Article

Predicting the magnitude of earthquakes using Grammatical Evolution

Constantina Kopitsa[1], Ioannis G. Tsoulos[2,*] and Vasileios Charilogis[3]

[1] Department of Informatics and Telecommunications, University of Ioannina, Greece;k.kopitsa@uoi.gr
[2] Department of Informatics and Telecommunications, University of Ioannina, Greece; itsoulos@uoi.gr
[3] Department of Informatics and Telecommunications, University of Ioannina, Greece; v.charilog@uoi.gr
* Correspondence: itsoulos@uoi.gr

Abstract: Throughout history, human societies have sought to explain natural phenomena through the lens of mythology. Earthquakes, as sudden and often devastating events, inspired a range of symbolic and mythological interpretations across different civilizations. It was not until the 18th and 19th centuries that a more positivist and scientific approach began to emerge regarding the explanation of earthquakes, recognizing their origin as stemming from processes occurring beneath the Earth's surface. A pivotal moment in the emergence of modern seismology was the Lisbon earthquake of 1755, which marked a significant shift towards scientific inquiry. This means that the question of how earthquakes occur has been resolved, thanks to advancements in scientific, geological, and geophysical research, it is now well understood that seismic events result from the collision, and movement of lithospheric or tectonic plates. The contemporary challenge that emerges, however, lies in whether such seismic phenomena can be accurately predicted. In this paper, a systematic attempt is made to use techniques based on Grammatical Evolution to determine the magnitude of earthquakes. These techniques use freely available data in which the history of large earthquakes is introduced before the application of the proposed techniques. From the execution of the experiments, it became clear that the use of these techniques can estimate the magnitude of earthquakes more effectively compared to other machine learning techniques from the relevant literature.

Keywords: Earthquakes; Machine learning; Evolutionary computation; Grammatical Evolution

Received:
Revised:
Accepted:
Published:

Citation: Kopitsa, C.; Tsoulos, I.G.; Charilogis, V. . Predicting the magnitude of earthquakes using Grammatical Evolution. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/

1. Introduction

Seismology, like most scientific disciplines, matured over the centuries through the rational evolution of human reasoning. For many centuries, the interpretation of earthquakes was rooted in mythology, and it was only during the era of the profound Industrial Revolution that it became widely accepted as a geological phenomenon rather than an act of divine punishment. Despite this, one of the earliest seismic instruments, known as the seismoscope, did not record the timing or duration of ground motion but merely signaled that seismic activity had taken place. This type of device was first developed by the Chinese scholar Zhang Heng as early as 132 CE [1]. Robert Mallet (1810 – 1881), often referred to as the "father of seismology" due to his pioneering contributions to the study of earthquakes, was an Irish geophysicist, civil engineer, and scientific researcher [2]. In today's era, at the threshold between the Fourth and Fifth Industrial Revolutions, the science of seismology continues its quest for the 'Holy Grail' of accurately predicting an impending earthquake. "According to the United States Geological Survey (USGS), neither the USGS nor any other scientific institution has ever successfully predicted a major earthquake. A valid earthquake prediction must specify three critical elements: (1) the exact date and time, (2) the precise location, and (3) the anticipated magnitude [3].

Nevertheless, earthquake prediction was long considered an unattainable goal prior to the advent of artificial intelligence technologies. In recent years, however, annual conferences of leading scientific bodies such as the Seismological Society of America and the American Geophysical Union have included dedicated sessions on the application of machine learning (ML) in geophysical sciences. These scientific advancements, driven by the integration of AI-based methods, have increasingly captured the attention of seismologists and researchers in the field [4]. The necessity of accurate prediction, is also evidenced by the following alarming data by the International Disaster Database EM-DAT https://www.emdat.be/ between 2010 and 2025 earthquakes have resulted in 337.372 deaths, 698.085 injuries and 1.547.581 individuals left homeless. These data are depicted in Table 1. It is important to note that EM-DAT records only those events that meet at least one of the following criteria: ten or more fatalities, one hundred or more individuals affected, the declaration of a state of emergency, or a call for international assistance. These figures underscore the significance of seismic disasters and lead us to emphasize the urgent need for scientific advances in earthquake prediction. In order to contextualize this necessity further, it is worth examining comparable human losses caused by other natural disasters.

Table 1. Natural Disasters from EM-DAT

2010 - 2025	Deaths	Injured people	Homeless people
Flood	82.644	111.954	7.552.142
Storms	51.091	125.266	3.671.100
Wildfires	1.622	13.810	94.925
Earthquakes	337.372	698.085	1.547.581

These are the stark figures that reveal how vulnerable humanity remains even in our modern era, in the face of natural disasters events that are further intensified, by the effects of climate change. Nevertheless, there is room for optimism, as researchers are increasingly employing artificial intelligence and other advanced methodologies in the pursuit of forecasting earthquakes (and other natural disasters) in a timely manner, with the ultimate goal of enabling populations to seek safety and reduce potential losses [5].Consequently, emphasis has been placed on various scientific approaches that investigate events potentially influencing or triggering the release of seismic energy. The following are some examples illustrating these approaches.

In the state of Oklahoma, researchers have developed a sophisticated Bayesian approach to assess the relationship between wastewater injection practices, and the occurrence of seismic events [6]. In early 2002, a study was conducted examining the correlation between long-period seismic tremors occurring at a depth of approximately 30 kilometers, and underlying tectonic processes [7]. The connection was established in 2016, highlighting that the careful and precise monitoring of slow-slip events may yield valuable insights into the probability of forthcoming large-magnitude earthquakes [8]. An alternative approach involves the systematic monitoring of potential precursory signals, preceding the onset of a major seismic event [9]. Several researchers have suggested alternative mechanisms, including electromagnetic anomalies and ionospheric disturbances, although investigations in this area remain in their preliminary stages [5]. In addition to these approaches, certain institutions have also explored the study of animal behavior as a potential means of anticipating seismic activity [10].

The following section presents researcher's efforts to forecast imminent seismic events through the application of machine learning techniques. The subsequent research proposes an efficient and accurate framework for estimating earthquake magnitudes directly from unprocessed single-station waveform data, yielding minimal average error and a standard deviation close to 0.2, without requiring instrument response adjustments, as validated using seismic records from Southern California [11]. Next, the following study demonstrated that machine learning–driven approaches enhance the accuracy of aftershock location forecasts and shed light on key physical parameters that may govern earthquake triggering during the most dynamic phase of the seismic cycle [12]. In the following paper, the researchers trained convolutional neural networks (CNNs) on over 18 million manually picked seismographs from Southern California, to estimate these parameters directly from raw waveform data, without the need for feature extraction. The model demonstrated high precision, achieving a standard deviation of just 0.023 seconds in arrival times and 95% accuracy in polarity classification [13]. Building on recent advancements, the upcoming study employs machine learning techniques on data-sets derived from shear laboratory experiments, aiming to uncover previously undetected signals that may precede seismic events [14]. Members of the previous study, subsequently conducted an analysis using a machine learning-based method initially developed in the laboratory. They processed extensive raw seismic data from Vancouver Island to distinguish the relevant signal from background seismic noise, an approach that may prove valuable in assessing whether and how a slow slip event could be coupled with or evolve into a major earthquake [15]. The next step involves, a high-resolution earthquake catalog developed through machine learning techniques, provides new insights into the complexity, and duration of earthquake sequences, as well as their interrelation with recent neighboring seismic events [16]. What follows is a study, that introduces a global deep learning model capable of concurrently detecting earthquakes and identifying seismic phases. The proposed model demonstrates superior performance compared to existing deep learning approaches and conventional detection and phase-picking methods. Notably, it enables the detection, and localization of approximately twice as many earthquakes, while utilizing less than one-third of the available seismic stations [17]. The subsequent study focused on California, and demonstrated that nearest-neighbor diagrams offer a straightforward and effective method for distinguishing between various seismic patterns, and evaluating the reliability of earthquake catalogs [18]. The following research team, concluded that the Weibull model provides a superior fit to the seismic data for California, exhibiting well-behaved tail characteristics. Furthermore, they demonstrated its robustness by applying it successfully to independent data sets from Japan, Italy, and New Zealand [19]. The consequent article, aimed to predict both the location and magnitude of potential earthquakes occurring in the following week, based on seismic data from the current week, focusing on seismogenic regions in southwestern China. The model achieved a testing accuracy of 70%, with corresponding precision, recall, and F1-score values of 63.63%, 93.33%, and 75.66%, respectively [20]. Moreover, the ensuing study focuses on predicting earthquake magnitudes in the Hindukush region. Four machine learning techniques namely the pattern recognition: neural network, recurrent neural network, random forest, and linear programming boost ensemble classifier, are individually implemented to model the relationships between computed seismic parameters, and the occurrence of future earthquakes [21]. Additionally, another study demonstrates that machine learning can effectively predict the timing and size of laboratory earthquakes by reconstructing and making sense of the system's intricate spatiotemporal loading history [22]. In a related study, a machine learning technique is applied to the regions of Japan, Turkey, Greece, and the Indian Subcontinent. The model reveals a relationship between the computed seismic data and the occurrence of future earthquakes [23]. The following paper compares the performance of a machine learning model, which uses a limited set of predictor variables surface roughness, peak frequency (fP), HV, VS30, and depth (Z2.5) to that of a physics-based model (GRA) that relies on detailed 1D velocity profiles. Results indicate that the machine learning approach, outperforms the physics-based modeling in terms of predictive accuracy [24]. Another study, investigates the physical and dynamic variations in seismic data, and introduces a novel machine learning method called Inverse Boosting Pruning Trees (IBPT). This approach is designed to provide short-term forecasts, based on satellite data from 1,371 earthquakes of magnitude six or higher, given their significant environmental impact [25]. In contemporary scientific research, there is a growing surge of interest in the prediction of seismic events. Advancements in data acquisition technologies, communication networks, edge–cloud computing, the Internet of Things (IoT), and big data analytics have created favorable conditions for the development of intelligent earthquake prediction models, enabling early warning systems in vulnerable regions [26]. In this context, it is noteworthy that in recent years there has been significant development of various models for the early detection of seismic events, which are now accessible to the general public through mobile devices in the form of applications (apps), TV media, or radio. For instance, in Japan, significant investment has been made in the timely dissemination of information regarding seismic events since 2007, through the implementation of the Earthquake Early Warning (EEW) system [27]. In this manner, an alert for an impending earthquake can be issued from up to one minute to several seconds before the event occurs [28]. Moreover, the United States has also developed its own earthquake warning system through the U.S. Geological Survey. Since 2016, the system known as Shake Alert has been implemented for the West Coast [29,30]. In Southern Europe, specifically at the University of Naples in Italy, a software model known as PRESTo has been developed, which is capable of detecting earthquakes approximately within the last ten seconds before their occurrence. It is abundantly clear that the science of machine learning, has entered the field of seismology with considerable momentum, a fact that is also reflected in the bibliographic references, with two articles authored by specialists in the fields of geological and geophysical sciences, which highlight how machine learning contributes to the advancement of earthquake prediction. In this article, a new generation of earthquake catalogs, developed through supervised machine learning, provides unprecedented detail in capturing seismic activity. The use of unsupervised machine learning to analyze this more comprehensive representation of seismicity is suggested as the most efficient path toward enhancing earthquake forecasting [31]. The same researchers, emphasize that machine learning and data mining techniques can greatly enhance our ability to process seismic data. In their review, they offer a comprehensive overview of machine learning applications in earthquake seismology, discuss recent advancements and existing challenges, and propose directions for future research [32].

