Article

Constructing artificial features with Grammatical Evolution for earthquake prediction

Constantina Kopitsa[1], Glykeria Kyrou[2], Vasileios Charilogis[3] and Ioannis G. Tsoulos[4,*]

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

Abstract

Over the course of centuries, humanity has evolved, acquired knowledge, and developed an understanding of the geological phenomenon known as the earthquake. Earthquakes are not the result of the wrath of mythological beings, but rather of the dynamic processes occurring beneath the Earth's crust specifically, the movement and interaction of tectonic / lithospheric plates. When one plate shifts relative to another, stress accumulates and is eventually released as seismic energy. This process is continuous and unstoppable. This phenomenon is well recognized in the Mediterranean region, where significant seismic activity arises from the northward convergence (4–10 mm per year) of the African plate relative to the Eurasian plate along a complex plate boundary. Consequently, our research will focus on the Mediterranean region, specifically examining seismic activity from 1990 - 2015 within the latitude range of 33–44° and longitude range of 17–44°. These geographical coordinates encompass 28 seismic zones, with the most active areas being Turkey and Greece. In this paper we achieved the construction of artificial features for the more effective discrimination of seismic events, utilizing the capabilities offered by Grammatical Evolution. Our results, as will be discussed in greater detail within the research, yield an average error of approximately 9%, corresponding to an overall accuracy of 91%.

Keywords: Earthquakes; Machine learning; Neural networks; Grammatical Evolution; Feature Construction

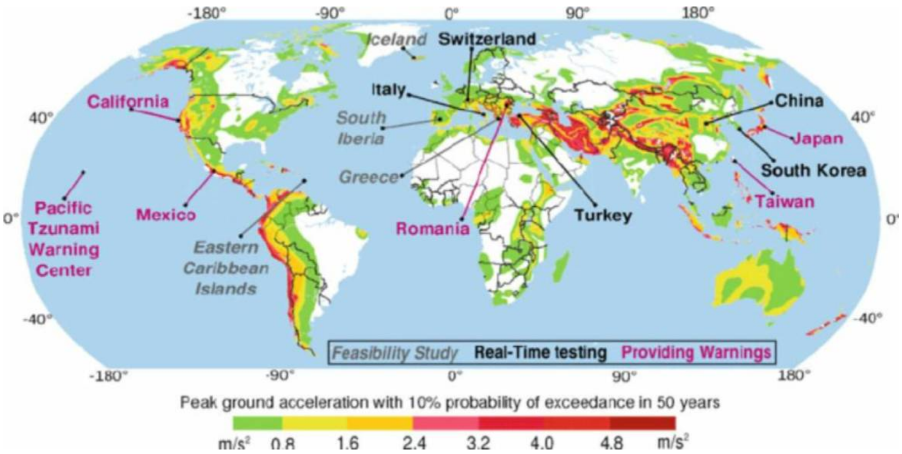
Received:
Revised:
Accepted:
Published:

Citation: Kopitsa, C.; Kyrou, G.; Charilogis, V.; Tsoulos, I.G. . Constructing artificial features with Grammatical Evolution for earthquake prediction. *Journal Not Specified* **2025**, *1*, 0. https://doi.org/

1. Introduction

When entering the single keyword "earthquakes" into Google Scholar, more than 3,910,000 results are retrieved, demonstrating the intense interest that exists in the field of seismology. Thanks to these studies and investigations, which evolve from an initial idea or theory into practical applications, it can be stated with confidence that humanity is now capable of achieving timely early warning before seismic events occur. Consequently, the pursuit of sustainability strengthens our resilience against seismic phenomena. This has been achieved through the implementation of early warning systems established across the globe, particularly in technologically advanced countries that are also seismically vulnerable regions. For this purpose, the UN Disaster Risk Reduction was established, which is "aimed at preventing new and reducing existing disaster risk and managing residual risk, all of which contribute to strengthening resilience and therefore to the achievement of sustainable development "[1]. An early achievement of Disaster Risk Reduction took place in Japan in 1960, when seismic sensors were installed along the railway infrastructure to ensure the automatic immobilization of trains [2]. The Japanese UrEDAS (Urgent Earthquake Detection and Alarm System) has been described as the "grandfather" of earthquake early warning systems in general, and of onsite warning systems in particular [3]. Since then, techniques and methods have advanced through technological progress. The next achievement in early earthquake warning was accomplished in Mexico in 1989 with the establishment of the Seismic Alert System (SAS) [4]. In 2006, Japan launched the Earthquake Early Warning system initially for a limited audience and subsequently for the general public, in order to ensure the effectiveness of EEW in disaster mitigation [5]. This allows an earthquake warning to be disseminated between several seconds and up to one minute prior to the occurrence of the event [6]. In Bucharest, Romania, an earthquake warning system, in 1999, was also developed, providing a preparation window of 25 seconds [7]. Also in Istanbul, in preparation for the anticipated earthquake, an early warning system was implemented in 2002 [8]. In Southern Europe, at the University of Naples in Italy, a software model called PRESTo (PRobabilistic and Evolutionary early warning SysTem) was developed, designed to estimates of earthquake location and size within 5–6 seconds [9]. Furthermore, the United States, through the U.S. Geological Survey, has established its own earthquake warning system. Since 2016, the ShakeAlert system has been operational along the West Coast [10,11]. Subsequently, a map is presented in Figure 1, illustrating Earthquake Early Warning Systems worldwide, with colors indicating the operational status of each system. Purple denotes operative systems that provide warnings to the public, black represents systems currently undergoing real-time testing, and gray is used for countries where feasibility studies are still in progress.

Figure 1. The map shows Earthquakes Early Warning Systems around the World



Within this framework, it is important to highlight that in recent years considerable emphasis has been placed on the advancement of diverse models for the early detection of seismic events, which have become available to the wider public through mobile applications, television broadcasts, and radio communication [12]. The primary function of Android applications is that, once an earthquake is detected, an alert is transmitted to all smartphones located within the affected area. Provided that the user is not in close proximity to the epicenter, the notification can be received in advance, allowing sufficient time to take protective action before the destructive seismic waves arrive [13–16]. By harnessing technology as an ally against natural disasters, humanity can move beyond the devastating consequences of major earthquakes, such as the 2004 Indian Ocean event with more than 220,000 fatalities, the 2011 Tōhoku earthquake in Japan with over 19,000 losses, and the 2023 Turkey–Syria earthquake with more than 43,000 deaths. Accordingly, resilience and sustainability for populations affected by seismic events encompass both physical and social infrastructures capable of withstanding earthquakes, while simultaneously safeguarding long-term well-being through disaster risk reduction, community preparedness, and equitable recovery.

