these grids.

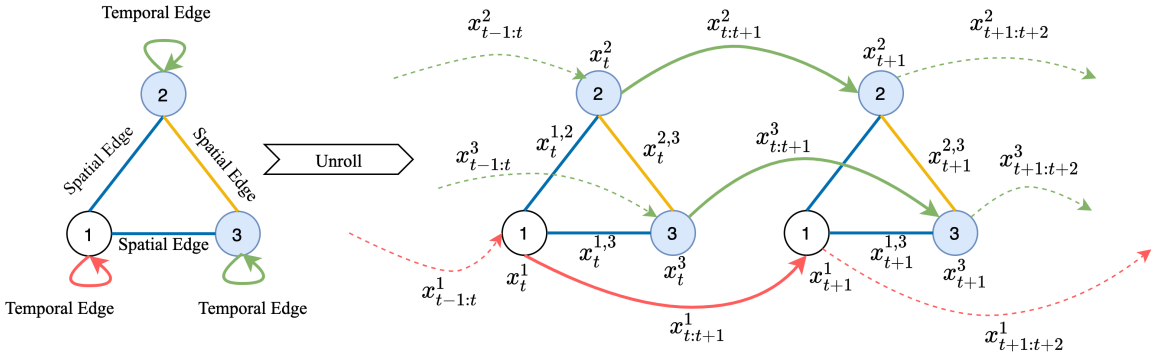


**Figure 5.3.** Spatio-temporal graph with 3 nodes and its expansion through time

SRNN model [4] utilizes a spatio-temporal graph to represent spatial and temporal relations and encode it into a factor graph to transform into a recurrent neural network architecture. Nodes in the spatio-temporal graph denotes the controlled variables such as human body component (head, arms, legs) in human motion prediction, street middle points in traffic speed prediction, and humans in crowd flow prediction. Similarly nodes represent the grid regions in the earthquake prediction. An example illustration of a spatio-temporal graph with 3 nodes and all existing spatial and temporal relations is shown in Figure 5.3.. These 3 nodes represent the subregions in the Figure 4.4. with corresponding region numbers. In terms of simplicity, only subregions numbered $1, 2, 3$ are taken into consideration and these subregions have spatially related with each other. This configuration is the first type of spatial relation, called as $all$, and assumes that all subregions are spatially related with each other, even if there is no adjoining between them. For example, there is no adjoining parts between $Node_1$ and $Node_3$, but spatial edge between $Node_1$ and $Node_3$ is set due to the assumption of all nodes having spatially related. On the other hand, the second type of spatial relation is called as $near$ which means that only neighboring nodes have spatial relations with each other. According to the $near$ configuration, the spatial edge between $Node_1$ and $Node_3$ in Figure 5.3. must be disconnected. Because, as seen in the Figure 4.4., there is no adjoining part between $subregion\ 1$ and $subregion\ 3$.

SRNN model supports the flexibility of defining classes on behalf of common features of nodes. While body parts, head, arm, and leg are the class types of human motion prediction due to factor sharing [4], in earthquake prediction problem, classes can be created with respect to the common characteristics of regional structures such as fault lines. Figure 5.3. shows basically an example where regions belonging to two different node classes $C = \{c_1, c_2\}$ are represented by different colors such as $x^1 \in c_1$ is white and $\{x^2, x^3\} \in c_2$ are blue. As a result of assigning nodes to different classes, new featured connections of spatial and temporal edges are formed. The spatial edge between $x^1$ and $x^2$ is colored with blue line which denotes member of $ce_1^{spatial}$, also same for between $x^1$ and $x^3$. The edge between $x^2$ and $x^3$ is colored with orange which denotes member of $ce_2^{spatial}$. Thus, we have two different spatial edge classes $C_e^{spatial} = \{ce_1^{spatial}, ce_2^{spatial}\}$. In the temporal edge view, the temporal edge of $x^1$ is represented with red line $ce_1^{temporal}$, temporal edges of $x^2$ and temporal edges of $x^3$ are represented by the green line which denotes the member of $ce_2^{temporal}$. Thus, we have two different temporal edge classes $C_e^{temporal} = \{ce_1^{temporal}, ce_2^{temporal}\}$.

In Figure 5.3., expansion of the spatio-temporal graph through time is also displayed. Node features $x_i^t$ represent the information of node at time $t$. Temporal edges represent the temporal relation between consecutive time slots for a given node. For example, $x_{t:t+1}^2$ is the temporal edge which captures the temporal relation of node 2 at time $t$ and $t+1$. Spatial edges represent the spatial relation between two distinct nodes at a specified time slot. For example, $x_t^{1,2}$ is the spatial edge which captures the spatial relation between node 1 and node 2 at time $t$. In addition to temporal and spatial relations, there is also state information which consist of all node features at a given time $t$, $\{x_t^1, x_t^2, ..., x_t^N\}$.

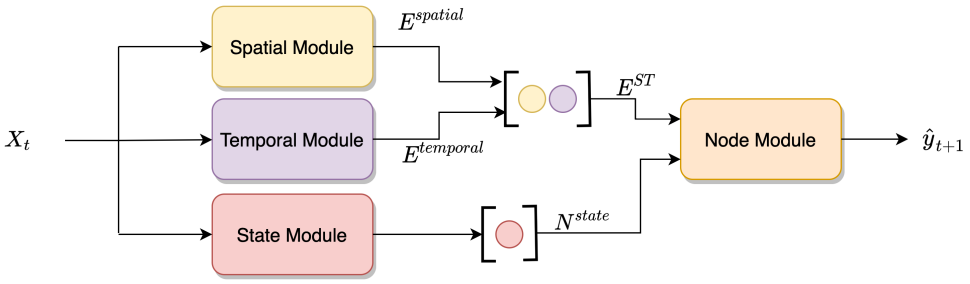


**Figure 5.4.** Structural Recurrent Neural Network Model (SRNN)

The corresponding SRNN architecture including the temporal, spatial, state and node modules is shown Figure 5.4.. The model output generates a probability score for each subregion at a given time slot. In the training, the probability score is compared with the ground truth

binary data of $\{1, 0\}$, and the calculated Mean Squared Error (MSE) loss value is backpropagate through modules. Thus, the weights of the modules are updated to reveal spatial and temporal relations. During testing, the probability scores are converted into binary values according to the threshold. For simplicity in the following subsections, the modules are described for a single class type and Section 5.4.5. discusses the extension of modules in the case of having more than one class.

### 5.4.1. Spatial Module

Spatial module derives the spatial relation of the nodes. The module utilizes the neighborhood information of nodes such as adjacency matrix. In the adjacency matrix, if $A^{1,2}$ has a value of $1$ then $Node_1$ and node $Node_2$ are spatially related, otherwise they are not spatially related. A sample adjacency matrix $A \in \mathbb{R}^{N \times N}$ is shown in (16) according to the $all$ configuration, all nodes are spatially related but not self related, hence all values except the diagonal values are set to 1. On the other hand, $near$ configuration, only neighboring nodes have spatial edges with each other, is shown in (17) with respect to the Figure 4.4.. In this example, the value of $A^{1,3}$ is $0$ on behalf of $Node_1$ and $Node_3$ has no neighbouring relation.