In the current work seismological data collected from NSF Seismological Facility were obtained and afterwards these data was pre-processed, where for each earthquake of magnitude greater than 5, the closest earthquakes that had preceded it in various parts of the planet within a critical distance were identified. Then, the number of these preceding seismic vibrations and their average magnitude were added to these recordings. The purpose of the above process is to achieve the most accurate prediction of the magnitude of an earthquake based on its characteristics and the earthquakes that have preceded it within a predetermined distance. An attempt was then made to predict the magnitude of earthquakes using machine learning techniques based on the Grammatical Evolution method [33]. Grammatical Evolution is an evolutionary technique where chromosomes (candidate solutions) are expressed as a series of production rules of a provided BNF grammar [34]. Grammatical Evolution was used in a series of problems, such as data fitting [35,36], composition of music [37], video games [38,39], energy problems [40], cryptography [41], economics [42] etc. The methods based on Grammatical Evolution was compared against various optimization techniques incorporated to train artificial neural networks [43,44] for the prediction of magnitude of earthquakes and the results are presented and discussed.

The limitations of the proposed work stem primarily from those we ourselves imposed, as it is focused on the analysis of data from a three-year period. Moreover, in order to avoid overfitting, we deliberately excluded a significant portion of data from each year and concentrated on seismic events with a magnitude greater than 5.

In this study, is was observed that machine learning and soft computing techniques have a longstanding presence in the field of seismology. For instance, Artificial Neural Networks (ANNs) were introduced into the field of seismology in 1994 [45]. The initial application of Deep Neural Networks (DNNs), featuring two hidden layers, emerged in 2002 [46], while the earliest implementation of Recurrent Neural Networks (RNNs) in the context of seismology appeared in 2007 [47]. However, they have yet to achieve the so-called "triple prediction" namely, the accurate forecasting: of the date and time, location, and precise magnitude of seismic events. For this reason, we adopt a groundbreaking approach by employing Grammatical Evolution in 2025, a novel technique within the domain. Furthermore, our initial approach involved the application of machine learning techniques. Specifically, we conducted experiments using the following algorithms: (LSTM, RBF, MLP (BP, RPROP, BFGS) & SVM). Subsequently, we carried out an exploratory experiment employing Grammatical Evolution. The results proved to be so compelling that we decided to shift our focus and pursue this novel direction in greater depth. Moreover, our approach distinguishes itself from other studies in the field, as most existing research relies on data obtained directly from seismographs and primarily focuses on localized regions. In contrast, we process historical earthquake data on a global scale, allowing for broader generalization and pattern recognition

Despite these constraints, our study highlighted the considerable potential of Grammatical Evolution in this domain, due to its consistently dynamic adaptability, high predictive accuracy, and low error rates even for earthquakes, occurring at distances ranging from 25 to 500 miles. Therefore, Grammatical Evolution proves to be both pioneering and innovative within the field, offering substantial promise to advance the discipline of seismology. The methods used in this work, which make systematic use of the Grammatical Evolution technique, include the construction of artificial features, the construction of programming rules, and the creation of Artificial Neural Networks. The above techniques can be used to effectively explore the objective problem space, as they have the ability to isolate the most important features of the problem but also to detect and present hidden correlations between the features of the problem.

The remaining of this paper has the following structure: the section 2 presents the used dataset as well as the proposed machine learning techniques, the section 3 illustrates the experimental results finally the section 4 presents some conclusions.

2. Materials and Methods

2.1. The used dataset

In this paper, there were used open data, available from NSF Seismological Facility for the Earth Consortium (SAGE) and especially from the Interactive Earthquake Browser http://ds.iris.edu/ieb/. The data were obtained from the NSF, as it offered greater functionality. Specifically, while the GEOFON program provided similar information, it imposed a limitation on the maximum number of earthquakes retrievable per request (1,000 events), which significantly constrained the selectable time range. This is particularly restrictive, given that approximately 1,000 seismic events can occur within a single day. The NSF SAGE Facility has been certified as a trustworthy data repository by the CoreTrustSeal

Standards and Certification Board. The data were collected for the years 2004, 2010, and 2012, with each year comprising over 100,000 recorded earthquake events. For every recorded earthquake event, comprehensive data were systematically gathered, including the date, exact time of occurrence, geographic coordinates,depth, and magnitude. This information enabled temporal analyses and the identification of seismic patterns across different time intervals. Moreover, our data set employs the Moment Magnitude scale, as it operates effectively across a broader range of earthquake sizes, and is applicable on a global scale. During the initial stages of preprocessing, it became evident that earthquake events should be organized based on lithospheric plate boundaries rather than national borders, which had initially been our approach. A preprocessing procedure was applied to the data, including the identification of the lithospheric plate associated with each earthquake. Subsequently, each earthquake location was cross-referenced with nearby volcanoes, where applicable, using data from the Smithsonian Institution's Global Volcanism Program https://volcano.si.edu/. During the course of our experiments, we decided to proceed with earthquakes of magnitude 5 and above, as including lower-magnitude events would result in the model being trained primarily on the prediction of minor seismic occurrences. We approached our research on earthquake magnitude prediction as a regression problem, employing the technique of Grammatical Evolution. To achieve this objective, we used the following features as input variables: date, time, latitude, longitude, depth, lithospheric plate, type of nearby volcano, magnitude, and magnitude code. In addition, a predefined distance, denoted as D_c in the experiments, from the epicenter of the target event (10 miles, 25 miles, 50 miles, 100 miles, and 500 miles) was also taken into account. The numerical output of the model predicted the earthquake's intensity with high predictive accuracy, mean absolute error below 0.5 magnitude units per event.

The following section provides a concise summary as well as a more detailed analysis of our data set. In the year 2004, a total of 666 geographic regions were classified, and used as an input for the regression model, within which 215,753 seismic events were recorded. Similarly, in 2010, 635 regions were defined and encoded as input features, corresponding to 327,909 earthquakes, while in 2012, 690 regions were established as categorical input variables, with a total of 405,153 recorded seismic events. Regarding tectonic plates, for each seismic event, we classified and used as an input feature the tectonic plates involved in the corresponding region. In total, 81 distinct combinations of lithospheric tectonic plates were identified. In relation to the classification of volcanoes, these were categorized into ten types: stratovolcano, volcanic field, lava dome, caldera, complex, compound, shield, pyroclastic, minor, and submarine. For each region containing a volcano, the corresponding category was marked as an input feature with a value of 1, while a value of 0 was assigned when no volcano of that type was present. Table 2 shows the classification of earthquakes into various classes, depending on their magnitude for each year studied in this work.