Subsequently, we will proceed with the presentation of related studies alongside our own, which progressively enhance both our sustainability and our capacity for prevention against seismic phenomena. Housner, in 1964, concluded that artificial earthquakes constitute adequate representations of strong-motion events for structural analysis and may serve as standard ground motions in the design of engineering structures [17]. Adeli, in 2009, proposed a novel feature extraction technique, asserting that when combined with a selected Probabilistic Neural Network (PNN), it can yield reliable prediction outcomes for earthquakes with magnitudes ranging from 4.5 to 6.0 on the Richter scale [18]. Zhou, in 2012, introduced a robust feature extraction approach, the Log-Sigmoid Frequency Cepstral Coefficients (LSFCC), derived from the Mel Frequency Cepstral Coefficients (MFCC), for the classification of ground targets using geophones. Employing LSFCCs, the average classification accuracy for tracked and wheeled vehicles exceeds 89% across three distinct geographical settings, achieved with a single classifier trained in only one of these environments [19]. Martinez-Alvarez, in 2013, investigated the utilization of various seismicity indicators as inputs for artificial neural networks. The study proposes combining multiple indicators—previously shown to be effective across different seismic regions—through the application of feature selection techniques [20]. Schmidt, in 2015, proposed an efficient and automated method for seismic feature extraction. The central concept of this approach is to interpret a two-dimensional seismic image as a function defined on the vertices of a carefully constructed underlying graph [21]. Narayanakumar, in 2016, extracted seismic features from a predetermined number of events preceding the main shock in order to perform earthquake prediction using the Backpropagation (BP) neural network technique [22]. Cortes, in 2016, sought to identify the parameters most effective for earthquake prediction. As various studies have employed different feature sets, the optimal selection of features appears to depend on the specific dataset used in constructing the model [23]. Asim, in 2018, developed a hybrid embedded feature selection approach designed to enhance the accuracy of earthquake prediction [24]. Chamberlain, in 2018, demonstrated that synthetic seismograms, when applied with matched-filter techniques, enable the detection of earthquakes even with limited prior knowledge of the source [25]. Okada, in 2018, employed observational data either by calibrating parameters within existing models or by deriving models and indicators directly from the data itself [26]. Lin, in 2018, employed the earthquake catalogue from 2000 to 2010, comprising events with a Richter magnitude (ML) of 5 and a depth of 300 km within the study area (21°–26° N, 119°–123° E). This dataset was utilized as training input to develop the initial earthquake magnitude prediction backpropagation neural network (IEMPBPNN) model, which was designed with two hidden layers [27]. Zhang, in 2019, proposed a precursory pattern-based feature extraction approach aimed at improving earthquake prediction performance. In this method, raw seismic data are initially segmented into fixed daily time intervals, with the magnitude of the largest earthquake within each interval designated as the main shock [28]. Rohas's, in 2019, paper reviewed the latest uses of artificial neural networks for automated seismic-data interpretation, focusing especially on earthquake detection and onset-time estimation [29]. Ali, in 2020, generated synthetic seismic data for a three-layer geological model and analyzed using Continuous Wavelet Transform (CWT) to identify seismic reflections in both the temporal and spatial domains [30]. Bamer, in 2021, demonstrated through comparison with several state-of-the-art studies, that the convolutional neural network autonomously learns to extract the pertinent input features and structural response behavior directly from complete time histories, rather than relying on a predefined set of manually selected intensity measures [31]. Wang, in 2023, reports that the accuracies of various AI models using the feature extraction dataset surpassed those obtained with the spectral amplitude dataset, demonstrating that the feature extraction approach more effectively emphasizes the distinctions among different types of seismic events [32]. Ozkaya, in 2024, developed a novel feature engineering framework that integrates the Butterfly Pattern (BFPat), statistical measures, and wavelet packet decomposition (WPD) functions. The proposed model achieved an accuracy of 99.58% in earthquake detection and 93.13% in three-class wave classification [33]. Sinha's review, in 2025, offers valuable insights into cutting-edge techniques and emerging directions in feature engineering for seismic prediction, highlighting the importance of interdisciplinary collaboration in advancing earthquake forecasting and reducing seismic risk [34]. Mahmoud, in 2025, investigates the application of machine learning approaches to earthquake classification and prediction using synthetic seismic datasets [35].

In contrast to the aforementioned studies, this paper introduces a novel approach constructing artificial features with Grammatical Evolution [36] for earthquake prediction. The method of Grammatical Evolution can be considered as a genetic algorithm [37] with integer chromosomes. Each chromosome contains a series of production rules from the provided Backus-Naur form (BNF) grammar [38] of the underlying language and hence this method can create programs that belong to this language. This procedure have been used with success in various cases, such as data fitting problems [39,40], problems that appear in economics [41], computer security problems [42], problems related to water quality [43], problems appeared in medicine [44], evolutionary computation [45], hardware issues in data centers [46], solution of trigonometric problems [47], automatic composition of music [48], dynamic construction of neural networks [49,50], automatic construction of constant numbers [51], playing video games [52,53], problems regarding energy [54], combinatorial optimization [55], security issues [56], automatic construction of decision trees [57], problems in in electronics [58], automatic construction of bounds for neural networks [59], construction of Radial Basis Function networks [60] etc. This research work focuses on the creation of artificial features from existing ones, aiming at two goals: on the one hand, it seeks to reduce the required information required for the correct classification of seismic data and on the other hand, it seeks to highlight the hidden nonlinear correlations that may exist between the existing features of the objective problem. In this way, a significant improvement in the classification of seismic data will be achieved.

The rest of this manuscript is organized as follows: the used dataset and the incorporated methods used in the conducted experiments are outlined in section 2, the experimental results are shown and discussed in section 3 and finally a detailed discussion is provided in section 4.

2. Materials and Methods

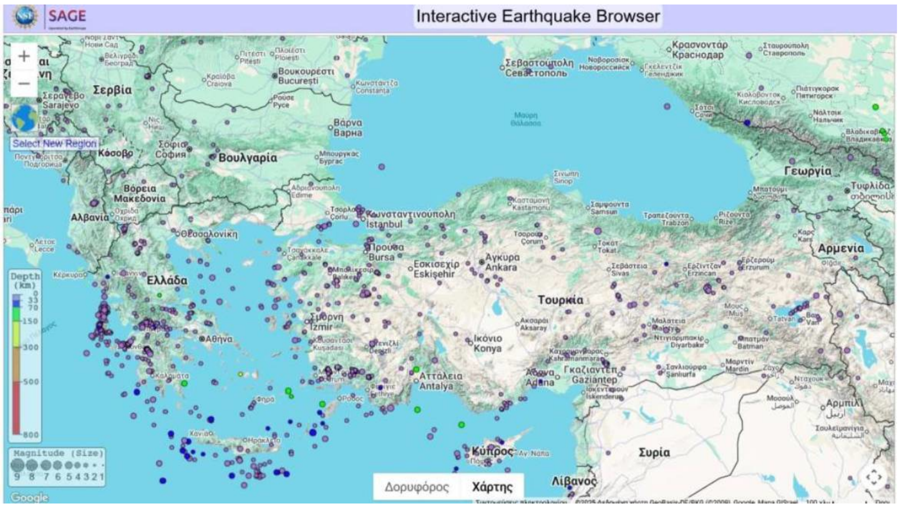
In this section, a detailed presentation of the datasets used as well as the machine learning techniques used in the experiments performed will be provided.

2.1. The Dataset Employed

In this study, we made use of open data from the NSF Seismological Facility for the Earth Consortium (SAGE) available from https://ds.iris.edu/(accessed on 25 November 2025), which is a platform offering an interactive global map that facilitates both data visualization and the real-time extraction of datasets from the displayed geographic regions. The area defined by latitudes 33°–44° and longitudes 17°–44° covers 28 seismic zones, with Turkey and Greece identified as the most seismically active regions. Furthermore, the selection of NSF data was driven by its advanced functionality and broad accessibility.

Notably, it supports the download of up to 25,000 records per file, thereby streamlining the workflow and improving the efficiency of information retrieval. The region under investigation is presented in Figure 2.

Figure 2. The study area.



2.2. Dataset Description

We obtained and systematically analyzed 511,064 earthquake events from 1990 to 2015, as this time period is of particular scientific interest due to the surge in seismic activity, as also illustrated in the graph of Figure 3. Specifically, we selected this time period because it encompasses a wide range of earthquake magnitudes, which provides a diverse dataset conducive to algorithm training and supports the construction of artificial features via Grammatical Evolution. Moreover, our analysis led us to the conclusion that the datasets from 1970–1989 and 2016–2025 lack the diversity observed in the dataset we selected. Specifically, classes 6 and 7 are entirely absent from the 1970–1989 records, while in the post-2016 data class 7 is missing and class 1 contains only a limited number of instances, namely 25.

