FastMapSVM: Classifying Complex Objects Using the FastMap Algorithm and Support-Vector Machines

Malcolm C. A. White,[1*] Kushal Sharma,[2] Ang Li,[2] T. K. Satish Kumar,[2] Nori Nakata[1,3]

[1]Massachusetts Institute of Technology,
Cambridge, MA 02139, USA
[2]University of Southern California
Los Angeles, CA 90007, USA
[3]Lawrence Berkeley National Laboratory
Berkeley, CA 94720, USA

*To whom correspondence should be addressed; E-mail: malcolmw@mit.edu.

Neural Networks and related Deep Learning methods are currently at the leading edge of technologies used for classifying objects. However, they generally demand large amounts of time and data for model training; and their learned models can sometimes be difficult to interpret. In this paper, we advance FastMapSVM—an interpretable Machine Learning framework for classifying complex objects—as an advantageous alternative to Neural Networks for general classification tasks. FastMapSVM combines the complementary strengths of FastMap and Support-Vector Machines. FastMap is an efficient linear-time algorithm that maps complex objects to points in a Euclidean space, while preserving pairwise non-Euclidean distances between them. We demonstrate the efficiency and effectiveness of FastMapSVM in the context of clas-

sifying seismograms. We show that its performance, in terms of precision, recall, and accuracy, is comparable to that of other state-of-the-art methods. However, compared to other methods, FastMapSVM uses significantly smaller amounts of time and data for model training. It also provides a perspicuous visualization of the objects and the classification boundaries between them. We expect FastMapSVM to be viable for classification tasks in many other real-world domains.

Introduction

Various Machine Learning (ML) and Deep Learning (DL) methods, such as Neural Networks (NNs), are popularly used for classifying objects. For example, a Convolutional NN (CNN) is used for classifying Sunyaev-Zel'dovich galaxy clusters [1], a densely connected CNN is used for classifying images [2], and a deep NN is used for differentiating the chest X-rays of Covid-19 patients from other cases [3]. However, they generally demand large amounts of time and data for model training; and their learned models can sometimes be difficult to interpret.

In this paper, we advance FastMapSVM [4]—an interpretable ML framework for classifying complex objects—as an advantageous alternative to NNs for general classification tasks. Whereas most ML algorithms learn diagnostic features of *individual objects* in a class, FastMapSVM leverages a domain-specific distance function on *pairs of objects*. It does this by combining the strengths of FastMap and Support-Vector Machines (SVMs). In its first stage, FastMapSVM invokes FastMap, an efficient linear-time algorithm that maps complex objects to points in a Euclidean space, while preserving pairwise distances between them. In its second stage, it invokes SVMs and kernel methods for learning to classify the points in this Euclidean space. The FastMapSVM framework that we implement in this paper is virtually identical in concept to the SupFM-SVM method of Ban *et al.* [4]; however, our development is novel in that it manifests several of the advantages that FastMapSVM offers over other methods that Ban *et al.* [4] only alluded to or altogether overlooked.

First, there are many real-world domains in which feature selection for *individual objects* is challenging, but a distance function on *pairs of objects* is well defined and easy to compute. In such domains, FastMapSVM is more easily applicable than other ML algorithms that focus on the features of individual objects. Examples of such real-world objects include audio signals, seismograms, DNA sequences, electrocardiograms, and magnetic-resonance images. While these objects are complex and may have many subtle features that are hard to recognize, there exists a well-defined distance function on pairs of objects that is easy to compute. For instance, individual DNA sequences have many complex and subtle features, but the *edit distance*[1] between two DNA sequences is well defined and easy to compute.

Second, because FastMapSVM generates a Euclidean embedding, it provides a perspicuous visualization of the objects and the classification boundaries between them. This aids human interpretation of the data and results. It also enables a human-in-the-loop framework for refining the processes of learning and decision making. Moreover, FastMapSVM is able to produce the visualization very efficiently because it invests only linear time in generating the Euclidean embedding.

Third, FastMapSVM uses significantly smaller amounts of time and data for model training compared to other ML algorithms. This is because, given N objects and their classification labels (training instances), FastMapSVM leverages $O(N^2)$ pieces of information via a distance function that is defined on every pair of objects. In contrast, ML algorithms that focus on individual objects leverage only $O(N)$ pieces of information. Despite considering $O(N^2)$ pieces of information to generate a Euclidean embedding, FastMapSVM invests only $O(N)$ time to do so.

[1]The edit distance between two strings is the minimum number of insertions, deletions, or substitutions that are needed to transform one to the other.

Fourth, FastMapSVM extends the applicability of SVMs and kernel methods to domains with complex objects. SVMs and associated kernel methods [5] constitute a very powerful ML framework for classification tasks. However, they are generally applicable only when the objects can be represented in a geometric space. As mentioned before, in many real-world domains, it is unwieldy to represent all the features of a complex object in a geometric space. In such domains, FastMapSVM satisfies this requirement by generating an alternative low-dimensional Euclidean embedding via a distance function.

In this paper, we demonstrate the efficiency and effectiveness of FastMapSVM in the context of classifying seismograms. In fact, this is a particularly illustrative domain because seismograms are complex objects with subtle features indicating diverse energy sources such as earthquakes, ocean-Earth interactions, atmospheric phenomena, and human-related activity. We address two fundamental, perennial questions in seismology: (a) Does a given seismogram record an earthquake? and (b) Which type of wave motion (e.g., compressional versus shear strain) is predominant in an earthquake seismogram? In Earthquake Science, answering these questions is referred to as *detecting earthquakes* and *identifying phases*, respectively. The development of efficient, reliable, and automated solution procedures that can be easily adapted to new environments is critical to modern research and engineering applications in this field, such as in developing Earthquake Early Warning Systems. Towards this end, we show that FastMapSVM is a viable ML framework. Through experiments, we show that FastMapSVM's various performance measures, such as precision, recall, and accuracy, are comparable to that of other state-of-the-art methods. However, we also show that, compared to those methods, FastMapSVM uses significantly smaller amounts of time and data for model training. Moreover, FastMapSVM provides a perspicuous visualization of the seismograms, their spread, and the classification boundaries between them.

The key novel contributions of this paper are as follows:

1. We advance FastMapSVM as an advantageous alternative to other ML algorithms for general classification tasks, such as NNs, by elucidating its algorithmic attributes.

2. We demonstrate that FastMapSVM performs comparably to state-of-the-art NNs for classifying seismograms using two orders of magnitude less time and data for model training.

3. We illustrate how domain knowledge can be explicitly incorporated into the classification task via the user-specified distance function.

4. We show how FastMapSVM extends the applicability of SVMs and kernel methods to domains with complex objects.

5. We provide an efficient implementation of FastMapSVM.

Results

Data

We assess the performance and robustness of FastMapSVM using two data sets. All waveforms used in this paper are bandpass filtered between 1 Hz and 20 Hz before analysis using a zero-phase Butterworth filter with four poles; we refer to this frequency band as our passband.

Stanford Earthquake Data Set (STEAD). The first data set is the Stanford Earthquake Data Set (STEAD) [6], a benchmark data set for training and testing algorithms in Earthquake Science, with over 1.2 million carefully curated, three-component (3C) seismograms. Data in STEAD contain signals from approximately 450 000 different earthquakes—each recorded by a seismometer located within 350 km of the epicenter—and represent seismic activity on every continent except Antarctica. About 100 000 signals in STEAD comprise only noise (i.e., do not contain earthquake-related signals).