Table 2. Classification of earthquakes.

	2004	2010	2012
Earthquakes	126.972	294.432	370.582
0 - 4.9 mag	126.003	292.387	369.101
5 - 5.9 mag	893	1.909	1.374
6 - 6.9 mag	68	117	92
7 - 7.9 mag	7	19	13
8 - 8.9 mag	1	0	2
9 - 10 mag	1	0	0

The transformation from raw seismic events to structured input for machine learning was conducted as follows:

1. On the Earthquake Interactive Browser platform, the "maximum earthquakes" parameter was set to 25,000 in order to extract the maximum available number of records.
2. The "time range" was then adjusted to correspond to the specific year or range of years targeted for data collection.
3. Subsequently, the "magnitude range" was specified, the filter was applied, and the dataset was downloaded in Excel format.
4. The final dataset was further processed by creating a separate column for each variable, including: Year, Month, Day, Time, Latitude, Longitude, Depth, Magnitude, Magnitude Code, Region, Region Code, Lithospheric/Tectonic Plate, Lithospheric/Tectonic Plate Code, Stratovolcano, Volcanic Field, Lava Dome, Caldera, Complex, Compound, Shield, Pyroclastic, Minor, and Submarine.

In order to enhance the reliability of the used dataset, extensive data cleaning was performed by removing several thousands of records from each year. For instance, in 2004, a total of 214.170 entries were excluded, in 2010, 325.278 records were removed, and in 2012, 403.035 were similarly discarded. The extensive data cleaning process played a crucial role in preventing our model from predominantly learning to predict the values of low-magnitude earthquakes, which vastly outnumbered higher-magnitude events, by several hundreds of thousands.

In the used preprocessing pipeline, categorical variables such as geographic regions, and lithospheric / tectonic plates were encoded using a unique integer-based labeling scheme. Specifically, each distinct geographic region and tectonic plate was assigned a unique numeric identifier, ranging sequentially from 1 up to the number of unique entries in the respective category. Regarding volcanic types, a binary encoding approach was applied. For each type of volcano (e.g., stratovolcano, caldera, shield, etc.), we created a binary feature indicating the presence or absence of that specific volcano type within a given region. A value of 1 was assigned if the volcano type was present, and 0 otherwise. This strategy enabled the model to capture the influence of specific volcanic characteristics while maintaining compatibility with the GE algorithm.

2.2. Grammatical Evolution preliminaries

The Grammatical Evolution procedure is considered as a variant of genetic algorithms [48,49], where the chromosomes are series of integer values that represent production rules of the underlying BNF grammar. The BNF grammars are denoted as sets $G = (N, T, S, P)$, with the following definitions:

- The set N represents the non - terminal symbols.
- The set T contains the terminal symbols of the language.
- The symbol $S \in N$ denotes the start symbol of the grammar.
- The set P holds the production rules of the grammar.

The production mechanism of Grammatical Evolution initiates from the symbol S and using a series of production rules, a valid program is created replacing non-terminal symbols with the right hand of the selected production rule. The rules are selected through the following procedure:

- **Read** the next element V from the current chromosome.
- **Select** the production rule using the equation:Rule = V mod N_R. The constant N_R stands for the total number of production rules for the non – terminal symbol that is currently under processing.

The previously used procedure is graphically illustrated in Figure 1.

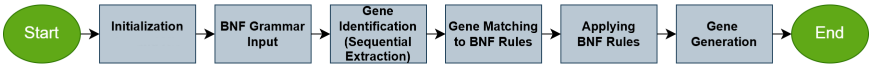

Figure 1. The production mechanism of the Grammatical Evolution procedure.

As a full working example of production consider the chromosome:

$$x = [9, 8, 6, 4, 16, 10, 17, 23, 8, 14]$$

and the grammar of Figure 2. The numbers shown in parentheses are the increasing numbers of the production rules for each non - terminal symbol. Denote with $d = 3$ the number of features (inputs) for the current dataset. The production of the valid expression $f(x) = x_2 + \cos(x_3)$ is performed through a series of steps, that are depicted in Table 3.

Figure 2. A full example of a BNF grammar that produces simple expression.

```
S::=<expr>   (0)
<expr> ::=  (<expr> <op> <expr>)  (0)
           | <func> ( <expr> )    (1)
           |<term>            (2)
<op> ::=     +      (0)
           | -      (1)
           | *      (2)
           | /      (3)
<func> ::=   sin  (0)
           | cos  (1)
           |exp   (2)
           |log   (3)
<term>::=<xlist>                  (0)
           |<dlist>.<dlist> (1)
<xlist>::=x1    (0)
           | x2 (1)
           .........
           | xd (d-1)
<dlist>::=<digit>                    (0)
           | <digit><digit>              (1)
           | <digit><digit><digit>       (2)
<digit>  ::= 0 (0)| 1 (1)| 2 (2)| 3 (3)| 4 (4)
           | 5 (5)| 6 (6)| 7 (7)| 8 (8)| 9 (9)
```

Table 3. The series of performed steps for the production of a valid expression.

Expression	Chromosome	Operation
<expr>	9,8,6,4,16,10,17,23,8,14	$9 \mod 3 = 0$
(<expr><op><expr>)	8,6,4,16,10,17,23,8,14	$8 \mod 3 = 2$
(<terminal><op><expr>)	6,4,16,10,17,23,8,14	$6 \mod 2 = 0$
(<xlist><op><expr>)	4,16,10,17,23,8,14	$4 \mod 3 = 1$
(x2<op><expr>)	16,10,17,23,8,14	$16 \mod 4 = 0$
(x2+<expr>)	10,17,23,8,14	$10 \mod 3 = 1$
(x2+<func>(<expr>))	17,23,8,14	$17 \mod 4 = 1$
(x2+cos(<expr>))	23,8,14	$23 \mod 2 = 2$
(x2+cos(<terminal>))	8,14	$8 \mod 2 = 0$
(x2+cos(<xlist>))	14	$14 \mod 3 = 2$
(x2+cos(x3))		

2.3. The rule production method

The rule construction method was initially presented in [50]. This method can produce rules in a human readable form, that can be used in classification and regression problems without any prior knowledge of the the objective problem. The main steps of this method have as follows:

- **Step 1 - Initialization step**.
  1. **Set** N_c the number of chromosomes and as N_g the maximum number of allowed generations.
  2. **Set** as p_s the selection rate of the genetic algorithm and as p_m the corresponding mutation rate.
  3. **Initialize** the chromosomes $c_i,\ i = 1, \ldots, N_c$. Each chromosome is considered as a set of randomly selected positive integers.
  4. **Set** $k = 0$, the generation counter.
- **Step 2 - Fitness calculation step**.
  1. **For** $i = 1, \ldots, N_c$ **do**
     (a) **Create** the program P_i for the chromosome c_i using the grammar of Figure 3 and the Grammatical Evolution production mechanism.
     (b) Set as the fitness value f_i for the chromosome c_i the training error of the produced program, calculated as:

$$f_i = \sum_{j=1}^{M} \left(P_i(x_j) - y_j\right)^2$$

     The set $\left(x_j, y_j\right),\ x \in R^N, j = 1, \ldots, M$ defines the training set of the objective problem, where the valuey_j is considered as the actual output for the input pattern x_j. In the current implementation of Grammatical Evolution, the fitness function is defined as the sum of squared errors between the predicted and actual earthquake magnitudes across the training set. This fitness function is crucial in guiding the evolutionary process by favoring candidate solutions (programs) that minimize prediction error. As earthquake magnitude prediction is a regression task, this formulation ensures that evolved models are optimized for minimizing the deviation from actual magnitudes.
  2. **End For**
- **Step 3 - Genetic operations step**.
  1. **Select** the best $(1 - p_s) \times N_c$ chromosomes from the current population. These chromosomes will be transferred intact to the next generation.
  2. **Create** $p_S N_c$ chromosomes with the assistance of the one - point crossover shown graphically in Figure 4. For every couple (z_1, z_2) of created offsprings two chromosomes should be chosen from the current population using tournament selection.
  3. **Mutation procedure**: For every element of each chromosome a random number $r \leq 1$ is selected. The corresponding element is altered randomly when $r \leq p_m$
- **Step 4 - Termination check step**.
  1. **Set** $k = k + 1$
  2. **If** $k < N_g$ goto Fitness Calculation Step.