Figure 3. Graphs seismic events from 1970 - 2025 (Area Study)

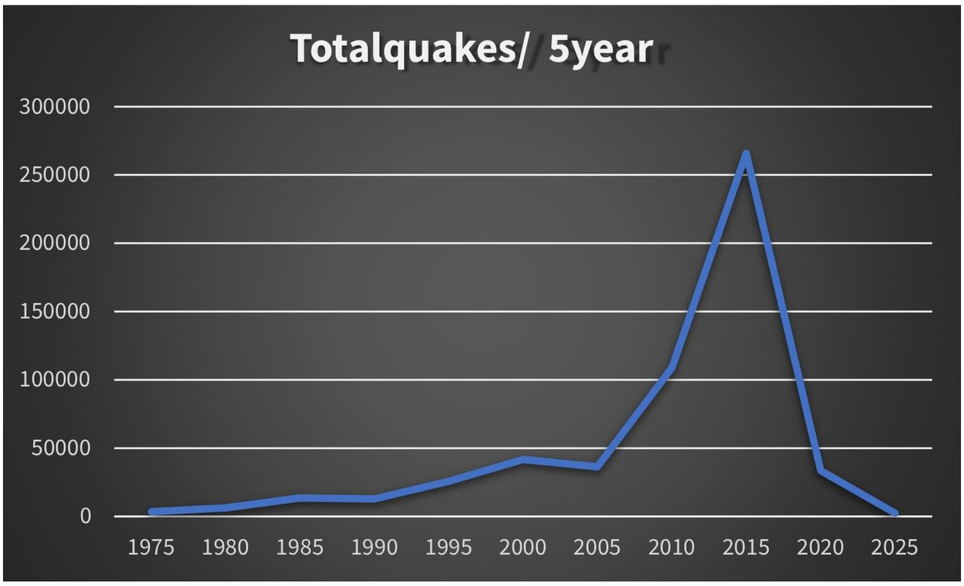


Table 1. Raw Data from NSF Interactive Earthquake Browser (1990 - 2015)

Raw Data	
Features	**Range**
Year	1990 - 2015
Month	1 - 12
Day	1- 31
Time	00:00:00 - 23:59:59
Latitude	33 - 44
Longitude	17 - 44
Region	1 - 28
Depth	0.00 - 800.00
Magnitude	1 - 10
Timestamp	

Table 2. Utilized Data from NSF Interactive Earthquake Browser (1990 - 2015)

Utilized Data		
Features	**Range**	**Class**
Year	1990 - 2015	
Epoch Code	1 - 12	0 -3
Day Code	1 - 31	0 - 2
Time Code	00:00:00 - 23:59:59	0 - 3
Latitude	33 - 44	
Longitude	17 - 44	
Depth Code	0.00 - 800.00	0 - 5
Previous Magnitude Code	1 - 10	1 - 7
Same Region Code	1 - 28	1 - 28
Lithospheric Code	1 - 7	1 - 7
Kp Code	0.000 - 9.000	0 - 5

On the platform that provides us with the Interactive Earthquake Browser we employed coordinates spanning latitudes 33°–44° and longitudes 17°–44°, considered magnitude values from 1.0 to 10.0, and incorporated all available depths by default within the depth range. The raw dataset included the following variables: year, month, day, time, latitude (Lat), longitude (Lon), depth, magnitude, region, and timestamp. Accordingly, Table 1 provides a detailed overview of the raw dataset.

Subsequently, a preprocessing procedure was applied to the dataset. This included the identification of the lithospheric plate associated with each earthquake, to which a unique code was assigned. Furthermore, the months were categorized according to the four seasons, the days were grouped into ten-day intervals, and the time of occurrence was classified into four periods (morning, noon, afternoon, and night). The focal depth was divided into six categories. In addition, a new column was created to indicate, with a binary value (0 or 1), whether an earthquake had previously occurred in the same region during the same season. Finally, the dataset was merged with the Kp index, representing geomagnetic storm activity, which was further classified into six distinct categories. The final dataset was further processed, including the following: Year, Epoch Code, Day Code, Time Code, Latitude, Longitude, Depth Code, Previous Magnitude Code, Same Region Code, Lithospheric/Tectonic Plate, Kp Code. This information is outlined in Table 2.

At the following stage of data processing, we elected to focus on earthquakes with a magnitude code of 2 and above, since the inclusion of lower-magnitude events would bias the model toward predicting minor seismic occurrences. Specifically, the earthquakes were divided into two large classes to optimize performance: the first comprising magnitude codes 2 and 3, and the second encompassing events with magnitudes greater than 3.

Following these steps, we proceeded with our experiments, utilizing approximately 10,000 seismic events in order to generate artificial features through Grammatical Evolution.

2.3. Global optimization methods for neural network training

In the current work two well - known global optimization methods were incorporated for neural network training [61,62], the Genetic Algorithm and the Particle Swarm Optimization (PSO) method. The Genetic algorithms is an evolutionary optimization method designed to minimize an objective function defined over a continuous search space. It operates on a population of candidate solutions, each represented as a vector of parameters and this population is evolved through a series of steps, where in each step some process that resemble natural processes are applied to the population. Genetic algorithms have applied on a wide range applications, that include training of neural networks [64,65]. Also, Algorithm 1 presents the basic steps of a genetic algorithm.

Algorithm 1 The main steps of a Genetic Algorithm..

INPUT
- f: objective function
- N_c: number of chromosomes
- N_g: maximum number of allowed generations
- p_s: the selection rate of the algorithm, with $p_s \leq 1$
- p_m: the mutation rate of the algorithm, with $p_m \leq 1$
- k: the generation counter
- a: uniformly distributed random numbers, in $\in [-0.5, 1.5]$

OUTPUT
-x_{best}, f_{best}

INITIALIZATION
- $k \leftarrow 0$

main pseudocode

while $k < N_g$ **do** // termination check
    for each $g_i, i \in \{1..N_c\}$ **do**
        $f_i \leftarrow f(g_i)$
    endfor
    Sort chromosomes by increasing fitness: $N_b \leftarrow (1 - p_s)xN_c$
    Select parents w, z randomly among the best N_b chromosomes
    Draw $a_i \in U[-0.5, 1.5]$
        $z_i \leftarrow a_i z_i + (1 - a_i) w_i$
        $w_i \leftarrow a_i w_i + (1 - a_i) z_i$
    for each $g_i, i \in \{N_b + 1..N_c\}$ **do**
        Replace g_i with z_i or w_i
    endfor
    for each $g_i, i \in \{1..N_c\}$ **do**
        for each gene, $j \in \{1....n\}$ **do**
            Draw $rU[0, 1]$
                if $r \leq p_m$ then
                    Mutate gene j of g_i
                endif
        endfor
    end for
    for each $g_i, i \in \{1..N_c\}$ **do**
        $f_i \leftarrow f(g_i)$
        Draw $ri \in U[0, 1]$
        if ($f_i < f_{best}$) **then**
            $xbest \leftarrow gi, fbest \leftarrow fi$
        endif
    endfor
    $k \leftarrow k + 1$
endwhile
return x_{best}, f_{best}

Particle Swarm Optimization (PSO) is a population-based search method inspired by how animals move and cooperate in groups like flocks of birds or schools of fish [67,68]. The PSO was widely used in a variety of practical problems as well as in neural network training [69,70]. The main steps of the PSO method are outlined in Algorithm 2.

Algorithm 2 The main steps of the PSO algorithm.