$$A = \begin{pmatrix} A^{1,1}=0 & A^{1,2}=1 & A^{1,3}=1 & A^{1,4}=1 & \cdots & A^{1,N}=1 \\ A^{2,1}=1 & A^{2,2}=0 & A^{2,3}=1 & A^{2,4}=1 & \cdots & A^{2,N}=1 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{N,1}=1 & A^{N,2}=1 & A^{N,3}=1 & A^{N,4}=1 & \cdots & A^{N,N}=0 \end{pmatrix} \tag{16}$$

$$A = \begin{pmatrix} A^{1,1}=0 & A^{1,2}=1 & A^{1,3}=0 & A^{1,4}=1 & \cdots & A^{1,9}=0 \\ A^{2,1}=1 & A^{2,2}=0 & A^{2,3}=1 & A^{2,4}=1 & \cdots & A^{2,9}=0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{9,1}=0 & A^{9,2}=0 & A^{9,3}=0 & A^{9,4}=0 & \cdots & A^{9,9}=0 \end{pmatrix} \tag{17}$$

Spatial module is composed from the spatial embedding layer, ReLU, dropout, LSTM layer, separation of nodes and sum function. All steps are shown in Figure 5.5..

$$SE = \phi(x_t^{spatial}; W^{SE}) \tag{18}$$

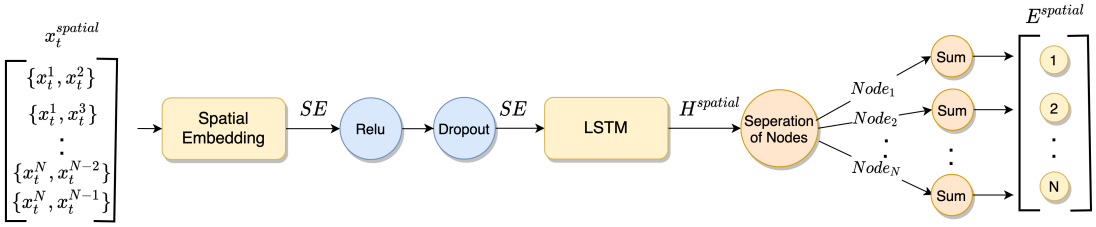


**Figure 5.5.** Spatial module

In Equation (18), the linear transformation function $\phi$, also called as spatial embedding is shown where $x_t^{spatial} \in \mathbb{R}^{ns \times 2}$ is spatial pair input matrix and $W^{SE} \in \mathbb{R}^{2 \times h_{SE}}$ is the learnable weight matrix, $h_{SE}$ is the hidden dimension of the spatial embedding and $SE \in \mathbb{R}^{ns \times h_{SE}}$ .

$$H_t^{spatial} = LSTM(Dropout(ReLU(SE)), H_{t-1}^{spatial}; W_{spatial}^{lstm}) \tag{19}$$

The output of spatial embedding layer is processed by ReLU function and dropout layer as shown in Equation (19), sequentially. In LSTM, $W_{spatial}^{lstm} \in \mathbb{R}^{h_{SE} \times h_s}$ and $h_s$ is the hidden dimension of the LSTM, $H_{t-1}$ is the hidden state at previous time, $t-1$. The output of the LSTM $H_t^{spatial} \in \mathbb{R}^{ns \times h_s}$ is processed by $nodeSeperation$ function and $sum$ operation on each node separately.

$$E^{spatial} = sum(f_{nodeSeperation}(H_t^{spatial})) \tag{20}$$

In Equation (20), $nodeSeperation$ is the function that splits the output of the LSTM, $ns \times h_s$ into $n$ chunks, each called $Node_n$. $Node_n$ gathers the spatial edges of $n$th node with respect to the $x_t^{spatial}$. As a result of gathering $n$th node chunk shape becomes $ns_n \times h_s$; where $ns_n$ is the total number of spatial edge pairs of the respective node. For example, if all nodes $(N = 9)$ have spatial edges, there are 8 spatial edge pairs for each node, in other words, 9 chunks with $8 \times h_s$ size at the sum operation which is responsible for dimension fixing. In the $sum$ operation, spatial chunk is summed $\mathbb{R}^{8 \times h_s} \rightarrow \mathbb{R}^{1 \times h_s}$ for each node and concatenation of all $Node_n$ results in $\mathbb{R}^{N \times h_s}$. Thus, the output of spatial module becomes $E^{spatial} \in \mathbb{R}^{N \times h_s}$.

### 5.4.2. Temporal Module

Temporal module consists of temporal embedding, ReLU, dropout, and LSTM layer as shown in Figure 5.6.. It gets the input $x_t^{temporal}$ which consists of the binary data of nodes at time $t$ and $t-1$. Thus, $x_t^{temporal}$ data shape becomes $N \times 2$.

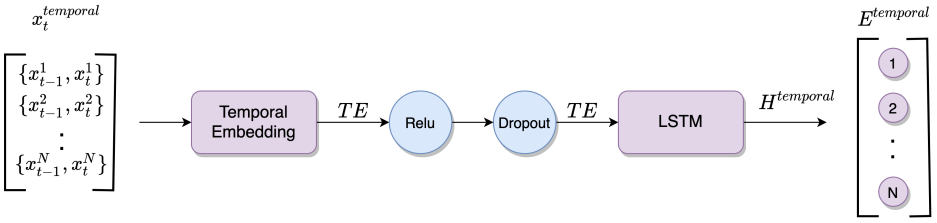


**Figure 5.6.** Temporal module

$$TE = \phi(x_t^{temporal}; W^{TE}) \tag{21}$$

In Equation (21), temporal embedding is a linear transformation of temporal data, where $W^{TE} \in \mathbb{R}^{2 \times h_{TE}}$ is the learnable weight matrix and $h_{TE}$ is the size of temporal embedding hidden layer size. Temporal embedding output becomes $TE \in \mathbb{R}^{N \times h_{TE}}$.

$$H_t^{temporal} = LSTM(Dropout(ReLU(TE)), H_{t-1}^{temporal}; W_{lstm}^{temporal}) \tag{22}$$

In Equation (22), the output of the temporal embedding $TE$ is processed by a ReLU and a dropout. Temporal LSTM layer uses processed $TE$ as input and it produces a temporal feature map for all nodes where $H_{t-1}^{temporal}$ is the hidden states at time $t-1$ of LSTM, $W_{lstm}^{temporal} \in \mathbb{R}^{h_{TE} \times h_t}$ learnable weight matrix of LSTM, and $h_t$ is the size of LSTM hidden layer. LSTM output becomes $H^{temporal} \in \mathbb{R}^{N \times h_t}$ which is also called as $E^{temporal}$ on behalf of the compact notation of spatial module. Thus, the output of the temporal module becomes $E^{temporal} \in \mathbb{R}^{N \times h_t}$.

### 5.4.3. State Module

State module is shown in Figure 5.7.. The input of the state module is composed of binary data of all nodes at time $t$, $x_t^{state} \in \mathbb{R}^{N \times 1}$.

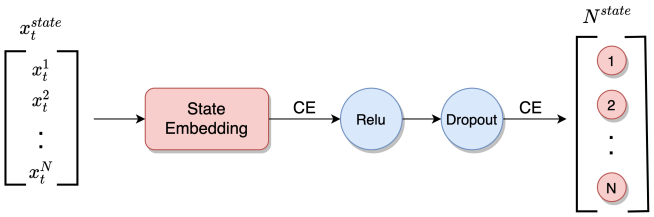


**Figure 5.7.** State module

$$CE = \phi(x_t^{state}; W^{CE}) \tag{23}$$

$$N^{state} = Dropout(ReLU(CE)) \tag{24}$$

Input data, $x_t^{state}$, is processed by the state embedding. In Equation (23), state embedding is a linear transformation of state $x_t^{state}$ where $W^{CE} \in \mathbb{R}^{1 \times h_{CE}}$ is the learnable weight matrix and $h_{CE}$ is the size of the state embedding. State embedding output becomes $CE \in \mathbb{R}^{N \times h_{CE}}$. Then, in Equation (24), $CE$ is forwarded to a ReLU and a dropout sequentially and results in $N^{state} \in \mathbb{R}^{N \times h_{CE}}$.