We use the entire STEAD data set to assess model performance for detecting earthquakes and subsets of various sizes (appropriately indicated below) to assess model sensitivity to training data size and hyperparameters. To assess model performance for identifying phases, we use a subset of 538 three-second, 3C seismograms from STEAD, all of which were recorded by station TA.109C; 269 start 1 s before a compressional (P-wave) phase arrival, and 269 start 1 s before a shear (S-wave) phase arrival.

Ridgecrest Data Set. The second data set, which we simply refer to as the *Ridgecrest* data set, comprises data recorded by station CI.CLC of the Southern California Seismic Network (SCSN) [7] on 5 July 2019, the first day of the aftershock sequence following the 2019 Ridgecrest, CA, earthquake pair, and on 5 December 2019, five months after the mainshocks. We use the earthquake catalog published by the Southern California Earthquake Data Center (SCEDC) [8] to extract 512 eight-second, 3C seismograms, 256 of which record both P- and S-wave phase arrivals from a nearby aftershock, and the remaining 256 of which record only noise. All 512 of these signals were recorded on 5 July 2019.

We use the Ridgecrest data set to first demonstrate the robustness of FastMapSVM against noisy perturbations. We then use it to demonstrate FastMapSVM's ability to detect new microseisms by automatically scanning a 600 s, continuous, 3C seismogram recorded between 01:00:00 and 01:10:00 (UTC) on 5 December 2019. Whereas the analysis on the STEAD data set demonstrates FastMapSVM's performance on a benchmark, the analysis on the Ridgecrest data set provides an example of a more realistic use case of FastMapSVM: After handpicking only a small number of earthquake and noise signals—a task that even a novice analyst can perform in a few hours—continually arriving seismic data can be automatically scanned for additional earthquake signals. This capability manifests the primary conclusion of the preceding robustness test: Even when earthquake signals are difficult to discern by the human eye, FastMapSVM can often reliably detect them.

STEAD Analysis

Detecting Earthquakes in STEAD. The *EQTransformer* DL model [9] for simultaneously detecting earthquakes and identifying phase arrivals is arguably the most accurate, publicly available model for this pair of tasks. The authors of EQTransformer report perfect precision and recall scores for detecting earthquakes in 10 % of the STEAD waveforms after training its more than 300 000 model parameters with 85 % (i.e., $\sim$1.08$\times$10^6) of the STEAD waveforms; 5 % of the STEAD waveforms were reserved for model validation.

Using only $\sim$1 % (i.e., 16 384) of the STEAD waveforms, we train FastMapSVM and classify the remaining 99 % of the data ($\sim$1.477$\times$10^6 waveforms) with precision, recall, and accuracy scores of 0.995, 0.973, and 0.975, respectively. Fig. 1 and Table 1 summarize these performance results. Equal numbers of randomly selected noise and earthquake waveforms make up the training data set, whereas $\sim$2.272$\times$10^5 noise and $\sim$1.249$\times$10^6 earthquake waveforms, respectively, make up the test data set. FastMapSVM incorrectly labels only 2.8 % of noise waveforms as earthquakes and 2.7 % of earthquake waveforms as noise.

Table 1. Model performance comparison. Shows a comparison between the detection performances of FastMapSVM and other NN models trained on the STEAD data set. Performance data for EQTransformer and CRED are taken from Table 1 of [9].

Model	Precision	Recall	F1	Training Size	Reference
EQTransformer	1.0	1.0	1.0	1.2$\times$10^6	[9]
CRED	1.0	0.96	0.98	1.2$\times$10^6	[10]
FastMapSVM	1.0	0.97	0.98	1.6$\times$10^4	This article

The model, which comprises of a 32-dimensional Euclidean embedding of seismograms, took 26 minutes to train on a 64-core workstation. This is significantly smaller in comparison to the training requirements of EQTransformer, which took roughly 89 hours on 4 parallel Tesla-

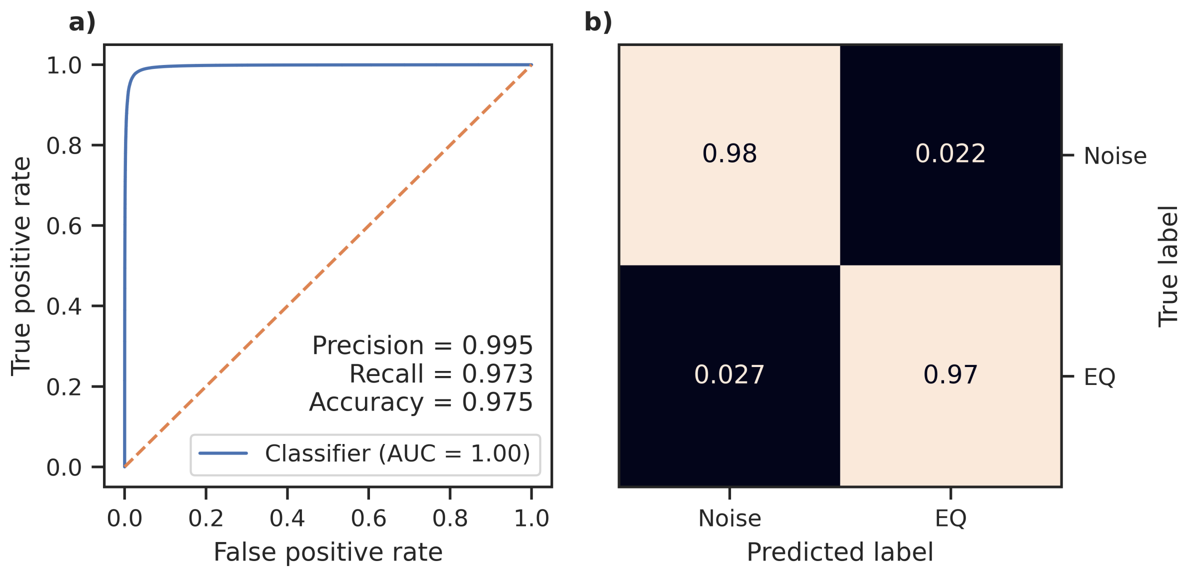

Fig. 1. FastMapSVM's performance detecting earthquakes in STEAD. Shows the performance of FastMapSVM on the STEAD data set for classifying Earthquake and Noise signals. (a) shows the Receiver Operating Characteristic (ROC) curve and the corresponding Area Under the Curve (AUC). In its inlay, it also shows the precision, recall, and accuracy achieved with the best model parameters. (b) shows the confusion matrix for the learned model with respect to classifying the Earthquake (EQ) and Noise signals.

V100 GPUs [9]. Classifying the test data took FastMapSVM roughly 5 hours. Using two orders of magnitude less training data and time, FastMapSVM competes with leading NN models trained to detect earthquakes using the STEAD data set. The complexity of the EQTransformer model (and the resultant demands placed on training data and time) are partly due to the fact that it detects earthquakes and identifies phases simultaneously. Although FastMapSVM can be trained for both of these tasks, a separate model must be trained for each. FastMapSVM convincingly outperforms the CRED model [10], which only detects earthquakes and was trained using the same data set as EQTransformer.