Figure 3. The BNF grammar used in the method that produces rules for classification and data fitting problems.

```
<S>::= <ifexpr> value=<expr> else value=<expr>  (0)
<ifexpr>::= if(<bexpr>) value=<expr> (0)
            |<ifexpr> else if(<bexpr>) value=<expr> (1)
<bexpr>::=<expr> <rop> <expr> (0)
            |<bexpr> <bop> <bexpr> (1)
<rop>::= > (0)
           |>= (1)
           |<  (2)
           |<= (3)
           |=  (4)
           |!= (5)
<bop>::=  & (0)
           | | (1)
<expr> ::=  (<expr> <op> <expr>)  (0)
           | <func> ( <expr> )    (1)
           |<term>              (2)
<op> ::=     +      (0)
           | -      (1)
           | *      (2)
           | /      (3)
<func> ::=   sin  (0)
           | cos  (1)
           |exp   (2)
           |log   (3)
<term>::=<xlist>                  (0)
           |<dlist>.<dlist> (1)
<xlist>::=x1    (0)
           | x2 (1)
           .........
           | xD (D-1)
<dlist>::=<digit>                  (0)
            | <digit><dlist>       (1)
<digit>  ::= 0 (0)| 1 (1)
           | 2 (2)| 3 (3)
           | 4 (4)| 5 (5)
           | 6 (6)| 7 (7)
           | 8 (8)| 9 (9)
```

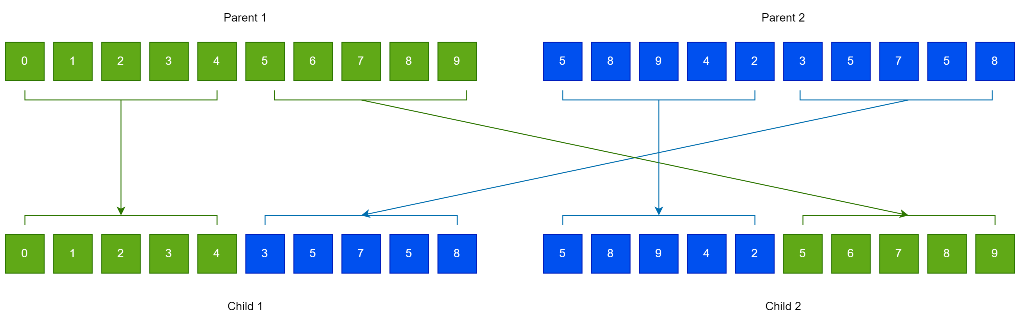

Figure 4. An example of the application of the one point crossover method in the Grammatical Evolution.

2.4. Constructed neural networks

Another method used in the conducted experiments is the neural network construction method [51]. This method can discover the optimal architecture of artificial neural networks as well as an estimation for the parameters of the network by using the Grammatical Evolution technique. This method has been applied in a series of problems, such as problems presented in chemistry [52], education problems [53], autism screening [54] etc. The BNF grammar incorporated in the neural construction procedure is depicted in Figure 5. This grammar can produce artificial neural networks in the following form:

$$N(\overrightarrow{x}, \overrightarrow{w}) = \sum_{i=1}^{H} w_{(d+2)i-(d+1)}\sigma\left(\sum_{j=1}^{d} x_j w_{(d+2)i-(d+1)+j} + w_{(d+2)i}\right) \quad (1)$$

The value H denotes the number of processing units (weights) of the neural network. Also, the function $\sigma(x)$ stands for the sigmoid function. Following the previous equation, it is deducted that the total number of parameters for this network are computed as follows:

$$n = (d+2)H \quad (2)$$

As an example consider the following neural network:

$$N(x) = 1.9\text{sig}(10.5x_1 + 3.2x_3 + 1.4) + 2.1\text{sig}(2.2x_2 - 3.3x_3 + 3.2) \quad (3)$$

This expression represents a neural network used in a problem with 3 inputs (x_1, x_2, x_3).The number of processing nodes is $H = 2$. This neural network is outlined graphically in Figure 6.

```
S:=<Sigval>                            (0)
<Sigval>::=<Node>                      (0)
           | <Node> + <Sigval>         (1)
<Node>::=<Number>*sig(<Sum>+<Number>)  (0)
<Sum>::= <Number>*<Xlist>              (0)
           |    <Sum>+<Sum>             (1)
<Xlist>::= x1         (0)
             |    x2  (1)
             ..............
             |    xd  (d-1)
<Number>::= (<Dlist>.<Dlist>)          (0)
             |    (-<Dlist>.<Dlist>)   (1)
<Dlist>::= <Digit>              (0)
             | <Digit><Dlist>  (1)
<Digit>::= 0        (0)
             |  1  (1)
             ...........
             |  9  (9)
```

Figure 5. The proposed grammar for the construction of artificial neural networks through Grammatical Evolution.

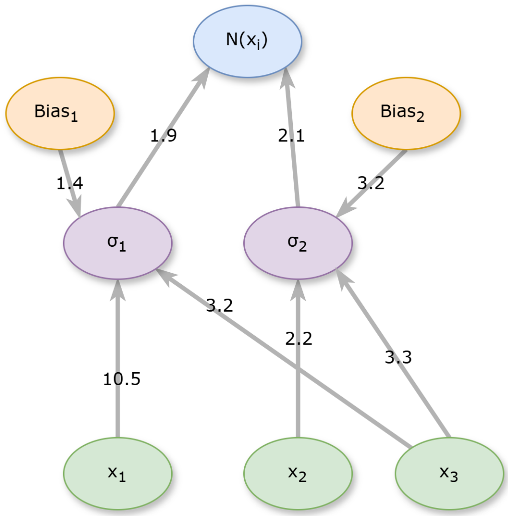

Figure 6. An example of a produced neural network.

The steps of the method used to construct artificial neural networks are shown below:

- **Step 1 - Initialization step**.
  1. **Define** as N_c the number of chromosomes in the genetic population and as N_g the total number of allowed generations.
  2. **Set** as p_s the used selection rate and as p_m the used mutation rate.
  3. **Initialize** the chromosomes $c_i,\ i = 1,\ldots,N_c$. Each chromosome is considered as a set of randomly selected positive integers.
  4. **Set** $k = 0$, as the generation counter.
- **Step 2 - Fitness calculation step**.
  1. **For** $i = 1,\ldots,N_c$ **do**
     (a) **Obtain** the chromosome c_i
     (b) **Create** the corresponding neural network $N_i(\overrightarrow{x},\overrightarrow{w})$ for this chromosome using the grammar of Figure 5.
     (c) **Calculate** the associated fitness value f_i as the training error of network $N_i(\overrightarrow{x},\overrightarrow{w})$ defined as:

$$f_i = \sum_{j=1}^{M}\left(N_i\left(\overrightarrow{x_j},\overrightarrow{w}\right) - y_j\right)^2$$

  2. **End For**
- **Step 3 - Application of genetic operations**. Apply the same genetic operations as in the algorithm of subsection 2.3.
- **Step 4 - Termination check step**.
  1. **Set** $k = k + 1$
  2. **Terminate** if $k \geq N_g$
  3. **Go to** fitness calculation step.

2.5. The feature construction method

Another technique, based on Grammatical Evolution, used on the conducted experiments is the Feature Construction technique, initially described in the work of Gavrilis et al [55]. This technique produces artificial features from the original ones, using non linear mappings produced by the Grammatical Evolution procedure. The BNF grammar used for this technique is outlined in Figure 7.

Figure 7. The grammar used in Feature Construction method.

```
S::=<expr>   (0)
<expr> ::=  (<expr> <op> <expr>)
           | <func> ( <expr> )
           |<term>
<op> ::=     +
           | -
           | *
           | /
<func> ::=   sin
           | cos
           |exp
           |log
<term>::=<xlist>
           |<dlist>.<dlist>
<xlist>::=x1
           | x2
           .........
           | xd
<dlist>::=<digit>
           | <digit><digit>
           | <digit><digit><digit>
<digit>  ::= 0 | 1 | 2 | 3| 4| 5| 6| 7| 8| 9
```

The produced features are evaluated by Radial Basis Function (RBF) networks [56,57] due to the speed of their associated training procedure. The steps of the feature construction technique have as follows:

1. **Initialization step**.
   (a) **Set** the parameters of the method: N_c the number of chromosomes, N_g the maximum number of allowed generations, p_s the selection rate and p_m the mutation rate.
   (b) **Initialize** the N_c chromosomes as sets of random integers.
   (c) **Set** as N_f the number of constructed features.
   (d) **Set** $k = 0$, the generation counter.
2. **Fitness calculation step**.
   (a) **For** $i = 1, \ldots, N_c$ **do**
      i. **Produce** N_f artificial features $y_1, y_2, \ldots, y_{N_f}$ for the processed chromosome c_i using the grammar of Figure 7.
      ii. **Modify** the original train set using the constructed features $y_1, y_2, \ldots, y_{N_f}$.
      iii. **Apply** a machine learning model denoted as $M(x)$ to the modified set and define as the fitness value f_i the training error of $M(x)$.
   (b) **End For**
3. **Genetic operations step**. Apply the same genetic operations as in the algorithm of subsection 2.3.
4. **Termination check step**.
   (a) **Set** $k = k + 1$
   (b) **Terminate** when $k \geq N_g$
   (c) **Go to** fitness calculation step.