### 5.4.4. Node Module

The node module is the last phase of the SRNN model. Spatial, temporal, and state module outputs are the input of node module. This module contains embedding layers, ReLU, dropout, and LSTM layer as shown in Figure 5.8..

$$E^{ST} = concat(E^{spatial}, E^{temporal}) \tag{25}$$

$$STE = Dropout(ReLU(\phi(E^{ST}; W^{ST}))) \tag{26}$$

In Equation (25), the output of the spatial and temporal modules are concatenated $E^{spatial}$ and $E^{temporal}$, resulting in $E^{ST} \in \mathbb{R}^{N \times (h_s + h_t)}$. Then, in (26), $E^{ST}$, is forwarded by a linear transformation where $W^{ST} \in \mathbb{R}^{h_{st} \times h_{STE}}$, $h_{st} = h_s + h_t$ and $h_{STE}$ is the hidden size of the

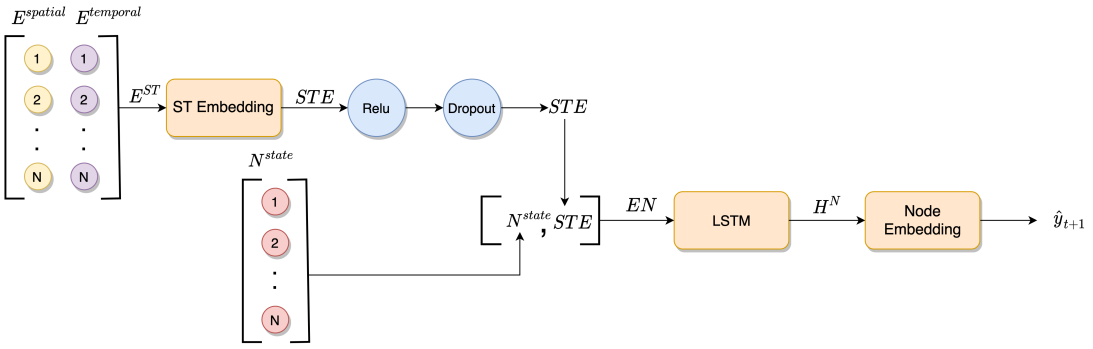


**Figure 5.8.** Node module

spatio-temporal embedding. The output of spatio-temporal embedding $STE$ is forwarded to a ReLU and a dropout sequentially as stated in (26) and results in $STE \in \mathbb{R}^{N \times h_{st}}$.

$$EN = concat((N^{state}, STE)) \tag{27}$$

$$H_t^{node} = LSTM(EN, H_{t-1}^{node}; W^{node}) \tag{28}$$

In Equation (27), the spatio-temporal embedding $STE$ and state of nodes $N^{state}$ at time $t$ are concatenated to be used as the input to the node LSTM, $EN \in \mathbb{R}^{N \times (h_{CE}+h_{st})}$. In Equation (28), $EN$ is forwarded to the node LSTM layer resulting in $H_t^{node} \in \mathbb{R}^{N \times h_{node}}$ where $h_{node}$ is the hidden size of the node LSTM layer.

$$\hat{y}_{t+1} = \phi(H_t^{node}; W^{NE}) \tag{29}$$

In Equation (29), the output of LSTM, $H_t^{node}$, is forwarded to node embedding layer where $W^{NE} \in \mathbb{R}^{h_{node} \times 1}$. The output of node embedding becomes $\hat{y}_{t+1} \in \mathbb{R}^{N \times 1}$. Thus, at the last step, node module produces the prediction scores for all nodes.

### 5.4.5. Multi Class SRNN Models

In the one class model, there is only one instance of each of the spatial, temporal, state and node modules in the architecture. However, in the multi-class model, spatial, temporal, state, and node modules are created separately for each class. Accordingly, forward operations are conducted with the respective class type module for each node and edge. Figure 5.9. shows the architecture that models the spatio-temporal graph in Figure 5.3. having 2 distinct node classes. The types of these node classes are determined by prior knowledge before training. This prior knowledge has been obtained from the tectonic plate features of the subregions. In Figure 5.3., it is assumed that $Node_2$ and $Node_3$ share the same tectonic features, so they are member of the same class while $Node_1$ is the member of the other class. Node classes create state modules and node modules that match white and blue colors. Temporal module and spatial modules are also visualized in Figure 5.9. by matching the colors of the spatial and temporal edges in Figure 5.3.. These colors are red and green for temporal edges, blue and yellow for spatial edges. Outputs of spatial and temporal modules for each class are appended and then concatenated to construct $E^{ST}$. Similarly, the output of state modules for each class is appended and finally the node modules for each class are processed to compute the result.

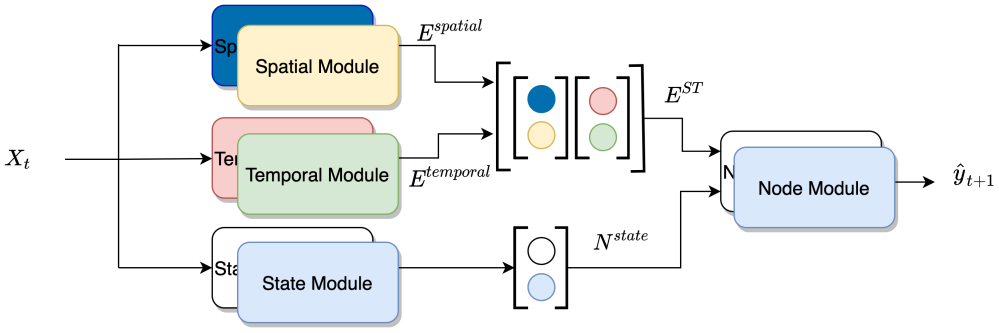


**Figure 5.9.** SRNN model for the multi class example given in Fig. 5.3.

# 6. EXPERIMENTS AND RESULTS

In this section, firstly, the details of the experimental setup are given such as threshold selection, model parameters, partitioning methods for the geographic regions. Then, the experiment results are discussed.

## 6.1. Experimental Setup

Data is divided into train, validation, and test as 75, 5, and 20 percent of the data, respectively. We tested the SRNN model architecture based on the following adaptations:

- **Spatial Relation:** Two distinct models are proposed with respect to the spatial locality. The first type is neighboring nodes, ($SRNN_{near}$), which is inspired from the Waldo Tobler's first law geography: "everything is related to everything else, but near things are more related than the distant things" [58]. Neighboring nodes are the nodes that have adjacent parts and have spatial edges with each other in the spatial module. The second, ($SRNN_{all}$), all nodes have spatial edges with each other in the spatial module.
- **Node Class:** If regions have the same tectonic features, then we regard these regions share features with each other. This aspect of classification also decreases model parameters. Firstly, in $SRNN$, we conduct experiments of all nodes have the same features, so there is only one instance of spatial, temporal, and node modules. Secondly, in $SRNNClass$, we construct our node classification based on faults in the grid cells. Figure 6.1. shows the faults and classification of grids in a $3 \times 3$ partitioning. Four classes are investigated with respect to fault zones[1]. Classes are selected from a dominant type of ground movement which is represented with different colors (c1:blue, strike-slip, c2:red, normal, c3:black, reverse/thrust, c4:no fault). Each class has its own representative temporal, spatial, node modules with respect to the prior knowledge of faults.