Sensitivity to Training Data Size and Hyperparameters. Two important questions concerning FastMapSVM are: (a) How much training data is needed to train the model? and (b) How many Euclidean dimensions are needed to represent the objects being classified? We address both these questions below.

To assess FastMapSVM's sensitivity to the amount of training data used, we obtain a suite of FastMapSVM models trained with various amounts of data. We score their performances on a subset of 16 384 test waveforms randomly selected from STEAD. We ensure that the test data is balanced with equal numbers of earthquake and noise seismograms (Fig. 2a). The precision appears relatively insensitive to the amount of training data; however, the accuracy and recall increase significantly with the amount of training data. This implies that the FastMapSVM models seldom classify noise as an earthquake, irrespective of the amount of training data. On the other hand, the frequency with which they classify earthquakes as noise decreases as the amount of training data increases. Such behaviour is unsurprising because it is highly unlikely for a noise signal to be more similar to a reference earthquake signal than to a reference noise signal, regardless of how many earthquake signals it is compared to. In contrast, it is relatively more likely for an earthquake signal to be sufficiently dissimilar from all reference earthquake signals and consequently get classified as noise when the number of reference earthquake signals is small. Therefore, generally speaking, correctly identifying noise is an easier task than correctly identifying an earthquake.

To assess FastMapSVM's sensitivity to the dimensionality of the Euclidean embedding, we obtain a suite of FastMapSVM models with a varying number of dimensions. We score their performances on the same balanced subset of test waveforms used to assess the model sensitivity to training data size above (Fig. 2b). All performance metrics, particularly, the recall, improve with an increasing number of dimensions. Moreover, the performance results are indicative of the "diminishing returns" property: Strong performance can be achieved with low-dimensional Euclidean embeddings, although small improvements are possible with high-dimensional Euclidean embeddings. The diminishing returns property is an attractive property from the perspective of visualization in low-dimensional Euclidean spaces and from the perspective of trading off performance against memory.

Identifying Phase Arrivals. As another illustration designed to demonstrate the effectiveness of the FastMapSVM framework, we use STEAD to train a model with a 32-dimensional Euclidean embedding. This model is trained to discriminate P- and S-wave phases using 268 seismograms from STEAD extracted for station TA.109C. We then test the model on 270 seismograms with classification accuracy, precision, and recall scores of 0.970, 0.891 , and 0.970, respectively (Fig. 3). The training and test data used in this analysis are both balanced across the P- and S-wave classes. Although these scores are relatively modest in comparison to those of state-of-the-art NNs designed for similar tasks, they demonstrate that FastMapSVM can be easily trained for strong performance using only small amounts of time and data.

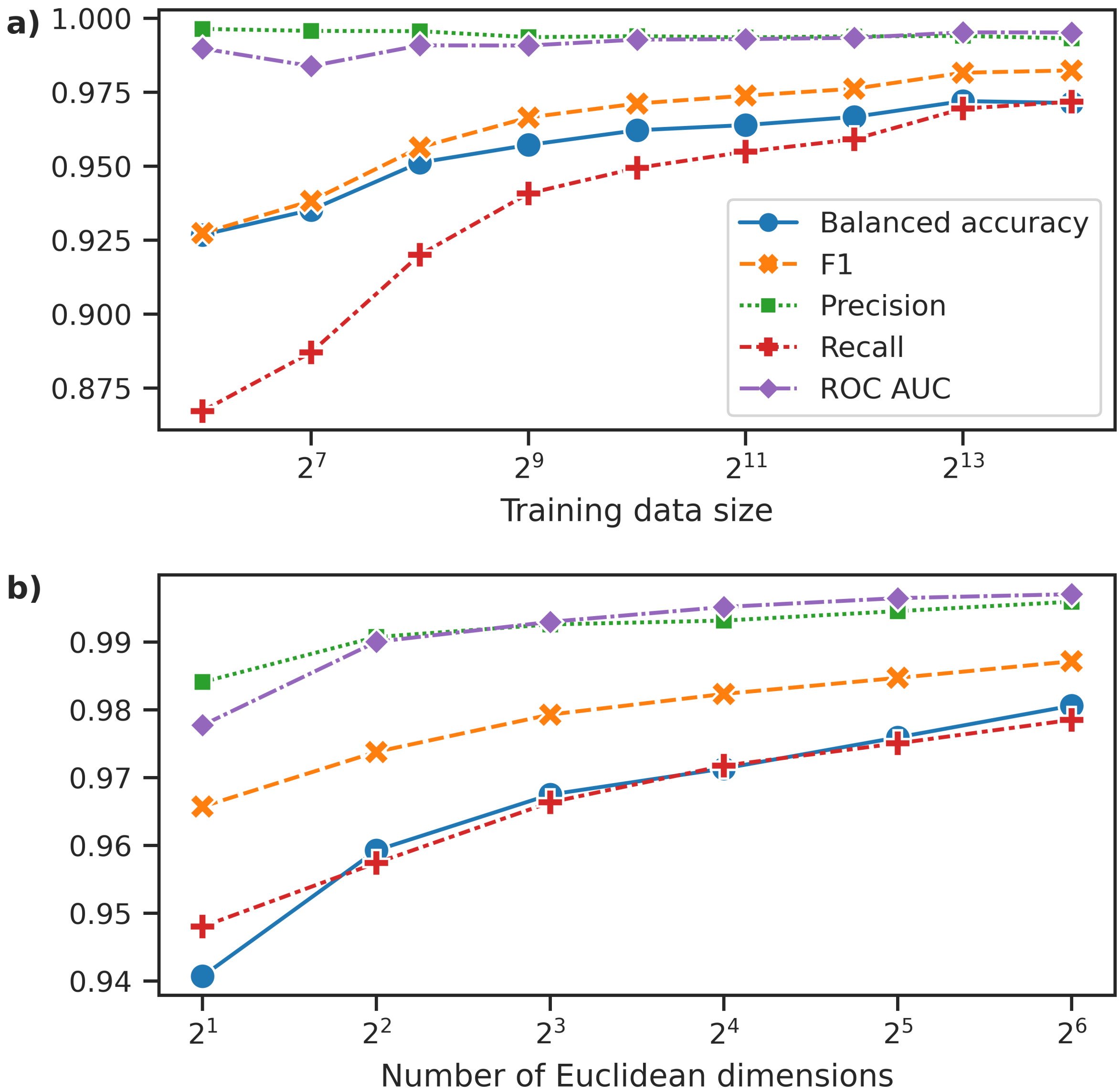

Fig. 2. FastMapSVM's sensitivity to size of training data and Euclidean embedding. Shows the performance of FastMapSVM on the STEAD data set for varying training data size and number of dimensions used for the Euclidean embedding. (a) shows the influence of the training data size, measured using the metrics of balanced accuracy, F1 score, precision, recall, and ROC AUC. (b) shows the influence of the number of dimensions used for the Euclidean embedding, measured using the same metrics.