3. Experimental results

The measurements from the years 2004, 2010 and 2012 was uses as test cases in the experimental results. For the neural network construction technique the freely available software NNC [59] was incorporated. The feature construction was performed by using the QFC software [60], that was also freely available. Furthermore, the WEKA programming tool [58] was also utilized in the conducted experiments. The validation of the experiments was performed using the ten - fold cross validation technique and all experiments were conducted on a machine running Debian Linux with 128GB of RAM. The average regression error on the test set was measured, using the following equation:

$$E_R(M(\overrightarrow{x})) = \frac{\sum_{i=1}^{N}\left(N(\overrightarrow{x_i}) - y_i\right)^2}{N} \tag{4}$$

Here the test set T is defined as the set $T = (x_i, y_i),\ i = 1, \ldots, N$ and the equation $M(\overrightarrow{x})$ denotes the application of the machine learning $M(x)$ on the input pattern $\overrightarrow{x}$. The values for the experimental parameters are outlined in Table 4. The parameters used in the individual Genetic Algorithms have been successfully used in the past in a multitude of research works and furthermore constitute a compromise between the speed and performance of the algorithms used in the present work.

Table 4. The values used for the experimental parameters.

PARAMETER	MEANING	VALUE
N_c	Chromosomes	500
N_g	Maximum number of generations	200
p_s	Selection rate	0.10
p_m	Mutation rate	0.05
N_f	Number of created features	2
H	Weights	10

Also, The following notation is used in the experimental table:

1. The column year denotes the recording year for the earthquakes.
2. The column D_c denotes the critical distance, expressed as mile, used in the pre - processing of the earthquake data.
3. The column LSTM denotes the incorporation of the Long short - term memory (LSTM) neural network [61] as implemented in the PyTorch programming library [62].
4. The column SVM stands for the usage of the Support Vector Machines [63] as coded in the LibSvm library [64].
5. The column MLP(BP) denotes the incorporation of the Back Propagation algorithm [65,66] in the training of a neural network with $H = 10$ processing nodes.
6. The column MLP(RPROP) represents the usage of the RPROP method [67,68] for the training of a neural network with $H = 10$ processing nodes.
7. The column MLP(BFGS) denotes the usage of the BFGS optimization procedure [69] for the training of an artificial neural network with $H = 10$ processing nodes.
8. The column RULE denotes the incorporation of the rule construction method, described in subsection 2.3.
9. The column NNC represents the usage of the Neural Network Construction method, provided in subsection 2.4.
10. The column FC stands for the usage of the Feature Construction technique, outlined in subsection 2.5.
11. The row AVERAGE stands for the average error for all years and critical distances.

Table 5. Experimental results using a series of machine learning methods.

YEAR	D_c	LSTM	SVM	MLP(BP)	MLP(RPROP)	MLP(BFGS)	RULE	NNC	FC
2004	10	0.25	0.29	0.44	0.24	0.82	0.16	0.16	0.17
2004	25	0.23	0.29	0.42	0.24	0.98	0.17	0.17	0.17
2004	50	0.24	0.29	0.43	0.24	0.87	0.17	0.17	0.16
2004	100	0.23	0.28	0.35	0.22	0.69	0.16	0.16	0.16
2004	500	0.24	0.26	0.45	0.27	0.65	0.16	0.16	0.16
2010	10	0.24	0.30	0.36	0.24	0.74	0.19	0.17	0.19
2010	25	0.24	0.30	0.37	0.21	0.49	0.19	0.18	0.17
2010	50	0.23	0.30	0.39	0.24	0.60	0.18	0.17	0.18
2010	100	0.24	0.30	0.31	0.27	0.40	0.19	0.18	0.19
2010	500	0.25	0.29	0.40	0.32	0.51	0.19	0.18	0.18
2012	10	0.21	0.28	0.33	0.22	0.45	0.18	0.17	0.19
2012	25	0.24	0.28	0.33	0.24	0.75	0.17	0.17	0.16
2012	50	0.22	0.27	0.36	0.23	0.21	0.17	0.17	0.16
2012	100	0.23	0.26	0.38	0.21	0.57	0.17	0.17	0.16
2012	500	0.22	0.25	0.35	0.29	0.85	0.17	0.17	0.16
AVERAGE		**0.234**	**0.283**	**0.378**	**0.245**	**0.639**	**0.175**	**0.170**	**0.171**

An example of constructed features for the year 2010 is presented below:

$$\begin{array}{rcl} f_1(x) & = & \frac{-4}{99.35}x_2 + \cos(x_3 - 877.38x_6 + \sin(43.56x_4 + x_6 - 99.35x_2)) \\ f_2(x) & = & 2x_{13} - 95.2x_2 + 948.94x_1 \end{array}$$

According to the data from the 5 for seismic magnitude prediction (2004) based on neighboring earthquakes, the following key observations are made: The grammatical evolution models (RULE, NNC, FC) continuously exhibit the lowest regression error (0.16–0.17) at all distances (D_c). They are much more stable and accurate than the other models, with an average error of 0.164. This suggests that grammatical evolution significantly improves performance. Among the models without grammatical evolution, LSTM and MLP-RPROP have the best performance with an average error around 0.24 and demonstrate decent stability at different distances. SVM has a slightly higher average error (0.282) but is also relatively stable. MLP-BP has a higher average error (0.418), with some improvement at greater distances (e.g., 0.35 at 100 miles). MLP-BFGS exhibits the worst error (average 0.802) and very high variance, particularly at short distances (0.98 at 25 miles), indicating significant instability. Distance (Dc) seems to minimally affect the error for most models (LSTM, SVM, MLP-RPROP, and the grammatical models), suggesting robustness. The significant exception is MLP-BFGS, which shows marked sensitivity to distance. In summary, the grammatical evolution models (RULE, NNC, FC) are clearly the most accurate and reliable for this prediction problem. Among the rest, LSTM and MLP-RPROP are the best choices due to balanced performance, while MLP-BFGS appears unsuitable due to high and unpredictable error.

For the year 2010, the grammatical evolution models (RULE, NNC, FC) maintain the lowest regression error (0.17–0.19) at all distances, with NNC achieving the best average (0.176). They are more stable than most non-evolutionary models, confirming that grammatical evolution provides reliable predictions. Among the other models, LSTM shows the best performance (average error 0.24) with remarkable stability at all distances. MLP-RPROP has a slightly higher average error (0.256) but with greater variance, particularly at long distances (0.32 at 500 miles). SVM remains stable (average 0.298) with no significant fluctuations. MLP-BP improves at medium distances (0.31 at 100 miles) but has a higher average error (0.366). MLP-BFGS improves compared to 2004 (average error 0.548) but still exhibits significant variance and high error at short distances (0.74 at 10 miles). Distance

(Dc) minimally affects most models, with the exception of MLP-BFGS and, to a lesser extent, MLP-RPROP, which show sensitivity. Overall, the grammatical models (especially NNC) and LSTM stand out as the most reliable approaches for this prediction problem.

Similarly, in 2012, the grammatical evolution models (RULE, NNC, FC) continue to maintain the lowest and most stable regression error at all distances. NNC and FC achieve the best average error (0.17 and 0.166 respectively), with FC demonstrating remarkable stability (0.16) at most distances. RULE has a slightly higher average (0.172). This confirms the superiority of grammatical evolution for this problem. Among the non-evolutionary models, LSTM stands out with the best average error (0.224) and very good stability at different distances. MLP-RPROP has a similar average (0.238) but greater variance, particularly at long distances (0.29 at 500 miles). SVM improves significantly compared to previous years (average error 0.268) and is stable. MLP-BP has the highest error (0.35) among non-evolutionary models, with increasing error at medium distances (0.38 at 100 miles). MLP-BFGS continues to have the highest average error (0.566) and explosive variance, ranging from extremely low error (0.21 at 50 miles) to very high (0.85 at 500 miles), making it unreliable. Distance (Dc) minimally affects the grammatical models, LSTM, and SVM. MLP-BP shows sensitivity at medium distances, while MLP-RPROP and particularly MLP-BFGS show strong sensitivity at specific distances. Overall, for 2012, the grammatical models (especially FC and NNC) and LSTM stand out as the most accurate and stable approaches. MLP-BFGS remains unsuitable due to unpredictable performance, while MLP-BP has the highest error among traditional models. There is a general improvement in SVM and stability in LSTM compared to previous years.

In Figure 8, the grammatical models (RULE, NNC, FC) exhibit the lowest and most stable error (0.16–0.17) across all distances, clearly outperforming the others. LSTM and MLP-RPROP are the best non-evolutionary models (average error ~0.24), while MLP-BFGS shows significant instability (average 0.802) with particularly high errors at short distances.

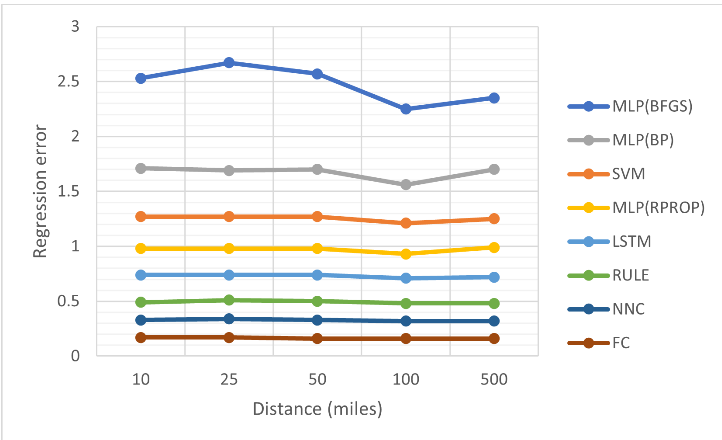

Figure 8. Error per point for each used method using the 2004 dataset and different values of the critical parameter D_c

In Figure 9, the grammatical models maintain their superiority with an average error of 0.176–0.188, with NNC standing out. LSTM continues to be the top non-evolutionary model (average 0.24) with remarkable stability, while MLP-BFGS improves compared to 2004 (average 0.548) but retains significant variation, especially at short distances.