[1] `https://esdynamics.geo.uni-tuebingen.de/faults/downloads.php`

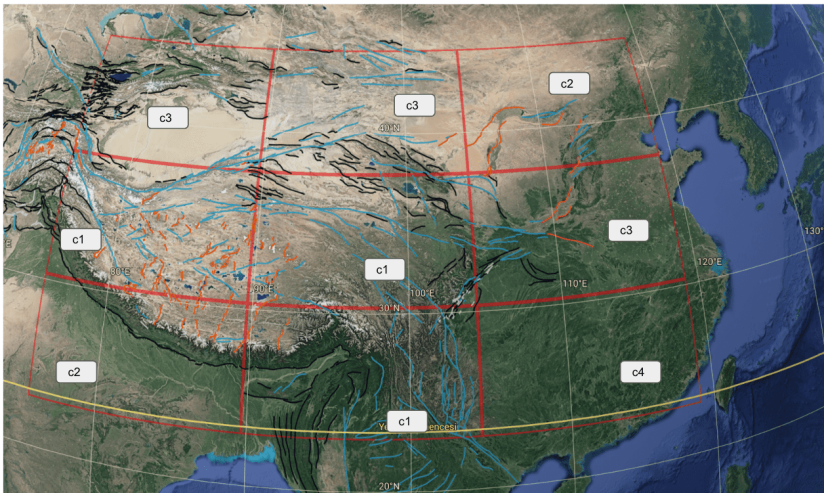


**Figure 6.1.** Classification of regions with respect to fault zones in a 3x3 partitioning

### 6.1.1. Threshold Selection

The scalar value which maximizes the $F_1$ in the validation data is selected as a threshold to decode the output scores into a binary value. This value is fixed among all test instances. We used training data to determine the threshold, however, it did not get any better results compared to using the one computed from the validation data. It is considered that determining the threshold value from the validation set produces better results, because the validation data are more similar to the test data than the training data due to their temporal proximity. Therefore, we only present the results generated by the threshold value determined using the validation set.

On the other hand, we have carried out attempts to determine a separate threshold value for each region. Although this resulted in an increase in the recall value, it caused a decrease in precision values and consequently produced almost the same $F_1$ score.

### 6.1.2. Parameters

Sequence length $S$ is selected as 12 for all models, i.e., 12 months, and the windows are shifted monthly. Early stop, which is a kind of regularization based on choosing when to stop training, is applied in the experiments and its value is determined as 4. The hyperparameters of the SRNN model is shown in Table 6.1.. It has been experimentally observed that the SRNN model performs the best with these hyperparameters.

**Table 6.1.** Hyperparameters

| Hyperparameter | Value |
|---|---|
| Learning rate | 0.0001 |
| Dropout rate | 0.5 |
| Spatial embedding size, $h_{SE}$ | 16 |
| Spatial LSTM size, $h_s$ | 16 |
| Temporal embedding size, $h_{TE}$ | 16 |
| Temporal LSTM size, $h_t$ | 16 |
| Node LSTM size, $h_{node}$ | 32 |
| Node embedding size, $h_{NE}$ | 16 |
| Optimizer | Adam |
| Loss function | Mean Squared Error |

### 6.1.3. Region Partitioning Methods

The region of interest is partitioned with two different approaches. First one is regular grid partitioning and second is quadtree partitioning. These are detailed in the following sections.

- **Regular Grid :** The datasets have been stored in the X tensor consisting of a $[L, J, K]$ dimensional binary values, with $J$ and $K$ being the number of grid cells in the horizontal and vertical order, respectively. The $X_{ljk}$ $(l \in L, j \in J, k \in K)$ element of this tensor is assigned to, $1$ if there is an earthquake with magnitude $4.5$ and above, $0$ otherwise. We also tested the nonbinary representation where the $X_{ljk}$ $(l \in L, j \in J, k \in K)$ element of this tensor is assigned to the number of the earthquakes with magnitude $4.5$ and above. The results are not significantly different compared with the binary representation and we limit our discussion with the binary case. For China, the $3 \times 3$ subregion example is shown in the Figure 6.2., $5 \times 5$ subregion example is shown in the Figure 6.3..

- **Quadtree :** In this type of partitioning, the area is recursively divided into four subregions. This partitioning can be represented by a quadtree showing successive n-dimensional space subdivisions in four convex subspaces. In quadtree, partitioning a subspace is divided into two hyperplanes parallel to the coordinate axes recursively until the number of earthquakes is less than the specified minimum number of earthquakes. The parameters of the minimum number of earthquakes and maximum degree level are selected intentionally to obtain a number close to the number of subregions in regular grid configuration. Thus, the minimum number of earthquakes is set to 600,

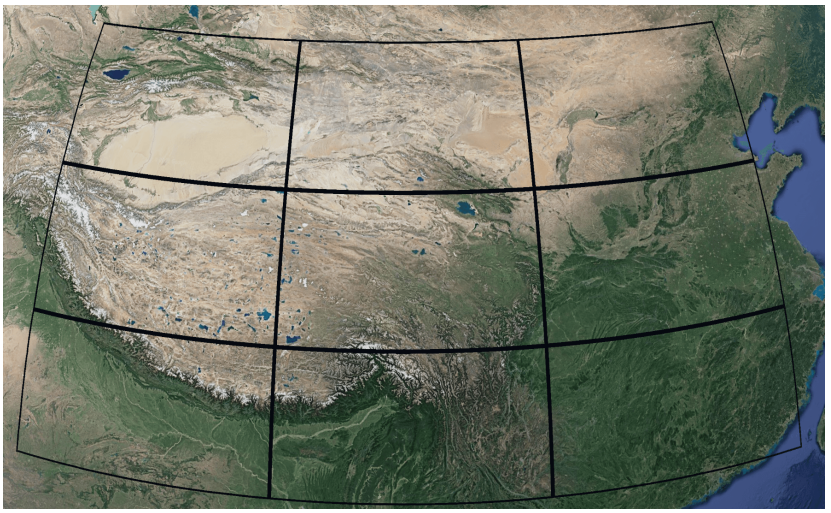

**Figure 6.2.** Regular grid example with $3 \times 3$ subregions in China

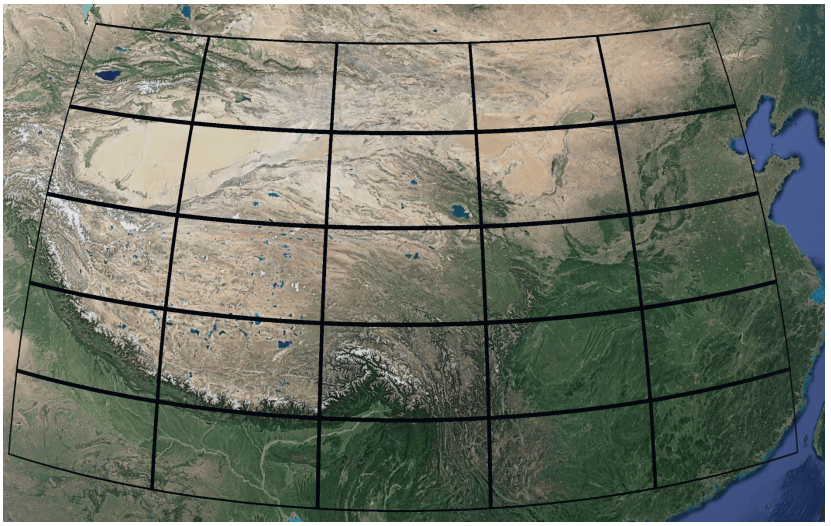

**Figure 6.3.** Regular grid example with $5 \times 5$ subregions in China

and the degree level is set to 3 to get 10 quads in order to compare with 9 regular grids. For comparing 25 regular grids, the minimum number of earthquakes is set to 375, and the degree level is set to 4 to get 25 quads in order to compare with 25 regular grids. An example of the quadtree with 10 subregions is illustrated in Figure 6.4. and an example with 25 subregions is illustrated in the Figure 6.5..