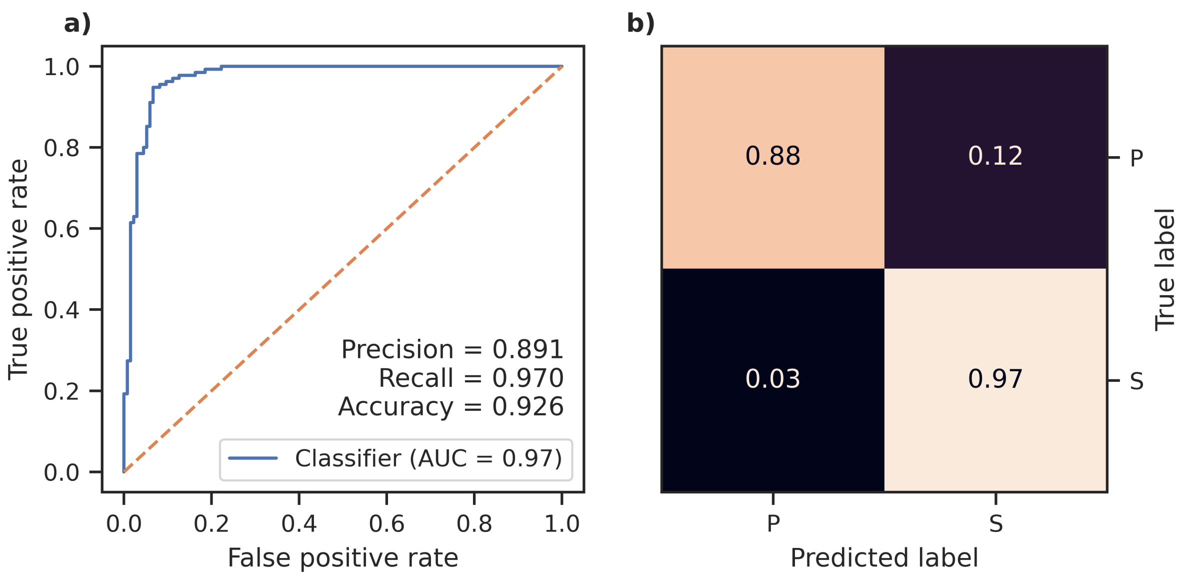

Fig. 3. FastMapSVM's performance identifying phases for station TA.109C in STEAD. Shows the performance of FastMapSVM on the STEAD data set for classifying P- and S-waves. (a) shows the ROC curve and the corresponding AUC. (b) shows the confusion matrix for the learned model with respect to classifying the P- and S-waves recorded by station TA.109C.

Ridgecrest Analysis

Robustness against Noisy Perturbations. It is critical that a classification framework is robust against noisy perturbations of inputs. In general, the robustness of FastMapSVM against noisy perturbations may depend on the characteristics of the data and the chosen distance function. For classifying seismograms, we demonstrate FastMapSVM's robustness against noisy perturbations made to the Ridgecrest data set using the correlation distance described in the Materials and Method section. We randomly select 8 earthquake signals and 8 noise signals to train a FastMapSVM model with a 4-dimensional Euclidean embedding. Each of the 496 remaining seismograms is circularly shifted by an offset (in seconds) chosen uniformly at random from the interval $[-2, 2]$. FastMapSVM has a nearly perfect classification accuracy; 2 noise signals are incorrectly labeled as earthquakes. We conduct a subsequent set of experiments in which this model's performance is scored after perturbing signals in the test data set with increasing amounts of Gaussian noise. For each trial, we perturb each signal in the test data set by adding Gaussian noise with mean 0 and standard deviation σ; σ increases by 0.5 after each trial. Fig. 4a shows how a waveform changes with increasing σ. Fig. 4b shows the performance of FastMapSVM with increasing σ. We observe that FastMapSVM continues to classify seismograms with high fidelity, even as earthquake signals become indiscernible to the human eye; e.g., the FastMapSVM model achieves $>$90 % accuracy and precision for $\sigma = 3$.

At first glance, some of the results of the foregoing experiments are counterintuitive. The recall remains at or close to 1 irrespective of the amplitude of the noisy perturbations. The model also accurately identifies earthquakes regardless of the magnitude of the noisy perturbations. In fact, the model misclassifies noise signals as earthquake signals more frequently when the magnitude of the noisy perturbations is increased. With enough added noise, the model classifies all signals as earthquake signals. This is because of the unique frequency content of the noisy perturbations (Supplementary Fig. S1). In our passband, the average frequency spectrum

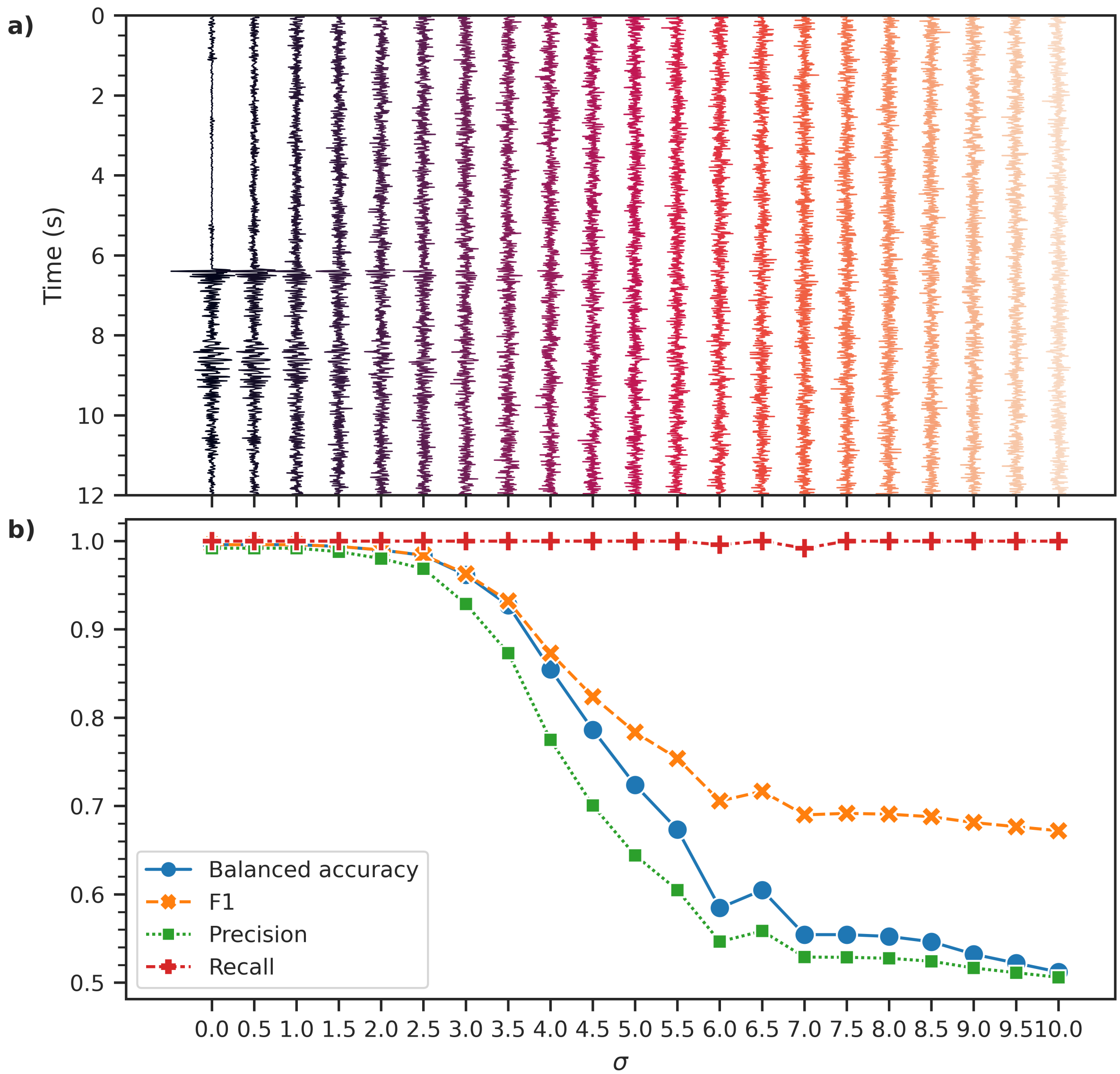

Fig. 4. FastMapSVM's robustness against noisy perturbations. Shows the performance of FastMapSVM on the Ridgecrest data set. (a) shows how a sample test waveform changes with the addition of increasing levels of Gaussian random noise with mean 0 and standard deviation σ. It uses a vertical time-axis and an increasing σ on the horizontal axis. (b) shows how the metrics of balanced accuracy, F1 score, precision, and recall change with increasing σ.