INPUT
- f: objective function
- m: number of particles with $x_i \in S$
- u_i: number of velocities with $u_i \in S$
- x_i: number of positions in Ω
- w: inertia
- $c1$,$c2$: constant numbers
- $r1$,$r2$: random numbers
- $iter$: iteration counter
- $iter_{max}$: max iterations
- p_i: vectors are best located values for every particle i

OUTPUT
-p_{best}

INITIALIZATION
- **Set** $iter \leftarrow 0$
- **Set** positions $x_i \in S, i = 1.....m$
- **Set** velocities $u_i, i = 1.....m$
-**For** each particle $i \in \{1..m\}$ **do**
-$p_i \leftarrow x_i$
-**End for**
-$p_{best} \leftarrow argmin_i f(x_i)$// global best

while $generation < Gmax$ **do**// termination check
    For each $i \in \{1..m\}$ **do**
        draw $r1, r2 \backsim U(0, 1)$
        $u_i \leftarrow wu$+$c1r1(p_i - x_i)$+$c2r2(p_{best} - x_i)$// update velocity
        $x_i \leftarrow x_i + u_i$// update position
        if $f(x_i) < f(p_i)$**then**
            $p_i \leftarrow x_i$
        End if
    End for
        $p_{best} \leftarrow argmin_i f(x_i)$// Update global best
        $iter \leftarrow iter + 1$
End while

2.4. The SVM method

The Support Vector Machine (SVM) is a supervised learning algorithm applied to both classification and regression problems [71]. The concept of Support Vector methods was introduced by V. Vapnik in 1965, in the context of addressing challenges in pattern recognition. During the 1990s, V. Vapnik formally introduced Support Vector Machine (SVM) methods within the framework of Statistical Learning. Since their introduction, SVMs have been extensively employed across diverse domains, including pattern recognition, natural language processing, and related applications [72]. For instance, the SVM algorithm has been employed both in studies on earthquake prediction and in research on the early detection of seismic phenomena, as demonstrated in the following works: Hybrid Technique Using Singular Value Decomposition (SVD) and Support Vector Machine (SVM) Approach for Earthquake Prediction [73], Using Support Vector Machine (SVM) with GPS Ionospheric TEC Estimations to Potentially Predict Earthquake Events [74], The efficacy of support vector machines (SVM) in robust determination of earthquake early warning magnitudes in central Japan [75]. As with other comparative analyses of models, SVMs require greater computational resources during training and exhibit reduced susceptibility to overfitting, whereas neural networks are generally regarded as more adaptable and capable of scaling effectively.

2.5. The neural network construction method

Another method used in the experiments to predict the category of seismic vibrations is the method of constructing artificial neural networks using Grammatical Evolution [76]. This method can detect the optimal architecture for neural networks as well as the optimal set of values for the corresponding parameters. This method was used in various cases, such as chemistry problems [77], solution of differential equations [78], medical problems [79], educational problems [80], detection of autism [81] etc. This method can produce neural networks in the following form:

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

In this equation the value H represents the number of used computation units (weights) for the neural network. The function $\sigma(x)$ stands for the sigmoid function, which is defined as:

$$\sigma(x) = \frac{1}{1+\exp(-x)} \tag{2}$$

2.6. The proposed method

The method proposed here to tackle the classification of seismic events is a procedure that constructs artificial features from the original ones using Grammatical Evolution. The method was initially presented in the paper of Gavrilis et al [82] and used in various cases in the past [83–85]. The main steps of this method have as follows:

1. **Initialization** step.
   (a) **Obtain** the train data and denote them using the M pairs $(x_i, t_i),\ i = 1..M$. The values t_i represent the actual output for the input pattern x_i.
   (b) **Set** the parameters of the used genetic algorithm:N_g for as the number of allowed generations, N_c for the number of chromosomes, p_s for the selection rate and p_m for the mutation rate.
   (c) **Set** as N_f the number of artificial features that will construct the current method.
   (d) **Initialize** every chromosome $g_i, i = 1, \ldots, N_g$ as a set of randomly selected integers.
   (e) **Set** $k = 1$, the generation counter.
2. **Genetic step**
   (a) **For** $i = 1, \ldots, N_g$ **do**
      i. **Construct** with Grammatical Evolution a set of N_f artificial features for the each chromosome g_i. The BNF grammar used for this procedure is shown in Figure 4.
      ii. **Transform** the original set of features using the constructed features and denote the new training set as $\left(x_{g_i,j}, t_j\right),\ j = 1, ..M$
      iii. **Train** a machine learning C on the new training set. The training error for this model will represent the fitness f_i for the current chromosome and it is computed as:

$$f_i = \sum_{j=1}^{M} \left(C\left(x_{g_i,j}\right) - t_j\right)^2 \tag{3}$$

      In the current work the Radial Basis Function networks (RBF) [86,87] were use d as the machine learning models. A decisive factor for this choice was the short training time required for these machine learning models.
      iv. **Perform** the selection procedure: Initially the the chromosomes are sorted according to the associated fitness values and the best $(1-p_s) \times N_C$ of them are copied intact to the next generation. The remaining chromosomes will be replaced by offsprings that will be produced during the crossover and the mutation procedure.

Figure 4. The grammar used for the proposed method.

```
S::=<expr>   (0)
<expr> ::=  (<expr> <op> <expr>)  (0)
           | <func> ( <expr> )    (1)
           |<terminal>            (2)
<op> ::=     +      (0)
           | -      (1)
           | *      (2)
           | /      (3)
<func> ::=   sin  (0)
           | cos  (1)
           |exp   (2)
           |log   (3)
<terminal>::=<xlist>                (0)
           |<digitlist>.<digitlist> (1)
<xlist>::=x1    (0)
           | x2 (1)
           .........
           | xN (N)
<digitlist>::=<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)
```

v. **Perform** the crossover procedure: The outcome of this process is a set of $p_s \times N_c$ new chromosomes. For every pair $(\tilde{z}, \tilde{w})$ of new chromosomes, two distinct chromosomes z and w are chosen from the current population using the process of tournament selection. Afterwards, the nee chromosomes are produced using the procedure of one - point crossover, that is illustrated in in Figure 5.

vi. **Execute** the mutation procedure: during this procedure a random number $r \in [0, 1]$ is selected for each element of every chromosome. The corresponding element is altered randomly if $r \leq p_m$.

(b) **EndFor**

3. **Set** $k = k + 1$
4. **If** $k \leq N_g$ goto **Genetic** Step
5. **Obtain the best chromosome** g^* from the current population.
6. **Create** the corresponding N_f features for g^* and apply this features to the set set and report the corresponding test error.

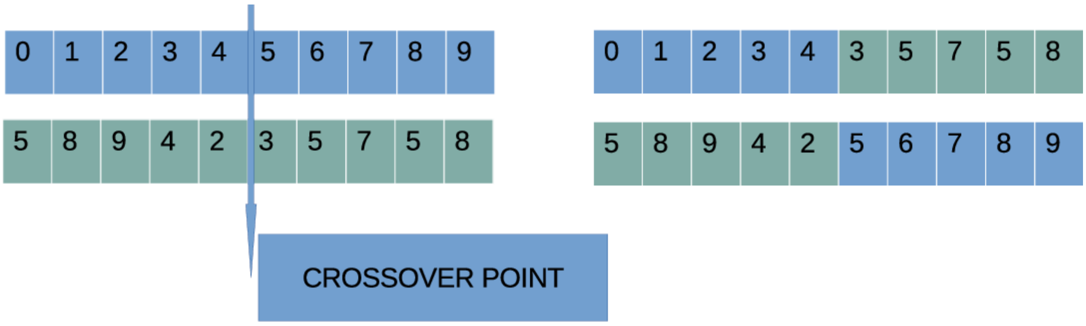


Figure 5. An example of the one - point crossover method.

3. Results

The methods used in the conducted experiments were coded in ANSI C++, with the assistance of the freely available optimization package of OPTIMUS [88]. Each experiment was conducted 30 times, using different seed for the random generator in each execution. Also, the procedure of ten - fold cross validation was incorporated to validate the conducted experiments. The values for the experimental parameters are listed in Table 3.

Table 3. The values of the experimental parameters.

PARAMETER	MEANING	VALUE
N_g	Number of generations	500
N_c	Number of chromosomes	500
p_s	Selection rate	0.9
p_m	Mutation rate	0.05
N_f	Number of features	2

Table 4. Experimental results using a series of machine learning methods.