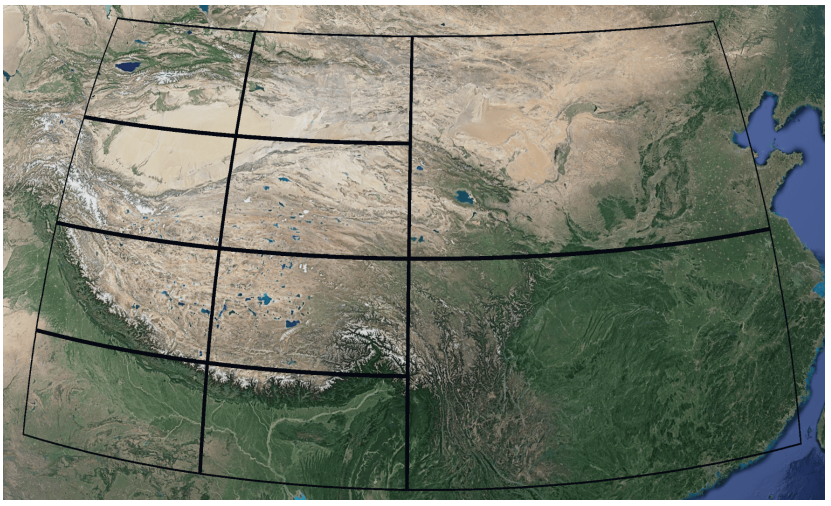

**Figure 6.4.** Quadtree example with 10 subregions in China

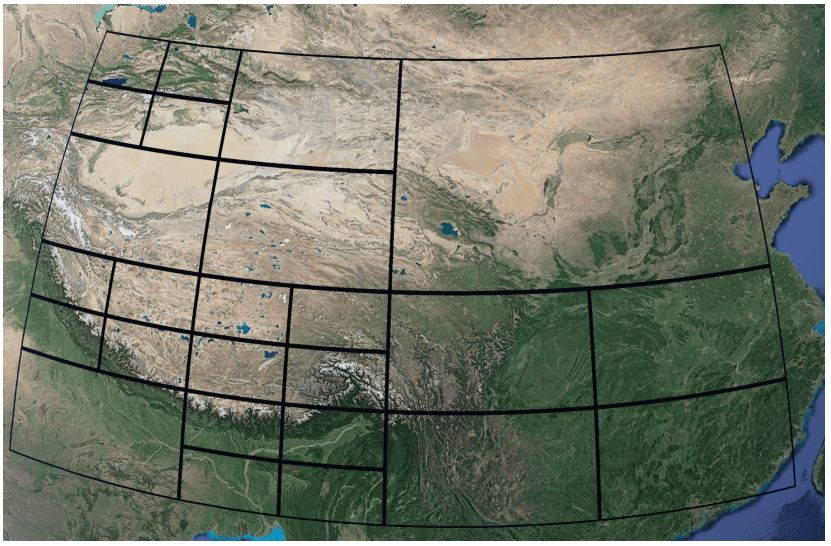

**Figure 6.5.** Quadtree example with 25 subregions in China

### 6.1.4. Sliding Windows

The process of sliding windows is shown in the Figure 6.6.. The data split ratio is set to be 0.75, 0.05, and 0.20 for training, validation, and test data, respectively. Models use the $S$ sequence length window data as input. The next month of the input sequence becomes the target. This process is displayed separately for the validation and test data in the Figure 6.6..

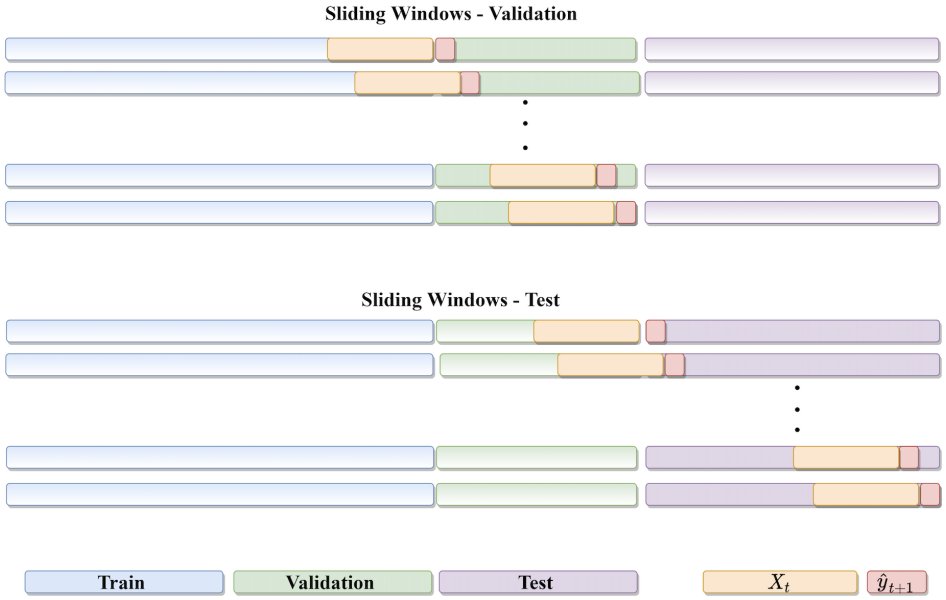


**Figure 6.6.** Sliding windows in validation and test

## 6.2. Performance Results

In this section, performance measurements are explained. Results of the experiments in both Turkey and China are presented. Moreover, experimental results of quadtree partitioning are provided in the China region.

### 6.2.1. Evaluation Metrics

$F_1$ score, precision (PR), recall (REC) are used to evaluate the test results. $TP$, $FP$, $FN$, and $TN$ denote True Positive $(Prediction : 1, Truth : 1)$, False Positive $(Prediction : 1, Truth : 0)$, and False Negative$(Prediction : 0, Truth : 1)$, True Negative $(Prediction : 0, Truth : 0)$ respectively. Consider the example where the prediction result for time $t$ is $\hat{y}_t = \{1, 0, 0\}$, and ground truth for time $t$ is $y_t = \{1, 0, 1\}$. For time $t$, the confusion matrix is calculated as $TP = 1$, $FP = 0$, $TN = 1$, $FN = 1$. For time $t + 1$, let assume $\hat{y}_{t+1} = \{0, 0, 1\}$, $y_{t+1} = \{0, 1, 0\}$, and the confusion matrix for time $t + 1$ becomes as $TP = 0$, $FP = 1$, $TN = 1$, $FN = 1$. The final confusion matrix is the sum of the confusion matrices of time $t$ and $t + 1$, and the total in this example results in $TP = 1$, $FP = 1$, $TN = 2$, $FN = 2$. Precision, recall, and $F_1$ score are determined by the sum

of $TP, FP, FN$ for all time windows in the test dataset. Formulations of the measures are shown in the following equations.

- **Precision** is calculated as the ratio of correctly predicted positive samples $(TP)$ to total predicted positive samples $(TP + FP)$. It is shown in the Equation (30).

$$Precision(PR) = \frac{TP}{TP + FP} \tag{30}$$

- **Recall** is calculated as the ratio of correctly predicted positive samples to total ground truth positives. It is shown in the Equation (31)

$$Recall(REC) = \frac{TP}{TP + FN} \tag{31}$$

- $F_1$ score is the harmonic mean of precision and recall. It is shown in the Equation (32).

$$F_1 = \frac{2 \times PR \times REC}{PR + REC} \tag{32}$$

### 6.2.2. Turkey

In the first set of experiments are presented in [55] with the SRNN model. It is shown that the experimental results of the SRNN model obtained better results than the SMA and LSTM models in [55].