of earthquake signals is nearly flat; whereas the average frequency spectrum of noise signals has prominent peaks near the low- and high-frequency endpoints. Because the noisy perturbations are Gaussian, their frequency spectrum is flat. This makes the frequency spectra of noisy perturbations more similar to those of earthquake signals than those of real noise signals. Thus, the recall and F1 scores get inflated when the amount of added noise increases. However, the accuracy and precision remain unbiased because accuracy is insensitive to false positives and precision penalizes false positives in equal proportion to rewarding true positives.

Automatic Scanning. We further demonstrate a use case-inspired application of FastMapSVM. We first train a model with 128 earthquake signals and 128 noise signals selected randomly from the Ridgecrest data set. We then use the trained model to automatically scan and detect earthquakes in a 600 s, continuous seismogram recorded by station CI.CLC between 01:00:00 and 01:10:00 (UTC) on 5 December 2019. We validate the results after automatically scanning the data. During this time period, the SCEDC earthquake catalog reports no earthquakes within 100 km of CI.CLC; however, FastMapSVM identifies 19 windows with earthquakes. Of these, 9 contain clear earthquake signals with easily discernible P- and S-wave arrivals (Fig. 5a). Another 7 of them contain signals that we believe are from earthquake sources but are difficult to discern, either because they have low signal-to-noise ratios, secondary phase arrivals, or both (Fig. 5b). The remaining 3 of them have ambiguous signals that may or may not be from genuine earthquake sources (Fig. 5c). The complete set of waveforms identified as containing earthquakes in this test, along with our manual categorizations of them, are available in the Supplementary Material (Figs. S2, S3, and S4).

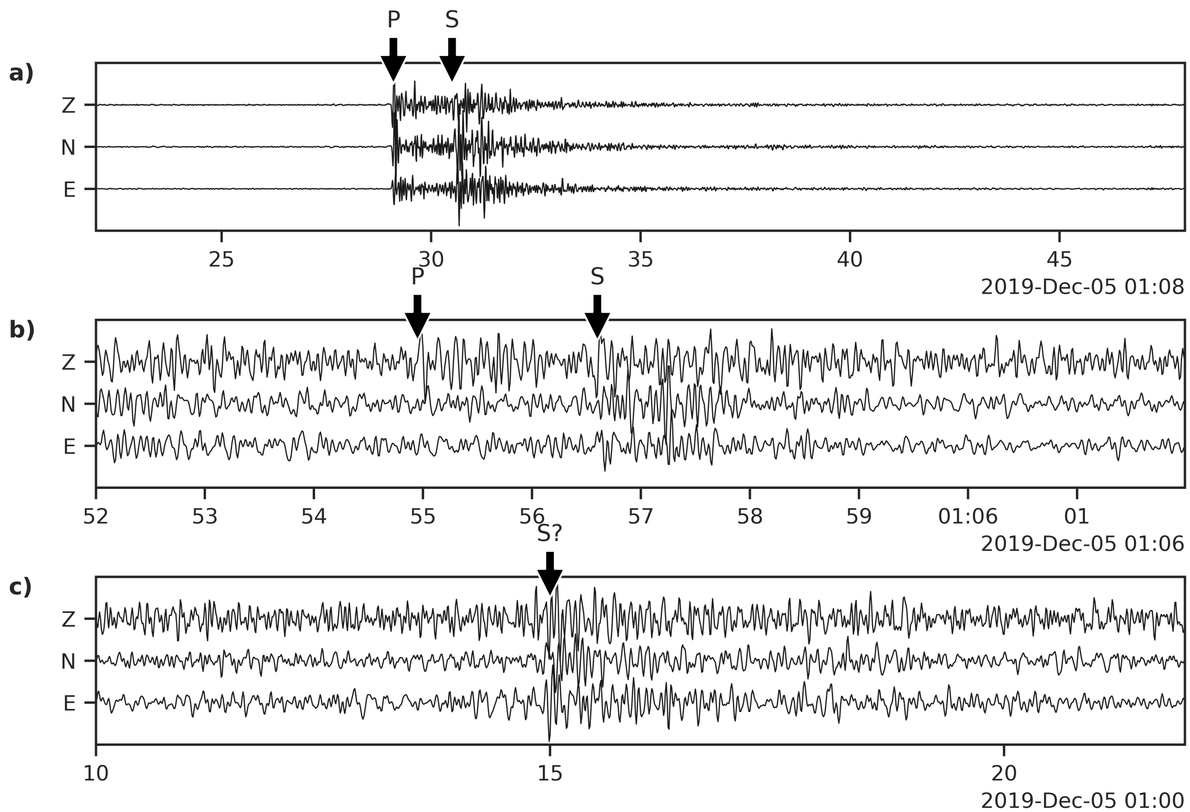

Fig. 5. Example results from an automatic scan for earthquakes using FastMapSVM. Shows example results of automatically scanning 600 s of data recorded by station CI.CLC. (a) shows a clear earthquake signal with easily discernible P- and S-wave arrivals. (b) shows an earthquake signal with low signal-to-noise ratio. The P- and S-wave arrivals are close to the noise level. (c) shows an ambiguous signal that may or may not be from an earthquake source.

Discussion

Although our conception and development of FastMapSVM is our own original and independent work, it is not entirely novel. In fact, Ban *et al.* [4] presented a virtually identical concept. Their key contribution was to combine the power of kernel methods, which typically require formulating an optimization problem in *dual form*, with learning algorithms formulated in the more efficient *primal form* (e.g., linear SVMs). To achieve this, Ban *et al.* [4] first map input data features to an *empirical feature space* using sparse representations of Radial Basis Function kernels, after which they employ a linear SVM to classify instances in this empirical feature space. They assess the performance of kernel Principal Component Analysis and three variants of FastMap for sparsely representing the non-linear kernel within this framework; however, they omit comparisons against any alternative methods, such as NNs. FastMapSVM has many advantages over existing ML methods for classifying complex objects like seismograms, which were largely overlooked by Ban *et al.* [4]. The potential of FastMapSVM was unrealized because these advantages were not made evident. In this section, we discuss some of these advantages, both in the specific context of classifying seismograms and in the general context of ML and data visualization.