DATASET	MLP(GEN)	MLP(PSO)	SVM	NNC	FC
GCT1990	39.98%	38.21%	14.30%	22.72%	13.50%
GCT1995	38.08%	33.22%	13.48%	21.50%	10.82%
GCT2000	26.70%	27.48%	10.19%	24.59%	6.52%
GCT2005	38.44%	33.19%	6.85%	27.85%	4.45%
GCT2010	52.89%	44.77%	14.46%	29.81%	9.97%
AVERAGE	**39.22%**	**35.37%**	**11.86%**	**25.29%**	**9.05%**

Table 4 reports the classification error rates for five temporal subsets of the seismic dataset (GCT1990, GCT1995, GCT2000, GCT2005, GCT2010), where the first column encodes the year of the data and the remaining columns correspond to the machine learning models MLP(GEN), MLP(PSO), SVM, NNC, and the proposed FC (Future Constructions) model. The values are expressed as percentage classification error, that is, the proportion of misclassified events between the two seismic classes defined during preprocessing (events with magnitude code 2–3 versus events with magnitude greater than 3). The following notation is used in this table:

1. The column MLP(GEN) denotes the error from the application of the genetic algorithm described in subsection 2.3 for the training of a neural network with 10 processing nodes.
2. The column MLP(PSO) represents the incorporation of the PSO method given in subsection 2.3 for the training of a neural network with 10 processing nodes.

3. The column SVM represent the application of the SVM method, described in subsection 2.4. In the current implementation the freely available library LibSvm [89] was used.
4. The column NNC represents the neural network construction method, described in subsection 2.5.
5. The column FC denotes the proposed feature construction technique, outlined in subsection 2.6.
6. The row AVERAGE denotes the average classification error for all datasets.

The most striking observation is that FC achieves the lowest error for all five datasets, with no exceptions. For every GCT year, the FC column attains the minimum error among all models, demonstrating a consistent and temporally robust superiority. This pattern is clearly reflected in the AVERAGE row: the mean classification error of FC is 9.05%, whereas the corresponding mean error of the best conventional baseline, the SVM, is 11.86%. Moving from 11.86% to 9.05% corresponds to an error reduction of approximately 24-25%, meaning that roughly one quarter of the misclassifications made by the SVM are eliminated when using FC. Compared with the neural approaches, the gain is even more pronounced: FC reduces the mean error from 39.22% to 9.05% for MLP(GEN), from 35.37% to 9.05% for MLP(PSO), and from 25.29% to 9.05% for NNC, effectively absorbing about 64-77% of their errors.

Examining the behaviour per year reveals how the proposed model interacts with the specific characteristics of each temporal subset. Dataset GCT2005 appears to be the "easiest" case for all models, with the SVM dropping to 6.85% error and FC reaching 4.45%, which corresponds to an accuracy of about 95.5%. At the opposite end of the spectrum, GCT2010 is clearly the most challenging dataset, as indicated by substantially increased errors for the conventional models (MLP(GEN) 52.89%, MLP(PSO) 44.77%, NNC 29.81%) and by a higher error for the SVM (14.46%). Even in this difficult scenario, FC keeps the error below 10% (9.97%), preserving a clear qualitative advantage in the most demanding temporal setting. A similar pattern emerges for the intermediate years 1995 and 2000, where FC-SVM absolute differences are in the range of 2-4 percentage points, corresponding to roughly 20-35% relative error reduction. This suggests that as the data become more complex, the benefit of the automatically constructed features generated by Grammatical Evolution becomes increasingly pronounced.

A second important finding is that the ranking of the conventional models remains stable across all years. The SVM is consistently the best among the standard baselines, followed by NNC and the two MLP variants, with MLP(PSO) systematically outperforming MLP(GEN). This stability reinforces the credibility of the table as a benchmark, indicating that the results are not driven by noise or random fluctuations but instead reflect a coherent hierarchy of model performance. On top of this stable baseline, FC does not merely swap the winner for a single dataset, it establishes a new performance level by consistently pushing the error rates to substantially lower values in every year.

From a practical standpoint, given that the final processed dataset contains on the order of 10,000 seismic events, an average error of 11.86% compared with 9.05% implies hundreds fewer misclassified instances when FC is used instead of a plain SVM trained on the original features. This has direct implications for early warning and risk assessment applications, where each reduction in misclassification translates into more reliable decision-making. The evidence from Table 4, combined with the methodological description, strongly supports the view that the feature construction phase based on Grammatical Evolution is not a minor refinement over existing classifiers, but rather a key component that reshapes the feature space so that the seismic classes become much more separable. This explains why the proposed FC model achieves an average error of approximately 9%, in full agreement with the reported overall accuracy of 91% in the abstract.

Overall, Table 6 shows that FC is not only the best-performing model in each individual year but also the most stable across different temporal subsets, confirming that the future-construction approach with Grammatical Evolution yields a substantial and consistent improvement over standard machine learning models for the discrimination of seismic events.

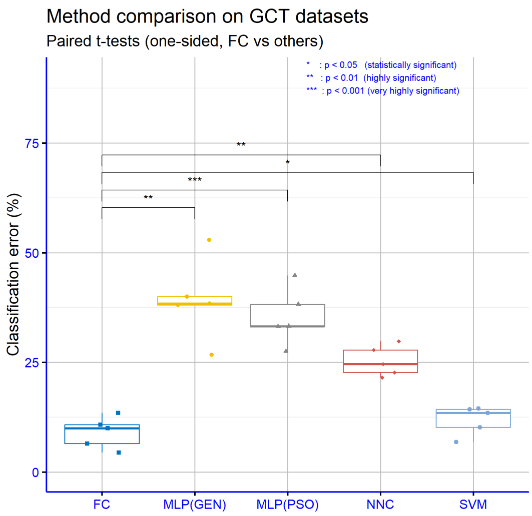


Figure 6. Comparison of FC and baseline classifiers on GCT datasets (paired one-sided t-tests on classification error)

The repeated-measures ANOVA with Method as the within-subject factor and DATASET (year) as the blocking factor confirms that the choice of classifier has a statistically significant effect on the classification error, indicating that the models are not equivalent in terms of predictive performance. Building on this global result, we conducted pairwise paired t-tests between the proposed FC model and each baseline, using one-sided alternatives that explicitly test the hypothesis that FC achieves lower mean error than its competitors. The resulting p-values, visualised as significance stars on the boxplot, show that the superiority of FC is statistically supported across all comparisons (Figure 6).

In particular, the comparisons FC vs MLP(GEN) and FC vs NNC yield ** ($p < 0.01$), indicating that the probability of observing differences of this magnitude in favour of FC purely by chance is below 1%. An even stronger effect emerges for FC vs MLP(PSO), which is marked with *** ($p < 0.001$), reflecting the very large performance gap between FC and the PSO-trained MLP. Finally, the comparison FC vs SVM is annotated with * ($p < 0.05$), showing that, although the numerical difference in error is smaller than in the neural baselines, it remains statistically significant in favour of FC. Overall, the pattern of significance codes (*, **, ***) corroborates the message of Table 6: FC is not only the best-performing model in terms of average classification error, but its advantage over all other methods is statistically significant, with particularly strong evidence against the MLP-based models and clear, though more moderate, evidence against the already very competitive SVM baseline.

3.1. Experiments with the number of features

An additional experiment was conducted, where the number of constructed features was changed from 1 to 4 for the proposed feature construction method. The corresponding experimental results are outlined in Table 5.

Table 5. Experimental results using the proposed method and a series of values for the number of constructed features N_f.