**Table 6.2.** Performance of models in Turkey dataset

| | 3x3 Regular Grids | | | 5x5 Regular Grids | | |
|---|---|---|---|---|---|---|
| Model | Precision | Recall | $F_1$ Score | Precision | Recall | $F_1$ Score |
| SMA | 0.50 | 0.99 | 0.67 | 0.30 | 0.94 | 0.46 |
| OneNeuron | 0.64 | 0.66 | 0.68 | 0.46 | 0.57 | 0.51 |
| LSTM | 0.66 | 0.68 | 0.67 | 0.49 | 0.70 | 0.58 |
| $SRNN_{all}$ | 0.63 | 0.80 | **0.71** | 0.60 | 0.56 | 0.58 |
| $SRNN_{near}$ | 0.62 | 0.82 | **0.71** | 0.58 | 0.60 | **0.59** |

The results of Turkey experiments are shown in Table 6.2.. Two types of spatial relations are represented as $Model_{all}$ and $Model_{near}$ notation. $Model_{all}$ indicates that the spatial module

captures the spatial relations as if each node is connected to all others and $Model_{near}$ denotes only neighboring nodes have spatial spatial relation with each other. The $SRNN_{near}$ model gets better $F_1$ score in both $3 \times 3$ subregions and $5 \times 5$ subregions.

### 6.2.3. China

More comprehensively, the results of the SMA, OneNeuron, LSTM, and the proposed SRNN models are explored within the China region. The results of the experiments are listed in Table 6.3.. $SRNN$ denotes to the architecture where all nodes are from the same factor set, there is a single class and all nodes belong to the same class. $SRNNClass$ refers to the architecture where nodes are grouped into 4 non-overlapping classes as stated in Section 6.1..

**Table 6.3.** Performance of models in China dataset

| | 3x3 Regular Grids | | | 5x5 Regular Grids | | |
|---|---|---|---|---|---|---|
| Model | Precision | Recall | $F_1$ Score | Precision | Recall | $F_1$ Score |
| SMA | 0.54 | 0.76 | 0.63 | 0.40 | 0.81 | **0.54** |
| OneNeuron | 0.74 | 0.75 | 0.75 | 0.37 | 0.71 | 0.49 |
| LSTM | 0.74 | 0.77 | 0.75 | 0.42 | 0.70 | 0.53 |
| $SRNN_{all}$ | 0.66 | 0.84 | 0.74 | 0.39 | 0.85 | 0.53 |
| $SRNN_{near}$ | 0.72 | 0.78 | 0.75 | 0.39 | 0.88 | **0.54** |
| $SRNNClass_{all}$ | 0.65 | 0.82 | 0.73 | 0.34 | 0.93 | 0.50 |
| $SRNNClass_{near}$ | 0.77 | 0.76 | **0.77** | 0.42 | 0.74 | **0.54** |

In data preparation, different scales of subregions are tested and the two of the commonly used partitioning in earlier studies are presented in Table 6.3.. In the $3 \times 3$ partitioning, $SRNNClass_{near}$ model achieves the highest $F_1$ score compared to the other models that indicates model captures the structural and spatial proximity as expected.

The loss values over epochs are shown in Figure 6.7. and $F_1$ scores over epochs are displayed in Figure 6.8. for $SRNNClass_{near}$. The change in the loss function over epochs shows that the $SRNNClass_{near}$ model quickly saturates after a few epochs but the highest F1 score is achieved after around 15 epochs. It can be concluded that the model can learn the latent space in a small number of epochs. On the other hand, in the $5 \times 5$ partitioning, even simple moving average (SMA) model shares the best $F_1$ score with SRNN model. Dividing all regions into smaller scale subregions results in the data sparsity problem, which makes

it more difficult to deal with spatio-temporal relations. Here, it must be noted that not all subregions are highly ranked earthquake zones and low ranked subregions can be opted out from the evaluation. Also, the first-order neighborhood in spatial proximity is restricted to capture the sequential effects of earthquakes on fault lines, but if the scale of the subregions is small, the higher-order neighborhood can be used.

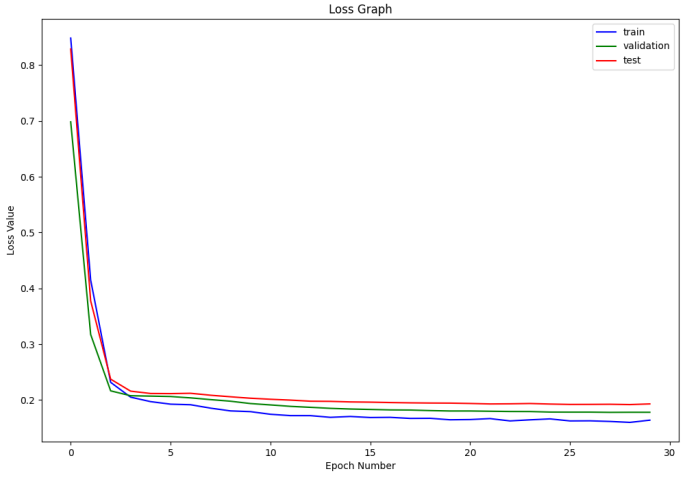


**Figure 6.7.** Loss over epochs $SRNNClass_{near}$ in China

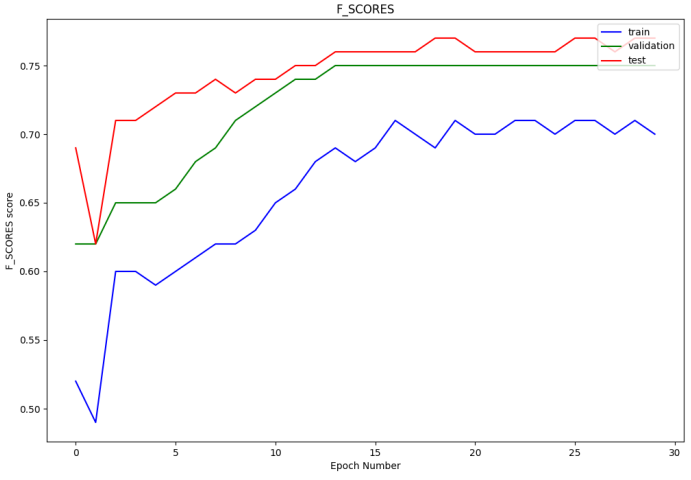


**Figure 6.8.** $F_1$ score over epochs $SRNNClass_{near}$ in China

The overall prediction results for the $LSTM$ model and $SRNNClass_{near}$ model together with the ground-truth target values are displayed in Figure 6.9.. In the figure, $x$ axes shows the time line in months for the duration of testing, and $y$ axis shows the region identifiers.

The presence of earthquakes for the relevant month is shown in black and otherwise shown in white. Regions 1, 4, 5, and 8 turn out to be highly ranked earthquake zones but regions 2, 3 and 9 have very sparse data. The results indicate that $LSTM$ model overfits and is prone to give the same predictions for all of subregions but with a small deviation in the region 7 over time, while $SRNNClass_{near}$ model has a better distribution of predictions in regions 1, 2, 3, 6, and 7. Confusion matrix for the $SRNNClass_{near}$ model is shown in Figure 6.10.. Regions 4, 5, 8 mostly resulted in a positive prediction that increased $TP$ and $FP$, while region 9 resulted in a negative class prediction that increased mostly $TN$ and $FN$. On the other hand, regions 2, 3, 6, 7 have reasonable distributions with respect to $TP$, $TN$, $FP$ and $FN$.