Many existing ML algorithms for classification do not leverage domain knowledge when used off the shelf. Although a domain expert can occasionally incorporate domain-specific features of the objects being classified into the classification task, doing so becomes increasingly difficult as the complexity of the objects increases. FastMapSVM enables domain experts to incorporate their domain knowledge via a distance function instead of relying on complex ML models to infer the underlying structure in the data entirely. In fact, in many real-world domains, it is easier to construct a distance function on pairs of objects than it is to extract features of individual objects. Examples include DNA strings, for which the edit distance is well defined, images, for which the Minkowski distance [11] is well defined, and text documents, for which the cosine similarity [12] is well defined. In all these domains, extracting features of individual objects is challenging. In the seismogram domain, our *a priori* knowledge that earthquake seismograms typically bear similarities to one another is encapsulated in a distance function that quantifies the normalized cross-correlation of the waveforms. This distance metric closely resembles other similarity metrics that have been extensively used in previous works in the Earthquake Science community [13–15].

In addition, many existing ML algorithms produce results that are hard to interpret or explain. For example, in NNs, a large number of interactions between neurons with nonlinear activation functions makes a meaningful interpretation or explanation of the results challenging. In fact, the very complexity of the objects in the domain can hinder interpretability and explainability. FastMapSVM mitigates these challenges and thereby supports interpretability and explainability. Although the objects themselves may be complex, FastMapSVM embeds them in a Euclidean space by considering only the distance function defined on pairs of objects. In effect, it simplifies the description of the objects by assigning Euclidean coordinates to them. Moreover, because the distance function is itself user-supplied and encapsulates domain knowledge, FastMapSVM naturally facilitates interpretability and explainability. It even provides a perspicuous visualization of the objects and the classification boundaries between them (Fig. 6). FastMapSVM produces such visualizations very efficiently because it invests only linear time in generating the Euclidean embedding.

FastMapSVM also uses significantly smaller amounts of time and data for model training compared to other ML algorithms. While NNs and other ML algorithms store abstract representations of the training data in their model parameters, FastMapSVM stores explicit references to some of the original objects, referred to as pivots. While making predictions, objects in the test instances are compared directly to the pivots using the user-supplied distance function.

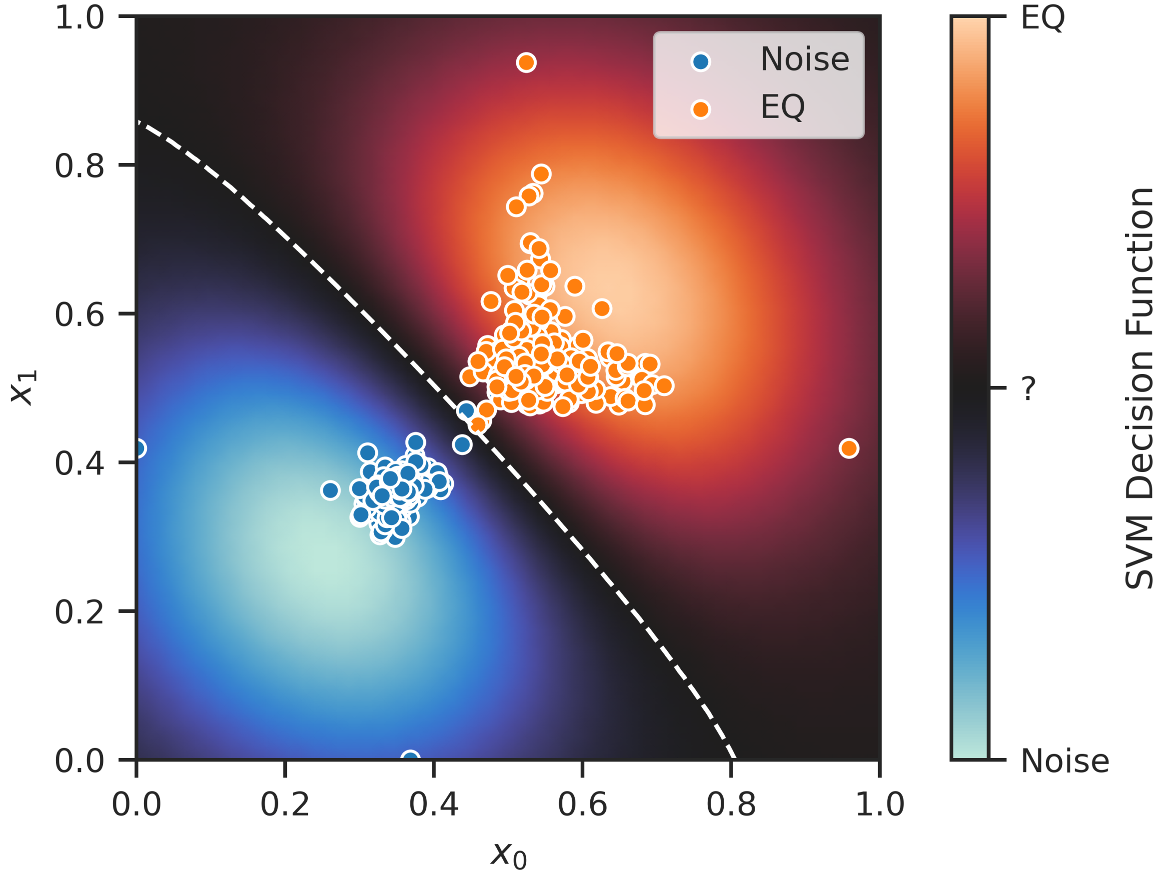

Fig. 6. Perspicuous visualization of seismograms and decision boundaries produced by FastMapSVM. Shows a visualization of FastMapSVM's classification boundary (dashed, white curve) and decision function (background) in a 2-dimensional Euclidean embedding of the training data from the Ridgecrest data set. EQ refers to earthquakes.

FastMapSVM thereby obviates the need to learn a complex transformation of the input data and thus significantly reduces the amount of time and data required for model training. Moreover, given N training instances, FastMapSVM leverages $O(N^2)$ pieces of information via the distance function, which is defined on every pair of objects. In contrast, ML algorithms that focus on individual objects leverage only $O(N)$ pieces of information.

In general, FastMapSVM extends the applicability of SVMs and kernel methods to domains with complex objects. With increasing complexity of the objects, deep NNs have gained more popularity compared to SVMs because it is unwieldy for SVMs to represent all the features of complex objects in Euclidean space. FastMapSVM, however, revitalizes the SVM approach by leveraging a distance function and creating a low-dimensional Euclidean embedding of the objects.

Overall, any application domain hindered by a paucity of training data but possessing a well-defined distance function on pairs of its objects can benefit from the advantages of FastMapSVM. Examples of such applications in Earthquake Science include analyzing and learning from data obtained by distributed acoustic sensing technology or during temporary deployments of "large-N" nodal arrays. Furthermore, the efficiency of FastMapSVM makes it suitable for real-time deployment, which is critical for engineering Earthquake Early Warning Systems.

Materials and Method

Our FastMapSVM method comprises two main components: (a) The FastMap algorithm [16] for embedding complex objects in a Euclidean space using a distance function, and (b) SVMs for classifying objects in the resulting Euclidean space. We explain the key algorithmic concepts behind each of these components below.