DATASET	$N_f = 1$	$N_f = 2$	$N_f = 3$	$N_f = 4$
GCT1990	11.03%	13.50%	14.62%	15.40%
GCT1995	9.41%	10.82%	9.98%	10.05%
GCT2000	6.51%	6.52%	6.75%	6.54%
GCT2005	4.40%	4.45%	4.42%	4.38%
GCT2010	9.72%	9.97%	9.82%	10.32%
AVERAGE	**8.21%**	**9.05%**	**9.12%**	**9.34%**

Table 5 investigates the behavior of the proposed FC model as the number of constructed features N_f varies from 1 to 4. The results show that performance is generally very stable, with the mean classification error ranging within a narrow band between 8.21% (for N_f = 1) and 9.34% (for N_f = 4). The lowest average error is obtained with a single constructed feature (N_f =1), whereas the configuration N_f =2, which is adopted as the default setting in Table 4, yields a mean error of 9.05%, very close to the optimum and with highly consistent behaviour across all years. In some datasets, such as GCT2005, slightly better values appear for larger N_f (e.g., 4.38% for N_f = 4 versus 4.40% for N_f = 1), but these gains are marginal and do not change the overall picture. Taken together, the results indicate that FC does not require a large number of future constructions to perform well: one or two constructed features are sufficient to achieve very low error rates, while further increasing N_f does not lead to systematic improvements and likely introduces redundant information that does not translate into better generalization. This supports the view that the quality of the features generated by Grammatical Evolution is more important than their quantity, and that the proposed choice N_f = 2 offers a well-balanced compromise between performance and model simplicity.

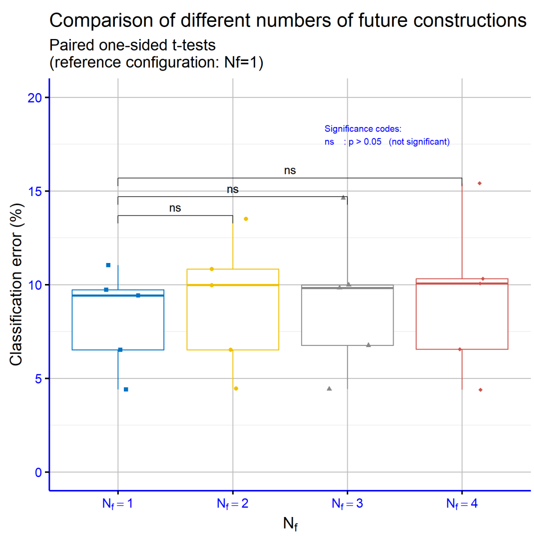


Figure 7. Comparison of FC and baseline classifiers on GCT datasets (paired one-sided t-tests on classification error)

The boxplot for Table 5, together with the paired t-tests, shows that none of the comparisons among the N_f = 1, N_f = 2, N_f = 3 and N_f = 4 configurations reaches statistical significance (all labelled as ns). This indicates that, given the available datasets, the performance of the FC model is essentially insensitive to the number of future constructions, and that using smaller values such as N_f = 1 or N_f = 2 achieves comparable accuracy without any statistically supported gains from increasing N_f (Figure 7).

3.2. Experiments with the number of generations

Moreover, in order to test the efficiency of the proposed method, an additional experiment was executed, where the number of generations was altered from 50 to 400.

Table 6. Experimental results using the proposed method and a series of values for the number generations N_g.

DATASET	$N_g = 50$	$N_g = 100$	$N_g = 200$	$N_g = 400$
GCT1990	13.61%	13.19%	13.50%	12.30%
GCT1995	11.06%	11.70%	10.82%	10.74%
GCT2000	6.53%	6.58%	6.52%	6.52%
GCT2005	4.58%	4.45%	4.45%	4.41%
GCT2010	9.88%	10.00%	9.97%	9.94%
AVERAGE	**9.13%**	**9.18%**	**9.05%**	**8.78%**

Table 6 focuses on the number of generations N_g of the evolutionary algorithm during feature construction and evaluates four values (50, 100, 200, 400). The results demonstrate that FC is remarkably robust with respect to this hyperparameter: the mean classification error remains very close across settings, ranging from 9.18% for N_g = 100 down to 8.78% for N_g = 400. The configuration N_g = 200, which is used as the default in the previous analysis, attains an average error of 9.05%, essentially indistinguishable from the more expensive setting N_g = 400, which improves the mean error only by about a quarter of a percentage point. At the level of individual datasets there are cases where a larger number of generations yields more noticeable gains (for instance, GCT1990 improves to 12.30% at N_g = 400), but the overall pattern is that most of the benefit is already captured within 100-200 generations, with further iterations providing diminishing returns. Thus, Table 6 suggests that the Grammatical Evolution process converges to useful future constructions relatively quickly, and that the choice N_g = 200 offers a very good trade-off between computational cost and predictive performance, avoiding unnecessary increases in training time without delivering substantial accuracy gains.

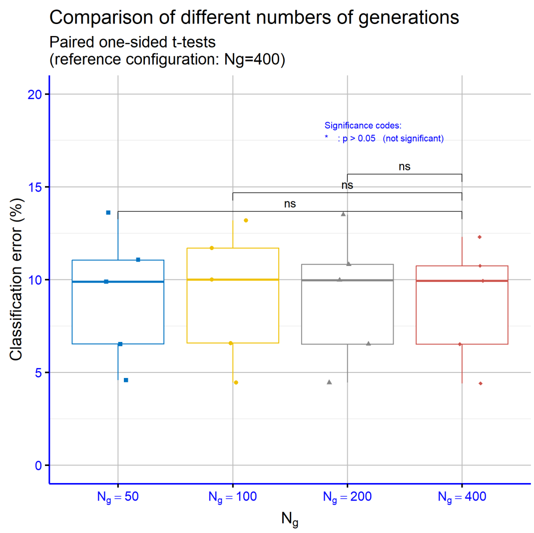


Figure 8. Comparison of FC and baseline classifiers on GCT datasets (paired one-sided t-tests on classification error)

In Figure 8, the paired t-tests for the different numbers of generations show that all pairwise comparisons are non-significant (ns), indicating that variations in N_g do not materially affect the classification error of the FC model.

4. Conclusions

This study investigates the use of Grammatical Evolution for constructing artificial features in earthquake prediction, applying several machine learning approaches, including MLP(GEN), MLP(PSO), SVM, and NNC, alongside Feature Construction (FC). The analysis is based on seismic data recorded between 1990 and 2015 within the geographical area defined by latitudes 33°–44° and longitudes 17°–44°. While all the aforementioned methods belong to the domain of machine learning, FC is distinguished as a feature engineering technique. Specifically, Feature Construction (FC) refers to the process of generating new, informative attributes from the existing dataset, thereby enhancing the representational capacity of the data and improving the performance of machine learning models. Following these steps, our experiments demonstrated that the FC technique yielded the best results, achieving the lowest mean error 9.05% corresponding to an overall accuracy of 91%. The SVM method achieved the second-best performance, with an average error of 11.86%. Consequently, we proceeded with the FC technique, which yielded the best results, and implemented artificially constructed features N_f with ranging from 1 to 4. Furthermore, to evaluate the efficiency of the proposed method, an additional experiment was conducted in which the number of generations N_g was varied from 50 to 400. Consequently, we applied 1, 2, 3, and 4 constructed features within this technique, with the single constructed featureN_f =1 exhibiting superior performance compared to the others producing the minimal average error 8.21%. Our experiment was further extended by incorporating a series of values for the number of generationsN_g (50, 100, 200, and 400), and the results indicated that N_g 400 generations yielded the best performance 8.78%. In contrast, selecting N_g 200 provides also an effective balance between computational cost and predictive performance. This study was conducted as a direct response to the challenge identified in our previous research, thereby extending and refining the scope of the earlier findings.

Author Contributions: Conceptualization, C.K. and I.G.T.; methodology, C.K.; software, I.G.T.; validation, G.K. C.S., V.C.; formal analysis, G.K..; investigation, C.K.; resources, C.S.; data curation, C.K..; writing original draft preparation, C.K.; writing review and editing, I.G.T.; visualization, V.C.; supervision, C.S.; project administration, C.S.; funding acquisition, C.S. 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: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

References

1. United Nations Office for Disaster Risk Reduction (UNDRR). 2017. The Sendai Framework Terminology on Disaster Risk Reduction. "Disaster risk reduction". Available from: https://www.undrr.org/terminology/disaster-risk-reduction. (accessed 13 November 2025).
2. Nakamura, Y. (1984). Development of earthquake early-warning system for the Shinkansen, some recent earthquake engineering research and practice in Japan. Proceeding of The Japanese National Committee of the International Association for Earthquake Engineering, June 1984, 224–238.
3. Nakamura, Y. (1988). On the urgent earthquake detection and alarm system (UrEDAS). In Proc. 9th World Conference on Earthquake Engineering, 1988 (pp. 673-678).