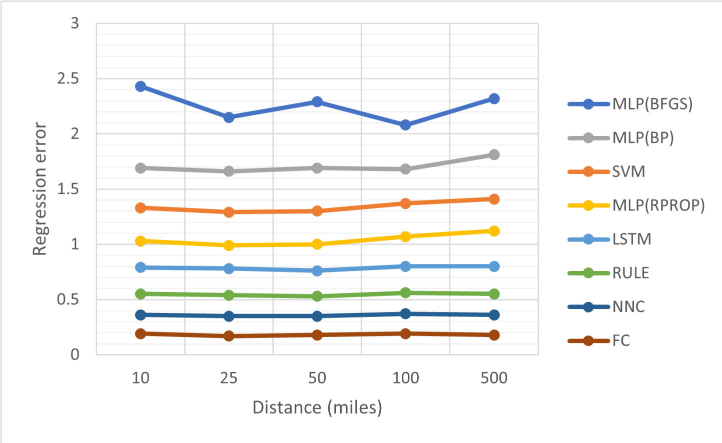

Figure 9. Error per point for each used method using the 2010 dataset and a variety of values of the critical parameter D_c

In Figure 10, the grammatical models achieve their best performance, with FC (average 0.166) and NNC (0.17) demonstrating exceptional stability. LSTM remains the most reliable non-evolutionary model (average 0.224), while MLP-BFGS shows extreme variation (from 0.21 to 0.85) despite its reduced average error (0.566).

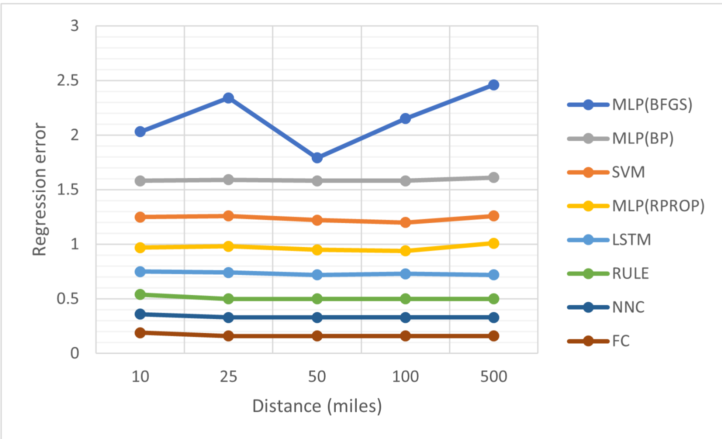

Figure 10. Error per point the machine learning methods using the 2012 and a series of values for the critical parameter D_c

According to the results of the Friedman test 11, where the overall difference between the models is extremely statistically significant (p=4.51e-18 < 0.0001, critical value=4.2863, critical difference=3.834), the following key stratifications are observed: The models with Grammatical Evolution (RULE, NNC, FC) show homogeneous performance (p=ns among them) but are significantly better (p=****) than MLP(BP) and MLP(BFGS). They also significantly outperform SVM (p=**** for NNC/FC, p=*** for RULE). LSTM (without Grammar-Based Evolution) does not differ significantly either from the models with Grammar-Based Evolution (p=ns) or from SVM, MLP(BP), and MLP(RPROP) (p=ns), with the only significant difference being against MLP(BFGS) (p=**). MLP(RPROP) (without Grammar-Based Evolution) does not differ from RULE (p=ns) but shows significantly higher error than NNC/FC (p=*) and MLP(BFGS) (p=*). MLP(BFGS) (without Grammar-Based Evolution) is the least reliable model, with significantly worse performance (p=***) compared to all models with Grammar-Based Evolution and significantly worse than LSTM (p=**) and MLP(RPROP) (p=*).

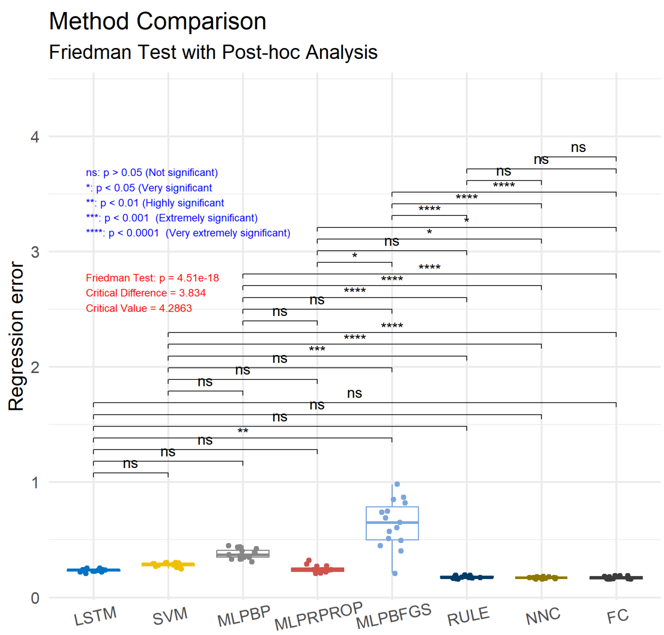

Figure 11. Method comparison using Friedman test.

In Figures 12 and 13, the prediction error analysis results for different machine learning models are presented. We observe significant differences between the two model categories. The analysis reveals that models without grammatical evolution exhibit greater variability in prediction errors. Notably, one of these models shows particularly high errors at certain distances, indicating a lack of stability. In contrast, the most effective model in this category maintains relatively low errors across all distances. The methods with grammatical evolution demonstrate remarkable stability and lower errors. The three models in this category show similar and consistent performance, with minimal variations across different distances and years. The effect of distance is clearly visible in models without grammatical evolution, where errors vary significantly. Conversely, in models with grammatical evolution, distance does not appear to significantly affect prediction accuracy. Temporally, while models without grammatical evolution show some improvement over time, models with grammatical evolution maintain stable performance throughout the study period. Overall, the results support the superiority of methods with grammatical evolution, which demonstrate greater reliability and stability under various conditions. This ability to effectively handle data variations makes them an ideal choice for applications requiring accurate and stable predictions.

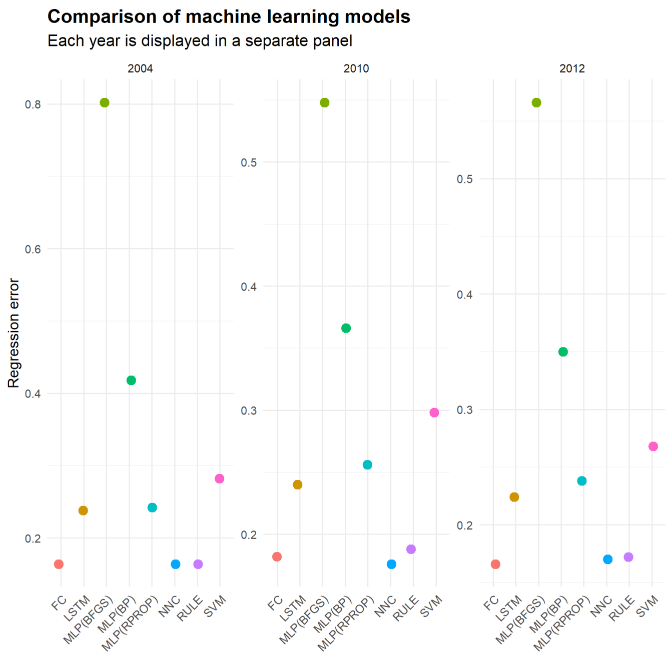

Figure 12. Comparison of machine learning models for every year participated in the experiments.

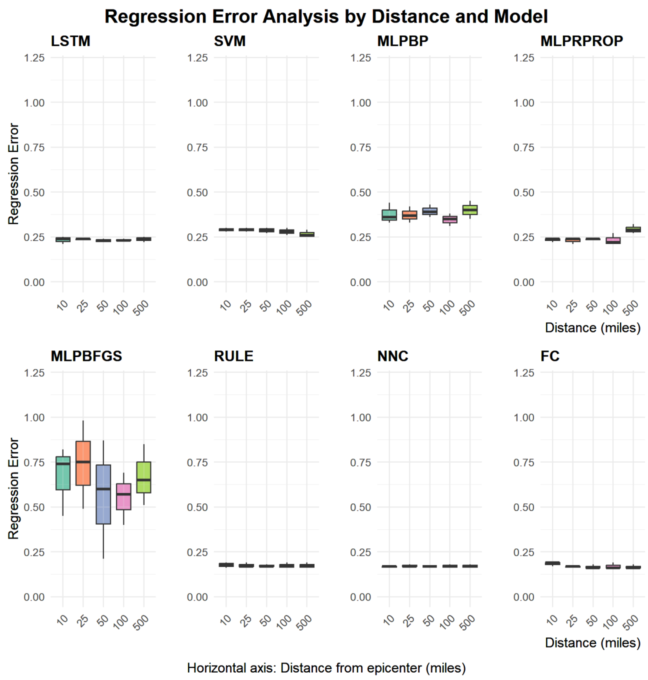

Figure 13. Regression error by distance and model.