Review of the FastMap Algorithm. FastMap [16] is a Data Mining algorithm that embeds complex objects—like audio signals, seismograms, DNA sequences, electrocardiograms, or magnetic-resonance images—into a K-dimensional Euclidean space, for a user-specified value of K and a user-supplied function $\mathcal{D}$ that quantifies the distance, or dissimilarity, between pairs of objects. The Euclidean distance between any two objects in the embedding produced by FastMap approximates the domain-specific distance between them. Therefore, similar objects, as quantified by $\mathcal{D}$, map to nearby points in Euclidean space whereas dissimilar objects map to distant points. Although FastMap preserves $O(N^2)$ pairwise distances between N objects, it generates the embedding in only $O(KN)$ time. Because of its efficiency, FastMap has already found numerous real-world applications, including in Data Mining [16], shortest-path computations [17], and solving combinatorial optimization problems on graphs [18].

Below, we review the FastMap algorithm [16] and describe our minor modifications to it. These modifications suit the purposes of the downstream classification task. Our review of FastMap also serves completeness and the readers' convenience.

FastMap embeds a collection of complex objects in an artificially created Euclidean space that enables geometric interpretations, algebraic manipulations, and downstream application of ML algorithms. It gets as input a collection of complex objects $\mathcal{O}$ and a distance function $\mathcal{D}(\cdot, \cdot)$, where $\mathcal{D}(O_i, O_j)$ represents the domain-specific distance between objects $O_i, O_j \in \mathcal{O}$. It generates a Euclidean embedding that assigns a K-dimensional point $\mathbf{p}_i = (p_{i,1}, p_{i,2}, \ldots, p_{i,K}) \in \mathbb{R}^K$ to each object O_i. A good Euclidean embedding is one in which the Euclidean distance $\|\mathbf{p}_i - \mathbf{p}_j\|_2 = \sqrt{\sum_{n=1}^{K}(p_{i,n} - p_{j,n})^2}$ between any two points $\mathbf{p}_i$ and $\mathbf{p}_j$ closely approximates $\mathcal{D}(O_i, O_j)$.

FastMap creates a K-dimensional Euclidean embedding of the complex objects in $\mathcal{O}$, for a user-specified value of K. In the first iteration, FastMap heuristically identifies the farthest pair of objects O_a and O_b in linear time. Once O_a and O_b are determined, every other object O_i

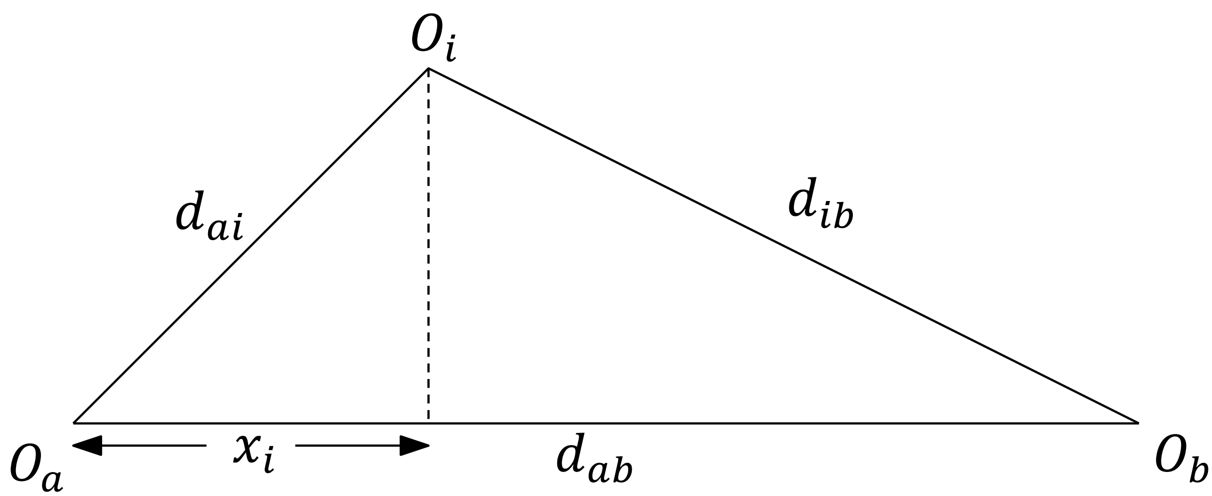

Fig. 7. "Cosine law" employed by FastMapSVM. The "cosine law" projection in a triangle.

defines a triangle with sides of lengths $d_{ai} = \mathcal{D}(O_a, O_i)$, $d_{ab} = \mathcal{D}(O_a, O_b)$, and $d_{ib} = \mathcal{D}(O_i, O_b)$ (Fig. 7). The sides of the triangle define its entire geometry, and the projection of O_i onto the line $\overline{O_a O_b}$ is given by

$$x_i = (d_{ai}^2 + d_{ab}^2 - d_{ib}^2)/(2d_{ab}). \tag{1}$$

FastMap sets the first coordinate of $\mathbf{p}_i$, the embedding of O_i, equal to x_i. In the subsequent $K-1$ iterations, FastMap computes the remaining $K-1$ coordinates of each object following the same procedure; however, the distance function is adapted for each iteration. In the first iteration, the coordinates of O_a and O_b are 0 and d_{ab}, respectively. Because these coordinates perfectly encode the true distance between O_a and O_b, the rest of $\mathbf{p}_a$ and $\mathbf{p}_b$'s coordinates should be identical for all subsequent iterations. Intuitively, this means that the second iteration should mimic the first one on a hyperplane that is perpendicular to the line $\overline{O_a O_b}$ (Fig. 8). Although the hyperplane is never explicitly constructed, it conceptually implies that the distance function for the second iteration should be changed for all i and j in the following way:

$$\mathcal{D}_{new}(O'_i, O'_j)^2 = \mathcal{D}(O_i, O_j)^2 - (x_i - x_j)^2 \tag{2}$$

in which O'_i and O'_j are the projections of O_i and O_j, respectively, onto this hyperplane, and $D_{new}(\cdot, \cdot)$ is the new distance function. The distance function is recursively updated according to Equation 2 at the beginning of each of the $K - 1$ iterations that follow the first.

Selecting Reference Objects. As described before, in each of the K iterations, FastMap heuristically finds the farthest pair of objects according to the distance function defined for that iteration. These objects are called pivots and are stored as reference objects. There are exactly $2K$ reference objects in our implementation because we prohibit any object from serving as a reference object more than once; however this restriction is not strictly necessary. Technically, finding the farthest pair of objects in any iteration takes $O(N^2)$ time. However, FastMap uses a linear-time "pivot changing" heuristic [16] to efficiently and effectively identify a pair of objects O_a and O_b that is very often the farthest pair. It does this by initially choosing a random object O_b and then choosing O_a to be the farthest object away from O_b. It then reassigns O_b to be the farthest object away from O_a.

In our adaptation of FastMap as a component of FastMapSVM, we require the farthest pair of objects O_a and O_b in each iteration to be of opposite classes. This maximizes the discriminatory power of the downstream SVM classifier. We achieve this requirement by implementing a minor modification of the pivot changing heuristic: We initially choose a random object O_b. We then choose O_a to be the farthest object away from O_b and of the opposite class. We finally reassign O_b to be the farthest object away from O_a and of the opposite class. It is implied that all previously used reference objects are excluded from consideration in all subsequent iterations when selecting reference objects.