4. Espinosa Aranda, J. M., Jimenez, A., Ibarrola, G., Alcantar, F., Aguilar, A., Inostroza, M., & Maldonado, S. (1995). Mexico City seismic alert system. Seismological Research Letters, 66, 42-53.
5. Kamigaichi, O., Saito, M., Doi, K., Matsumori, T., Tsukada, S. Y., Takeda, K., ... & Watanabe, Y. (2009). Earthquake early warning in Japan: Warning the general public and future prospects. Seismological Research Letters, 80(5), 717-726.
6. Earthquake Early Warning System (Japan). Wikipedia. Available online: https://en.wikipedia.org/wiki/Earthquake_Early_Warning_(Japan) (accessed 14 November 2025).
7. Wenzel, F., Oncescu, M. C., Baur, M., Fiedrich, F., & Ionescu, C. (1999). An early warning system for Bucharest. Seismological Research Letters, 70(2), 161-169.
8. Alcik, H., Ozel, O., Apaydin, N., & Erdik, M. (2009). A study on warning algorithms for Istanbul earthquake early warning system. Geophysical Research Letters, 36(5).
9. Satriano, C., Elia, L., Martino, C., Lancieri, M., Zollo, A., & Iannaccone, G. (2009, December). The Earthquake Early Warning System for Southern Italy: Concepts, Capabilities and Future Perspectives. In AGU Fall Meeting Abstracts (Vol. 2009, pp. S13A-1726).
10. ShakeAlert. US Geological Survey, Earthquake Hazards Program. Available online: https://earthquake.usgs.gov/data/shakealert/ (accessed 15 November 2025).
11. ShakeAlert. Earthquake Early Warning (EEW) System. Available online: https://www.shakealert.org/(accessed 15 November 2025).
12. Kopitsa, C., Tsoulos, I. G., & Charilogis, V. (2025). Predicting the Magnitude of Earthquakes Using Grammatical Evolution. Algorithms, 18(7), 405.
13. Earthquake app. Available from: https://earthquake.app/#banner_1 (accessed 16 November 2025).
14. Android Earthquake Alert System. Available from: https://crisisresponse.google/android-early-earthquake-warnings (accessed 16 November 2025).
15. Greece Earthquakes. Available from: https://play.google.com/store/apps/details?id=com.greek.Earthquake&pli=1 (accessed 16 November 2025).
16. Earthquake Network. Available from:https://sismo.app/(accessed 16 November 2025).
17. Housner, G. W., & Jennings, P. C. (1964). Generation of artificial earthquakes. Journal of the Engineering Mechanics Division, 90(1), 113-150.
18. Adeli, H., & Panakkat, A. (2009). A probabilistic neural network for earthquake magnitude prediction. Neural networks, 22(7), 1018-1024.
19. Zhou, Q., Tong, G., Xie, D., Li, B., & Yuan, X. (2012). A seismic-based feature extraction algorithm for robust ground target classification. IEEE Signal Processing Letters, 19(10), 639-642.
20. Martínez-Álvarez, F., Reyes, J., Morales-Esteban, A., & Rubio-Escudero, C. (2013). Determining the best set of seismicity indicators to predict earthquakes. Two case studies: Chile and the Iberian Peninsula. Knowledge-Based Systems, 50, 198-210.
21. Schmidt, L., Hegde, C., Indyk, P., Lu, L., Chi, X., & Hohl, D. (2015, April). Seismic feature extraction using Steiner tree methods. In 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 1647-1651). IEEE.
22. Narayanakumar, S., & Raja, K. (2016). A BP artificial neural network model for earthquake magnitude prediction in Himalayas, India. Circuits and Systems, 7(11), 3456-3468.
23. Asencio-Cortés, G., Martínez-Álvarez, F., Morales-Esteban, A., & Reyes, J. (2016). A sensitivity study of seismicity indicators in supervised learning to improve earthquake prediction. Knowledge-Based Systems, 101, 15-30.
24. Asim, K. M., Idris, A., Iqbal, T., & Martínez-Álvarez, F. (2018). Earthquake prediction model using support vector regressor and hybrid neural networks. PloS one, 13(7), e0199004.
25. Chamberlain, C. J., & Townend, J. (2018). Detecting real earthquakes using artificial earthquakes: On the use of synthetic waveforms in matched-filter earthquake detection. Geophysical Research Letters, 45(21), 11-641.
26. Okada, A., & Kaneda, Y. (2018, May). Neural network learning: Crustal state estimation method from time-series data. In 2018 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO) (pp. 141-146). IEEE.
27. Lin, J. W., Chao, C. T., & Chiou, J. S. (2018). Determining neuronal number in each hidden layer using earthquake catalogues as training data in training an embedded back propagation neural network for predicting earthquake magnitude. Ieee Access, 6, 52582-52597.
28. Zhang, L., Si, L., Yang, H., Hu, Y., & Qiu, J. (2019). Precursory pattern based feature extraction techniques for earthquake prediction. IEEE Access, 7, 30991-31001.
29. Rojas, O., Otero, B., Alvarado, L., Mus, S., & Tous, R. (2019). Artificial neural networks as emerging tools for earthquake detection. Computación y Sistemas, 23(2), 335-350.
30. Ali, A., Sheng-Chang, C., & Shah, M. (2020). Continuous wavelet transformation of seismic data for feature extraction. SN Applied Sciences, 2(11), 1835.