To verify the stability of the application of Grammatical Evolution to the present seismic data, an additional experiment was performed where random noise in the range of 0.1%-5% was applied to the problem features one by one. The neural construction method guided by the Grammatical Evolution was incorporated for the data with the noise and the average regression error as measured on the test is shown in Table 6. The noise was applied in the results of the year 2010, where the critical distance D_c was set to 10 miles.

Table 6. Experimental results for the year 2010 and $D_c = 10$ using the Neural Construction technique. A random noise ranging from 0.001 to 0.05 was applied to the data. The bold notation is used to indicate the higher test error as measured on the corresponding test set.

	NOISE PERCENT				
FEATURE	0.1%	0.05%	1%	2%	5%
1	0.176	**0.228**	0.175	0.175	0.175
2	**0.177**	0.175	0.175	0.172	0.175
3	0.175	**0.176**	0.175	0.175	0.175
4	**0.175**	0.175	0.175	0.175	0.175
5	**0.175**	0.175	0.175	0.175	0.175
6	0.175	**0.176**	0.175	0.175	0.175
7	0.175	0.175	0.175	**0.193**	0.175
8	**0.175**	0.175	0.175	0.175	0.175
9	**0.175**	0.175	0.175	0.175	0.175
10	**0.175**	0.175	0.175	0.175	0.175
11	0.175	0.175	0.175	0.175	**0.176**
12	0.176	0.175	0.175	0.175	**0.176**
13	0.175	0.175	0.176	**0.176**	0.175
14	0.176	0.175	0.176	**0.176**	0.175
15	0.175	0.175	**0.176**	0.175	0.175

As one can see from the above experimental results, only in a few cases of noise and only in a limited number of features did a partial deviation in the error appear compared to the case where there is no noise in the data.

4. Conclusions

The article presents an innovative approach to predicting earthquake magnitudes using the Grammatical Evolution method, which offers significant advantages over traditional machine learning techniques. Experimental results for the years 2004, 2010, and 2012 demonstrate that models employing grammatical evolution (RULE, NNC, FC) consistently achieve lower and more stable regression errors compared to models without grammatical evolution (MLP(BP), MLP(RPROP), MLP(BFGS)). The superiority of these models is evident both in reducing mean error and in their resilience to changes in distance, indicating better adaptability to varying spatial conditions. The study's conclusions emphasize the capability of grammatical evolution to effectively identify and exploit structures and relationships in seismic data. Despite the observed improvement in non-grammatical evolution models over time, their performance remains inferior to that of models incorporating grammatical evolution. This demonstrates that using grammatical evolution provides more reliable predictions, a critical factor for applications such as earthquake early warning systems.

As future directions, the study suggests exploring the exact mechanisms through which grammatical evolution enhances performance, focusing on analyzing the structure of the generated rules. Additionally, the scalability of the method to larger and more complex datasets could be investigated, as well as its comparison with other advanced machine learning techniques. Another avenue could involve optimizing the method's hyperparameters and incorporating additional geophysical parameters for even more accurate predictions. Finally, the real-time application of the model and its integration into early warning frameworks could be significant steps toward the practical utilization of this research's findings.

Furthermore, as previously mentioned in the introduction, several developments of various models have focused on short-term early warning systems, typically offering a lead time of approximately one minute, based on real-time detection of tectonic or lithospheric plate movement as it occurs. In contrast, our study not only targets seismic events of magnitude 5.0 and above, but also takes a step further by demonstrating the potential to predict such events before any plate movement is detected. Consequently, our model can potentially be integrated into a broader early warning framework capable of delivering mid to long-term predictions of earthquake occurrences, ranging from hours or days to even months in advance, thereby offering a significantly enhanced preparedness window, and risk mitigation efforts.

Author Contributions: C.K., V.C. and I.G.T. conceived of the idea and the methodology, and C.K. and V.C. implemented the corresponding software. C.K. conducted the experiments, employing objective functions as test cases, and provided the comparative experiments. V.C. performed the necessary statistical tests. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been financed by the European Union : Next Generation EU through the Program Greece 2.0 National Recovery and Resilience Plan , under the call RESEARCH – CREATE – INNOVATE, project name “iCREW: Intelligent small craft simulator for advanced crew training using Virtual Reality techniques" (project code:TAEDK-06195).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Seismograph. Britannica, Earth Sciences. Available from: https://www.britannica.com/science/seismograph(accessed on 20 May 2025).
2. Robert Mallet. Britannica, Civil Engineering. Available from: https://www.britannica.com/biography/Robert-Mallet (accessed on 20 May 2025).
3. Can you predict earthquakes? United States Geological Survey. Available from: https://www.usgs.gov/faqs/can-you-predict-earthquakes (accessed on 20 May 2025).
4. Bergen, K. J., Chen, T., & Li, Z. (2019). Preface to the focus section on machine learning in seismology. Seismological Research Letters, 90(2A), 477-480.
5. How machine learning might unlock earthquake prediction. MIT Technology Review, article by: Allie Hutchison. Published on: December 29, 2023. Available from: https://www.technologyreview.com/2023/12/29/1084699/machine-learning-earthquake-prediction-ai-artificial-intelligence/ (accessed on 20 May 2025).
6. Hincks, T., Aspinall, W., Cooke, R., & Gernon, T. (2018). Oklahoma’s induced seismicity strongly linked to wastewater injection depth. Science, 359(6381), 1251-1255.
7. Obara, K. (2002). Nonvolcanic deep tremor associated with subduction in southwest Japan. Science, 296(5573), 1679-1681.
8. Obara, K., & Kato, A. (2016). Connecting slow earthquakes to huge earthquakes. Science, 353(6296), 253-257.
9. Bletery, Q., & Nocquet, J. M. (2023). The precursory phase of large earthquakes. Science, 381(6655), 297-301.
10. The secret knowledge of animals. Max Planck Institute of Animal Behavior. Published on: April 12 2024. Available from: https://www.ab.mpg.de/578863/news_publication_21821500_transferred?c=413930 (accessed on 20 May 2025).
11. Mousavi, S. M., & Beroza, G. C. (2020). A machine-learning approach for earthquake magnitude estimation. Geophysical Research Letters, 47(1), e2019GL085976.
12. DeVries, P. M., Viégas, F., Wattenberg, M., & Meade, B. J. (2018). Deep learning of aftershock patterns following large earthquakes. Nature, 560(7720), 632-634.
13. Ross, Z. E., Meier, M. A., & Hauksson, E. (2018). P wave arrival picking and first-motion polarity determination with deep learning. Journal of Geophysical Research: Solid Earth, 123(6), 5120-5129.
14. Rouet-Leduc, B., Hulbert, C., Lubbers, N., Barros, K., Humphreys, C. J., & Johnson, P. A. (2017). Machine learning predicts laboratory earthquakes. Geophysical Research Letters, 44(18), 9276-9282.
15. Rouet-Leduc, B., Hulbert, C., & Johnson, P. A. (2019). Continuous chatter of the Cascadia subduction zone revealed by machine learning. Nature Geoscience, 12(1), 75-79.
16. Tan, Y. J., Waldhauser, F., Ellsworth, W. L., Zhang, M., Zhu, W., Michele, M., ... & Segou, M. (2021). Machine-learning-based high-resolution earthquake catalog reveals how complex fault structures were activated during the 2016–2017 central Italy sequence. The Seismic Record, 1(1), 11-19.