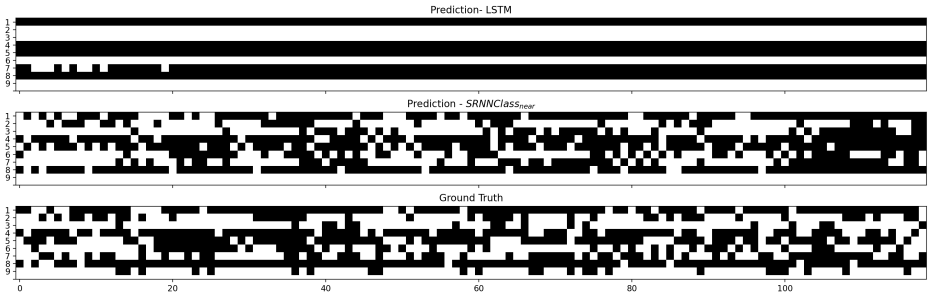


**Figure 6.9.** Time expanded view of prediction results for China dataset using 3x3 subregions


**Figure 6.10.** Confusion matrix for each region using $SRNNClass_{near}$ in China

In order to achieve an even distribution in all subregions, experiments have carried out in which the area is divided with quadtree partitioning. The experiment results are shown in Table 6.4..

In quadtree experiments, LSTM and SRNN models get the same $F_1$ scores. Regular grids gets best $F_1$ score $0.77$ for $3 \times 3$ subregions, $0.54$ for $5 \times 5$ subregions, while quadtree partioning gets $0.72$ for $10$ subregions and $0.52$ for $25$ subregions. We also aimed to produce more

**Table 6.4.** Performance of models in China dataset with quadtree partitioning

| | Quadtree 10 Subregions | | | Quadtree 25 subregions | | |
|---|---|---|---|---|---|---|
| Model | Precision | Recall | $F_1$ Score | Precision | Recall | $F_1$ Score |
| LSTM | 0.59 | 0.92 | 0.72 | 0.42 | 0.68 | 0.52 |
| $SRNN_{all}$ | 0.57 | 0.97 | 0.72 | 0.41 | 0.69 | 0.51 |

accurate estimates for intensive earthquake locations by dividing the earthquake-intensive regions into smaller subregions, but splitting the dense region data into smaller subregions did not improve model performance over regular grid partitioning.

# 7. CONCLUSION

Prediction of earthquakes is an active interdisciplinary research topic. Not much progress has been made so far, but with the advancement of computational systems and deep learning models, recent models and empirical results gave insights into the prediction problem. Although it seems very difficult to make consistent and accurate predictions, any work to be done on this subject is important.

In this thesis, we introduced deep learning models to capture the structural and spatio-temporal properties of the data in earthquake prediction. The proposed models utilize the spatial proximity and structural classification based on the fault lines. The proposed models are adaptations of the structural recurrent neural network architecture that is shown to achieve considerable success in different spatio-temporal problems having structural representations such as in traffic speed prediction, group activity prediction, and human trajectory prediction.

All models are tested with the catalog datasets including the location, timestamp, and Richter scale magnitudes of earthquakes collected within the region of Turkey and China. These two regions have distinct earthquake characteristics and zones and hence provides a general perspective on distinct regions. On the overall, the proposed SRNN models achieve slightly better results and do not overfit with the training data. With the utilization the spatial proximity and structural classification based on the fault lines, $SRNNClass_{near}$ model results in higher $F_1$ scores compared with the baseline (SMA) and state of the art models such as One Neuron, and LSTM.

The datasets are collected from earthquake catalogs that have a range of Richter scale. These earthquakes may be classified as main shocks or aftershocks. The source of data do not include such classification and hence in the partitioning of data into monthly time windows, there is no simple way to differentiate main shocks from aftershocks and hence all earthquakes are considered as shocks. It may be possible to integrate the notation of aftershocks in the model as any aftershock tends to appear after the main shock in closer locations and typically with a smaller magnitude in consecutive time windows.

One of the problems of using recurrent networks is the order of data is considered in consecutive time slots. To alleviate this limitation, attention networks can be utilized. We also conducted experiments by adding the attention module to the SRNN model, however it did

not perform better than the proposed models, so the results of these experiments with attention module are not presented but further studies on the attention mechanism to capture the aftershocks is our ongoing research direction.

A considerable volume of research studies the spatial data using evenly distributed partitioning such as rectangular subregions. Another possible research direction could be representing the closed boundaries of subregions via density or distance based partitioning the space according to the scale and distribution of earthquakes in closed proximity. In addition, different threshold selection approaches can be utilized for decreasing false positives by considering the Precision-Recall (PR) Curve and the Receiver Operating Characteristics (ROC) Curve.

# REFERENCES


[1] Robert J Geller. Earthquake prediction: a critical review. *Geophysical Journal International*, 131(3):425–450, **1997**.

[2] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, **2015**.

[3] Clarence R Allen. Responsibilities in earthquake prediction: to the seismological society of america, delivered in edmonton, alberta, may 12, 1976. *Bulletin of the Seismological Society of America*, 66(6):2069–2074, **1976**.

[4] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5308–5317. **2016**.

[5] Youngjoo Kim, Peng Wang, and Lyudmila Mihaylova. Structural recurrent neural network for traffic speed prediction. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5207–5211. IEEE, **2019**.

[6] Sovan Biswas and Juergen Gall. Structural recurrent neural network (srnn) for group activity analysis. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1625–1632. IEEE, **2018**.

[7] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE international Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, **2018**.

[8] Arnaud Mignan and Marco Broccardo. One neuron versus deep learning in aftershock prediction. *Nature*, 574(7776):E1–E3, **2019**.

[9] Q. Wang, Y. Guo, L. Yu, and P. Li. Earthquake prediction based on spatio-temporal data mining: An lstm network approach. *IEEE Transactions on Emerging Topics in Computing*, 8(1):148–158, **2020**. doi:10.1109/TETC.2017.2699169.

[10] 72 - earthquake prediction: An overview. In William H.K. Lee, Hiroo Kanamori, Paul C. Jennings, and Carl Kisslinger, editors, *International Handbook of Earthquake and Engineering Seismology, Part B*, volume 81 of *International Geophysics*, pages 1205 – 1216. Academic Press, **2003**.

[11] Robert Burridge and Leon Knopoff. Model and theoretical seismicity. *Bulletin of the seismological society of america*, 57(3):341–371, **1967**.

[12] Michio OTSUKA. A chain-reaction-type source model as a tool to interpret the magnitude-frequency relation of earthquakes. *Journal of Physics of the Earth*, 20(1):35–45, **1972**.

[13] Donald L Turcotte. *Fractals and chaos in geology and geophysics*. Cambridge university press, **1997**.

[14] Charles F Richter. An instrumental earthquake magnitude scale. *Bulletin of the seismological society of America*, 25(1):1–32, **1935**.

[15] Magnitude types. `https://www.usgs.gov/natural-hazards/earthquake-hazards/science/magnitude-types?qt-science_center_objects=0#qt-science_center_objects`. (Access date: on 12/29/2020).

[16] Paul A Johnson and Xiaoping Jia. Nonlinear dynamics, granular media and dynamic earthquake triggering. *Nature*, 437(7060):871–874, **2005**.

[17] Christopher H Scholz. *The mechanics of earthquakes and faulting*. Cambridge university press, **2019**.

[18] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, **1997**.

[19] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, **1986**.

[20] Sepp Hochreiter. Recurrent neural net learning and vanishing gradient. *International Journal of Uncertainity, Fuzziness and Knowledge-Based Systems*, 6(2):107–116, **1998**.

[21] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, **1997**.

[22] S Mostafa Mousavi, William L Ellsworth, Weiqiang Zhu, Lindsay Y Chuang, and Gregory C Beroza. Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. *Nature communications*, 11(1):1–12, **2020**.