31. Bamer, F., Thaler, D., Stoffel, M., & Markert, B. (2021). A monte carlo simulation approach in non-linear structural dynamics using convolutional neural networks. Frontiers in Built Environment, 7, 679488.
32. Wang, T., Bian, Y., Zhang, Y., & Hou, X. (2023). Using artificial intelligence methods to classify different seismic events. Seismological Society of America, 94(1), 1-16.
33. Ozkaya, S. G., Baygin, M., Barua, P. D., Tuncer, T., Dogan, S., Chakraborty, S., & Acharya, U. R. (2024). An automated earthquake classification model based on a new butterfly pattern using seismic signals. Expert Systems with Applications, 238, 122079.
34. Sinha, D. K., & Kulkarni, S. (2025, June). Advancing Seismic Prediction through Machine Learning: A Comprehensive Review of the Transformative Impact of Feature Engineering. In 2025 International Conference on Emerging Trends in Industry 4.0 Technologies (ICETI4T) (pp. 1-8). IEEE.
35. Mahmoud, A., Alrusaini, O., Shafie, E., Aboalndr, A., & Elbelkasy, M. S. (2025). Machine Learning-Based Earthquake Prediction: Feature Engineering and Model Performance Using Synthetic Seismic Data. Appl. Math, 19(3), 695-702.
36. O'Neill, M., & Ryan, C. (2002). Grammatical evolution. IEEE Transactions on Evolutionary Computation, 5(4), 349-358.
37. Kramer, O. (2017). Genetic algorithms. In Genetic algorithm essentials (pp. 11-19). Cham: Springer International Publishing.
38. 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.
39. 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.
40. 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.
41. A. Brabazon, M. O'Neill, Credit classification using grammatical evolution, Informatica **30.3**, 2006.
42. S. Şen, J.A. Clark. A grammatical evolution approach to intrusion detection on mobile ad hoc networks, In: Proceedings of the second ACM conference on Wireless network security, 2009.
43. L. Chen, C.H. Tan, S.J. Kao, T.S. Wang, Improvement of remote monitoring on water quality in a subtropical reservoir by incorporating grammatical evolution with parallel genetic algorithms into satellite imagery, Water Research **42**, pp. 296-306, 2008.
44. J. I. Hidalgo, J. M. Colmenar, J.L. Risco-Martin, A. Cuesta-Infante, E. Maqueda, M. Botella,J. A. Rubio, Modeling glycemia in humans by means of Grammatical Evolution, Applied Soft Computing **20**, pp. 40-53, 2014.
45. J. Tavares, F.B. Pereira, Automatic Design of Ant Algorithms with Grammatical Evolution. In: Moraglio, A., Silva, S., Krawiec, K., Machado, P., Cotta, C. (eds) Genetic Programming. EuroGP 2012. Lecture Notes in Computer Science, vol 7244. Springer, Berlin, Heidelberg, 2012.
46. M. Zapater, J.L. Risco-Martín, P. Arroba, J.L. Ayala, J.M. Moya, R. Hermida, Runtime data center temperature prediction using Grammatical Evolution techniques, Applied Soft Computing **49**, pp. 94-107, 2016.
47. C. Ryan, M. O'Neill, J.J. Collins, Grammatical evolution: Solving trigonometric identities, proceedings of Mendel. Vol. 98. 1998.
48. 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.
49. Lídio Mauro Limade Campo, R. Célio Limã Oliveira,Mauro Roisenberg, Optimization of neural networks through grammatical evolution and a genetic algorithm, Expert Systems with Applications **56**, pp. 368-384, 2016.
50. K. Soltanian, A. Ebnenasir, M. Afsharchi, Modular Grammatical Evolution for the Generation of Artificial Neural Networks, Evolutionary Computation **30**, pp 291–327, 2022.
51. I. Dempsey, M.O' Neill, A. Brabazon, Constant creation in grammatical evolution, International Journal of Innovative Computing and Applications **1** , pp 23–38, 2007.
52. 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.
53. 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.
54. 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.
55. N. R. Sabar, M. Ayob, G. Kendall, R. Qu, Grammatical Evolution Hyper-Heuristic for Combinatorial Optimization Problems, IEEE Transactions on Evolutionary Computation **17**, pp. 840-861, 2013.
56. 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.
57. P.J. Pereira, P. Cortez, R. Mendes, Multi-objective Grammatical Evolution of Decision Trees for Mobile Marketing user conversion prediction, Expert Systems with Applications **168**, 114287, 2021.

58. F. Castejón, E.J. Carmona, Automatic design of analog electronic circuits using grammatical evolution, Applied Soft Computing **62**, pp. 1003-1018, 2018.
59. Tsoulos, I. G., Tzallas, A., & Karvounis, E. (2023). Constructing the Bounds for Neural Network Training Using Grammatical Evolution. Computers, 12(11), 226.
60. Tsoulos, I. G., Varvaras, I., & Charilogis, V. (2024). RbfCon: Construct Radial Basis Function Neural Networks with Grammatical Evolution. Software (2674-113X), 3(4).
61. Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11).
62. Suryadevara, S., & Yanamala, A. K. Y. (2021). A Comprehensive Overview of Artificial Neural Networks: Evolution, Architectures, and Applications. Revista de Inteligencia Artificial en Medicina, 12(1), 51-76.
63. Sivanandam, S. N., & Deepa, S. N. (2008). Genetic algorithms. In Introduction to genetic algorithms (pp. 15-37). Berlin, Heidelberg: Springer Berlin Heidelberg.
64. Kalogirou, S. A. (2004). Optimization of solar systems using artificial neural-networks and genetic algorithms. Applied Energy, 77(4), 383-405.
65. Chiroma, H., Noor, A. S. M., Abdulkareem, S., Abubakar, A. I., Hermawan, A., Qin, H., ... & Herawan, T. (2017). Neural networks optimization through genetic algorithm searches: a review. Appl. Math. Inf. Sci, 11(6), 1543-1564.
66. Kramer, O. (2017). Genetic algorithms. In Genetic algorithm essentials (pp. 11-19). Cham: Springer International Publishing.
67. Wang, D., Tan, D., & Liu, L. (2018). Particle swarm optimization algorithm: an overview. Soft computing, 22(2), 387-408.
68. Jain, N. K., Nangia, U., & Jain, J. (2018). A review of particle swarm optimization. Journal of The Institution of Engineers (India): Series B, 99(4), 407-411.
69. Meissner, M., Schmuker, M., & Schneider, G. (2006). Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training. BMC bioinformatics, 7(1), 125.
70. Garro, B. A., & Vázquez, R. A. (2015). Designing artificial neural networks using particle swarm optimization algorithms. Computational intelligence and neuroscience, 2015(1), 369298.
71. Suthaharan, S. (2016). Support vector machine. In Machine learning models and algorithms for big data classification: thinking with examples for effective learning (pp. 207-235). Boston, MA: Springer US.
72. Q. Wang. (2022). Support Vector Machine Algorithm in Machine Learning, IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 2022, pp. 750-756, doi: 10.1109/ICAICA54878.2022.9844516.
73. Astuti, W., Akmeliawati, R., Sediono, W., & Salami, M. J. E. (2014). 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.
74. Asaly, S., Gottlieb, L. A., Inbar, N., & Reuveni, Y. (2022). Using support vector machine (SVM) with GPS ionospheric TEC estimations to potentially predict earthquake events. Remote Sensing, 14(12), 2822.
75. Reddy, R., & Nair, R. R. (2013). The efficacy of support vector machines (SVM) in robust determination of earthquake early warning magnitudes in central Japan. Journal of Earth System Science, 122(5), 1423-1434.
76. I.G. Tsoulos, D. Gavrilis, E. Glavas, Neural network construction and training using grammatical evolution, Neurocomputing **72**, pp. 269-277, 2008.
77. 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.
78. I.G. Tsoulos, D. Gavrilis, E. Glavas, Solving differential equations with constructed neural networks, Neurocomputing **72**, pp. 2385-2391, 2009.
79. I.G. Tsoulos, G. Mitsi, A. Stavrakoudis, S. Papapetropoulos, Application of Machine Learning in a Parkinson's Disease Digital Biomarker Dataset Using Neural Network Construction (NNC) Methodology Discriminates Patient Motor Status, Frontiers in ICT 6, 10, 2019.
80. 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.
81. E.I. Toki, J. Pange, G. Tatsis, K. Plachouras, I.G. Tsoulos, Utilizing Constructed Neural Networks for Autism Screening, Applied Sciences **14**, 3053, 2024.
82. Dimitris Gavrilis, Ioannis G. Tsoulos, Evangelos Dermatas, Selecting and constructing features using grammatical evolution, Pattern Recognition Letters **29**,pp. 1358-1365, 2008.
83. George Georgoulas, Dimitris Gavrilis, Ioannis G. Tsoulos, Chrysostomos Stylios, João Bernardes, Peter P. Groumpos, Novel approach for fetal heart rate classification introducing grammatical evolution, Biomedical Signal Processing and Control **2**,pp. 69-79, 2007

84. Otis Smart, Ioannis G. Tsoulos, Dimitris Gavrilis, George Georgoulas, Grammatical evolution for features of epileptic oscillations in clinical intracranial electroencephalograms, Expert Systems with Applications **38**, pp. 9991-9999, 2011
85. A. T. Tzallas, I. Tsoulos, M. G. Tsipouras, N. Giannakeas, I. Androulidakis and E. Zaitseva, Classification of EEG signals using feature creation produced by grammatical evolution, In: 24th Telecommunications Forum (TELFOR), pp. 1-4, 2016.
86. J. Park and I. W. Sandberg, Universal Approximation Using Radial-Basis-Function Networks, Neural Computation 3, pp. 246-257, 1991.
87. 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.
88. Tsoulos, I.G.; Charilogis, V.; Kyrou, G.; Stavrou, V.N.; Tzallas, A. OPTIMUS: A Multidimensional Global Optimization Package. J. Open Source Softw. 2025, 10, 7584.
89. 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.