17. Mousavi, S. M., Ellsworth, W. L., Zhu, W., Chuang, L. Y., & Beroza, G. C. (2020). Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. Nature communications, 11(1), 3952.
18. Hsu, Y. F., Zaliapin, I., & Ben-Zion, Y. (2024). Informative modes of seismicity in nearest-neighbor earthquake proximities. Journal of Geophysical Research: Solid Earth, 129(3), e2023JB027826.
19. Bayliss, K., Naylor, M., & Main, I. G. (2019). Probabilistic identification of earthquake clusters using rescaled nearest neighbour distance networks. Geophysical Journal International, 217(1), 487-503.
20. Saad, O. M., Chen, Y., Savvaidis, A., Fomel, S., Jiang, X., Huang, D., ... & Chen, Y. (2023). Earthquake forecasting using big data and artificial intelligence: A 30-week real-time case study in China. Bulletin of the Seismological Society of America, 113(6), 2461-2478.
21. Asim, K. M., Martínez-Álvarez, F., Basit, A., & Iqbal, T. (2017). Earthquake magnitude prediction in Hindukush region using machine learning techniques. Natural Hazards, 85, 471-486.
22. Corbi, F., Sandri, L., Bedford, J., Funiciello, F., Brizzi, S., Rosenau, M., & Lallemand, S. (2019). Machine learning can predict the timing and size of analog earthquakes. Geophysical Research Letters, 46(3), 1303-1311.
23. Hoque, A., Raj, J., Saha, A., & Bhattacharya, P. (2020). Earthquake magnitude prediction using machine learning technique. In Trends in Computational Intelligence, Security and Internet of Things: Third International Conference, ICCISIoT 2020, Tripura, India, December 29-30, 2020, Proceedings 3 (pp. 37-53). Springer International Publishing.
24. Zhu, C., Cotton, F., Kawase, H., & Nakano, K. (2023). How well can we predict earthquake site response so far? Machine learning vs physics-based modeling. Earthquake Spectra, 39(1), 478-504.
25. Xiong, P., Tong, L., Zhang, K., Shen, X., Battiston, R., Ouzounov, D., ... & Zhou, H. (2021). Towards advancing the earthquake forecasting by machine learning of satellite data. Science of the Total Environment, 771, 145256.
26. Bhatia, M., Ahanger, T. A., & Manocha, A. (2023). Artificial intelligence based real-time earthquake prediction. Engineering Applications of Artificial Intelligence, 120, 105856.
27. Earthquake Early Warning System. Japan Meteorological Agency. Available from:https://www.jma.go.jp/jma/en/Activities/eew.html (accessed on 20 May 2025).
28. Earthquake Early Warning System (Japan). Wikipedia. Available from: https://en.wikipedia.org/wiki/Earthquake_Early_Warning_(Japan) (accessed on 20 May 2025).
29. ShakeAlert. US Geological Survey, Earthquake Hazards Program. Available from: https://earthquake.usgs.gov/data/shakealert/ (accessed on 20 May 2025).
30. ShakeAlert. Earthquake Early Warning (EEW) System. Available from:https://www.shakealert.org/ (accessed on 20 May 2025).
31. Beroza, G. C., Segou, M., & Mostafa Mousavi, S. (2021). Machine learning and earthquake forecasting—next steps. Nature communications, 12(1), 4761.
32. Mousavi, S. M., & Beroza, G. C. (2023). Machine learning in earthquake seismology. Annual Review of Earth and Planetary Sciences, 51(1), 105-129.
33. M. O'Neill, C. Ryan, Grammatical evolution, IEEE Trans. Evol. Comput. **5**,pp. 349–358, 2001.
34. J. W. Backus. The Syntax and Semantics of the Proposed International Algebraic Language of the Zurich ACM-GAMM Conference. Proceedings of the International Conference on Information Processing, UNESCO, 1959, pp.125-132.
35. C. Ryan, J. Collins, M. O'Neill, Grammatical evolution: Evolving programs for an arbitrary language. In: Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T.C. (eds) Genetic Programming. EuroGP 1998. Lecture Notes in Computer Science, vol 1391. Springer, Berlin, Heidelberg, 1998.
36. M. O'Neill, M., C. Ryan, Evolving Multi-line Compilable C Programs. In: Poli, R., Nordin, P., Langdon, W.B., Fogarty, T.C. (eds) Genetic Programming. EuroGP 1999. Lecture Notes in Computer Science, vol 1598. Springer, Berlin, Heidelberg, 1999.
37. A.O. Puente, R. S. Alfonso, M. A. Moreno, Automatic composition of music by means of grammatical evolution, In: APL '02: Proceedings of the 2002 conference on APL: array processing languages: lore, problems, and applications July 2002 Pages 148–155.
38. E. Galván-López, J.M. Swafford, M. O'Neill, A. Brabazon, Evolving a Ms. PacMan Controller Using Grammatical Evolution. In: , et al. Applications of Evolutionary Computation. EvoApplications 2010. Lecture Notes in Computer Science, vol 6024. Springer, Berlin, Heidelberg, 2010.
39. N. Shaker, M. Nicolau, G. N. Yannakakis, J. Togelius, M. O'Neill, Evolving levels for Super Mario Bros using grammatical evolution, 2012 IEEE Conference on Computational Intelligence and Games (CIG), 2012, pp. 304-31.
40. D. Martínez-Rodríguez, J. M. Colmenar, J. I. Hidalgo, R.J. Villanueva Micó, S. Salcedo-Sanz, Particle swarm grammatical evolution for energy demand estimation, Energy Science and Engineering **8**, pp. 1068-1079, 2020.
41. C. Ryan, M. Kshirsagar, G. Vaidya, G. et al. Design of a cryptographically secure pseudo random number generator with grammatical evolution. Sci Rep **12**, 8602, 2022.
42. C. Martín, D. Quintana, P. Isasi, Grammatical Evolution-based ensembles for algorithmic trading, Applied Soft Computing **84**, 105713, 2019.
43. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.

44. G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems **2**, pp. 303-314, 1989.
45. Lakkos S, Hadjiprocopis A, Compley R, Smith P (1994) A neural network scheme for earthquake prediction based on the Seismic Electric Signals. In: Proceedings of the IEEE conference on neural networks and signal processing, Ermioni, 681-689.
46. Negarestani A, Setayeshi S, Ghannadi-Maragheh M, Akashe B (2002) Layered neural networks based analysis of radon concentration and environmental parameters in earthquake prediction. J. Environmental Radioactivity 62:225-233.
47. Panakkat A, Adeli H (2007) Neural network models for earthquake magnitude prediction using multiple seismicity indicators. Int. J. Neural Systems 17:13-33
48. Holland, J. H. (1992). Genetic algorithms. Scientific american, 267(1), 66-73.
49. Sastry, K., Goldberg, D., & Kendall, G. (2005). Genetic algorithms. Search methodologies: Introductory tutorials in optimization and decision support techniques, 97-125.
50. I.G. Tsoulos, Learning Functions and Classes Using Rules, AI. **3**, pp.751-763, 2022.
51. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.
52. G.V. Papamokos, I.G. Tsoulos, I.N. Demetropoulos, E. Glavas, Location of amide I mode of vibration in computed data utilizing constructed neural networks, Expert Systems with Applications **36**, pp. 12210-12213, 2009.
53. V. Christou, I.G. Tsoulos, V. Loupas, A.T. Tzallas, C. Gogos, P.S. Karvelis, N. Antoniadis, E. Glavas, N. Giannakeas, Performance and early drop prediction for higher education students using machine learning, Expert Systems with Applications **225**, 120079, 2023.
54. E.I. Toki, J. Pange, G. Tatsis, K. Plachouras, I.G. Tsoulos, Utilizing Constructed Neural Networks for Autism Screening, Applied Sciences **14**, 3053, 2024.
55. Dimitris Gavrilis, Ioannis G. Tsoulos, Evangelos Dermatas, Selecting and constructing features using grammatical evolution, Pattern Recognition Letters **29**,pp. 1358-1365, 2008.
56. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation 3, pp. 246-257, 1991.
57. H. Yu, T. Xie, S. Paszczynski, B. M. Wilamowski, Advantages of Radial Basis Function Networks for Dynamic System Design, in IEEE Transactions on Industrial Electronics **58**, pp. 5438-5450, 2011.
58. M. Hall, F. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: an update. ACM SIGKDD explorations newsletter **11**, pp. 10-18, 2009.
59. Tsoulos, I. G., Tzallas, A., & Tsalikakis, D. (2019). NNC: A tool based on Grammatical Evolution for data classification and differential equation solving. SoftwareX, 10, 100297.
60. I.G. Tsoulos, QFC: A Parallel Software Tool for Feature Construction, Based on Grammatical Evolution, Algorithms **15**, 295, 2022.
61. Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. Neural computation, 31(7), 1235-1270.
62. Ketkar, N., Moolayil, J., Ketkar, N., & Moolayil, J. (2021). Introduction to pytorch. Deep learning with python: learn best practices of deep learning models with PyTorch, 27-91.
63. Xue, H., Yang, Q., & Chen, S. (2009). SVM: Support vector machines. In The top ten algorithms in data mining (pp. 51-74). Chapman and Hall/CRC.
64. Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST), 2(3), 1-27.
65. Vora, K., & Yagnik, S. (2014). A survey on backpropagation algorithms for feedforward neural networks. International Journal of Engineering Development and Research, 1(3), 193-197.
66. K. Vora, S. Yagnik, A survey on backpropagation algorithms for feedforward neural networks, International Journal of Engineering Development and Research **1**, pp. 193-197, 2014.
67. Pajchrowski, T., Zawirski, K., & Nowopolski, K. (2014). Neural speed controller trained online by means of modified RPROP algorithm. IEEE transactions on industrial informatics, 11(2), 560-568.
68. Hermanto, R. P. S., & Nugroho, A. (2018). Waiting-time estimation in bank customer queues using RPROP neural networks. Procedia Computer Science, 135, 35-42.
69. M.J.D Powell, A Tolerant Algorithm for Linearly Constrained Optimization Calculations, Mathematical Programming **45**, pp. 547-566, 1989.