[23] Thibaut Perol, Michaël Gharbi, and Marine Denolle. Convolutional neural network for earthquake detection and location. *Science Advances*, 4(2):e1700578, **2018**.

[24] Zachary E Ross, Yisong Yue, Men-Andrin Meier, Egill Hauksson, and Thomas H Heaton. Phaselink: A deep learning approach to seismic phase association. *Journal of Geophysical Research: Solid Earth*, 124(1):856–869, **2019**.

[25] Ashif Panakkat and Hojjat Adeli. Neural network models for earthquake magnitude prediction using multiple seismicity indicators. *International journal of neural systems*, 17(01):13–33, **2007**.

[26] Ashif Panakkat and Hojjat Adeli. Recurrent neural network for approximate earthquake time and location prediction using multiple seismicity indicators. *Computer-Aided Civil and Infrastructure Engineering*, 24(4):280–292, **2009**.

[27] JP Huang, XA Wang, Y Zhao, C Xin, and H Xiang. Large earthquake magnitude prediction in taiwan based on deep learning neural network. *Neural Network World*, 28(2):149–160, **2018**.

[28] Xingbo Fu, Feng Gao, Jiang Wu, Xinyu Wei, and Fangwei Duan. Spatiotemporal attention networks for wind power forecasting. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 149–154. IEEE, **2019**.

[29] Roman Kail, Alexey Zaytsev, and Evgeny Burnaev. Recurrent convolutional neural networks help to predict location of earthquakes. *arXiv preprint arXiv:2004.09140*, **2020**.

[30] Masashi Hayakawa. Earthquake prediction with electromagnetic phenomena. In *AIP Conference Proceedings*, volume 1709, page 020002. AIP Publishing LLC, **2016**.

[31] Min Jiang. Easily magnetic anomalies earthquake prediction. In *MATEC Web of Conferences*, volume 63, page 01020. EDP Sciences, **2016**.

[32] JN Thomas, F Masci, Jeffrey J Love, et al. On a report that the 2012 m6: 0 earthquake in italy was predicted after seeing an unusual cloud formation. *Natural Hazards and Earth System Sciences (NHESS)*, **2015**.

[33] Jing Fan, Zhong Chen, Liang Yan, Jing Gong, and Dong Wang. Research on earthquake prediction from infrared cloud images. In *MIPPR 2015: Remote Sensing Image Processing, Geographic Information Systems, and Other Applications*, volume 9815, page 98150E. International Society for Optics and Photonics, **2015**.

[34] Masashi Hayakawa, Hiroyuki Yamauchi, Nobuyo Ohtani, Mitsuaki Ohta, Susumu Tosa, Tomokazu Asano, Alexander Schekotov, Jun Izutsu, Stelios M Potirakis, Konstantinos Eftaxias, et al. On the precursory abnormal animal behavior and electromagnetic effects for the kobe earthquake (m˜ 6) on april 12, 2013. *Open Journal of Earthquake Research*, 5(03):165, **2016**.

[35] Rex V Allen. Automatic earthquake recognition and timing from single traces. *Bulletin of the Seismological Society of America*, 68(5):1521–1532, **1978**.

[36] M Baer and U Kradolfer. An automatic phase picker for local and teleseismic events. *Bulletin of the Seismological Society of America*, 77(4):1437–1445, **1987**.

[37] Anthony Lomax, Claudio Satriano, and Maurizio Vassallo. Automatic picker developments and optimization: Filterpicker—a robust, broadband picker for real-time seismic monitoring and earthquake early warning. *Seismological Research Letters*, 83(3):531–540, **2012**.

[38] Steven J Gibbons and Frode Ringdal. The detection of low magnitude seismic events using array-based waveform correlation. *Geophysical Journal International*, 165(1):149–166, **2006**.

[39] Clara E Yoon, Ossian O'Reilly, Karianne J Bergen, and Gregory C Beroza. Earthquake detection through computationally efficient similarity search. *Science advances*, 1(11):e1501057, **2015**.

[40] Qi Zhang, Tong Xu, Hengshu Zhu, Lifu Zhang, Hui Xiong, Enhong Chen, and Qi Liu. Aftershock detection with multi-scale description based neural network.

In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 886–895. IEEE, **2019**.

[41] Winda Astuti, Rini Akmeliawati, Wahju Sediono, and Momoh-Jimoh E Salami. Hybrid technique using singular value decomposition (svd) and support vector machine (svm) approach for earthquake prediction. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(5):1719–1728, **2014**.

[42] Risul Islam Rasel, Nasrin Sultana, GM Azharul Islam, Mahfuzul Islam, and Phayung Meesad. Spatio-temporal seismic data analysis for predicting earthquake: Bangladesh perspective. In *2019 Research, Invention, and Innovation Congress (RI2C)*, pages 1–5. IEEE, **2019**.

[43] Phoebe MR DeVries, Fernanda Viégas, Martin Wattenberg, and Brendan J Meade. Deep learning of aftershock patterns following large earthquakes. *Nature*, 560(7720):632–634, **2018**.

[44] Arnaud Mignan and Marco Broccardo. A deeper look into 'deep learning of aftershock patterns following large earthquakes': Illustrating first principles in neural network physical interpretability. In *International Work-Conference on Artificial Neural Networks*, pages 3–14. Springer, **2019**.

[45] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys*, 51(4), **2018**. ISSN 0360-0300. doi:10.1145/3161602.

[46] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, **2006**.

[47] Xiaoguang Niu, Ying Zhu, and Xining Zhang. Deepsense: A novel learning mechanism for traffic prediction with taxi gps traces. In *2014 IEEE global communications conference*, pages 2745–2750. IEEE, **2014**.

[48] Antonios Karatzoglou, Nikolai Schnell, and Michael Beigl. A convolutional neural network approach for modeling semantic trajectories and predicting future locations. In *International Conference on Artificial Neural Networks*, pages 61–72. Springer, **2018**.

[49] Jianming Lv, Qing Li, Qinghui Sun, and Xintong Wang. T-conv: A convolutional neural network for multi-scale taxi trajectory prediction. In *2018 IEEE international conference on big data and smart computing (bigcomp)*, pages 82–89. IEEE, **2018**.

[50] Meng Chen, Xiaohui Yu, and Yang Liu. Pcnn: Deep convolutional networks for short-term traffic congestion prediction. *IEEE Transactions on Intelligent Transportation Systems*, 19(11):3550–3559, **2018**.

[51] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 984–992. **2018**.

[52] Xingjian Shi, Zhihan Gao, Leonard Lausen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Deep learning for precipitation nowcasting: A benchmark and a new model. In *Advances in neural information processing systems*, pages 5617–5627. **2017**.

[53] Amol Kapoor, Xue Ben, Luyang Liu, Bryan Perozzi, Matt Barnes, Martin Blais, and Shawn O'Banion. Examining covid-19 forecasting using spatio-temporal graph neural networks. *arXiv preprint arXiv:2007.03113*, **2020**.

[54] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, **2017**.

[55] A. Doğan and E. Demir. Earthquake prediction in turkey with structural recurrent neural networks. In *2020 28th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. **2020**.

[56] Ömer Emre, Tamer Y Duman, Selim Özalp, Fuat Şaroğlu, Şeyda Olgun, Hasan Elmacı, and Tolga Can. Active fault database of turkey. *Bulletin of Earthquake Engineering*, 16(8):3229–3275, **2018**.

[57] Yong-Fei Zheng, Wen-Jiao Xiao, and Guochun Zhao. Introduction to tectonics of china. *Gondwana Research*, 23(4):1189 – 1206, **2013**.

[58] Harvey J. Miller. Tobler's first law and spatial analysis. *Annals of the Association of American Geographers*, 94(2):284–289, **2004**. ISSN 00045608, 14678306.