For a test object not seen before, its Euclidean coordinates in the K-dimensional embedding can be computed by using only its distances to the reference objects. This is based on the reasonable assumption that the new test object would not preclude the stored reference objects

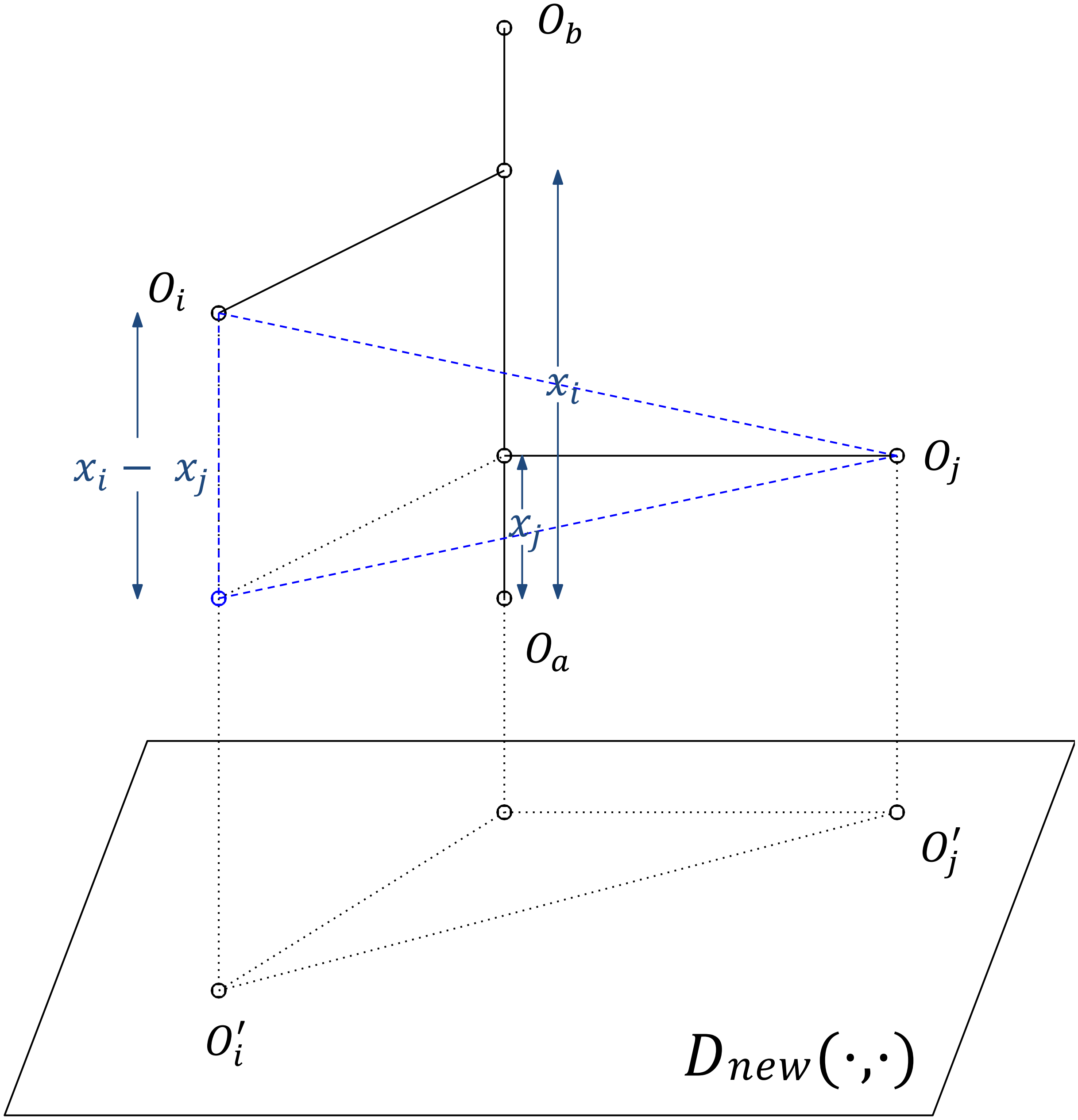

Fig. 8. Hyperplane projection employed conceptually by FastMapSVM. Projection onto a hyperplane that is perpendicular to $\overline{O_a O_b}$.

from being pivots if the K-dimensional Euclidean embedding was recomputed along with the new test object. In any case, the assumption is not strictly required since the stored reference objects are close to being the farthest pairs.

Choosing the Distance Function $\mathcal{D}$. The distance function should yield non-negative values for all pairs of objects and 0 for identical objects. We can use a variety of distance functions, such as the Wasserstein distance, the Jensen-Shannon divergence, or the Kullback-Leibler divergence. We can also use more domain-specific knowledge in the distance function, as described below.

In the Earthquake Science community, the normalized cross-correlation operator, denoted here by $\star$, is popularly used to measure similarity between two waveforms. For two zero-mean, single-component seismograms O_i and O_j with lengths n_i and n_j, respectively, and starting with index 0, the normalized cross-correlation is defined with respect to a lag τ as follows:

$$(O_i \star O_j)[\tau] \triangleq \frac{1}{\sigma_i \sigma_j} \sum_{m=0}^{n_i-1} O_i[m] \widehat{O}_j[m + \ell - \tau] \tag{3}$$

in which, without loss of generality, we assume that $n_i \geq n_j$. σ_i and σ_j are the standard deviations of O_i and O_j, respectively. Moreover, ℓ and $\widehat{O}_j$ are defined as follows:

$$\ell \triangleq \frac{n_j - n_j \pmod 2}{2} - \left(n_i \pmod 2\right)\left(1 - n_j \pmod 2\right) \tag{4}$$

and

$$\widehat{O}_j[m] \triangleq \begin{cases} O_j[m] & \text{if } 0 \leq m < n_j \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Equipped with this knowledge, we first define the following distance function that is appropriate for waveforms with a single component:

$$\mathcal{D}(O_i, O_j) \triangleq 1 - \max_{0 \leq \tau \leq n_i - 1} |(O_i \star O_j)[\tau]| \tag{6}$$

Based on this, we define the following distance function that is appropriate for waveforms with L components:

$$\mathcal{D}(O_i, O_j) \triangleq 1 - \frac{1}{L} \max_{0 \leq \tau \leq n_i - 1} \left| \sum_{l=1}^{L} (O_i^l \star O_j^l)[\tau] \right| \tag{7}$$

Here, each component O_i^l of O_i, or O_j^l of O_j, is a channel representing a 1-dimensional data stream. A channel is associated with a single standalone sensor or a single sensor in a multi-sensor array.

We use the distance function defined in Equation 7 with $L = 3$ for 3C seismograms. Our choice is motivated by the extensive use of similar equations in Earthquake Science to detect earthquakes using matched filters [13–15]. We will investigate other distance functions in future work.

Enabling SVMs and Kernel Methods. SVMs are particularly good for classification tasks. When combined with kernel functions, they recognize and represent complex nonlinear classification boundaries very elegantly [5]. Moreover, soft-margin SVMs with kernel functions [19] can be used to recognize both outliers and inherent nonlinearities in the data. While the SVM machinery is very effective, it requires the objects in the classification task to be represented as points in a Euclidean space. Often, it is very difficult to represent complex objects like seismograms as precise geometric points without introducing inaccuracy or losing domain-specific representational features. In such cases, NNs have been more effective than SVMs. FastMapSVM revitalizes SVM technology for classifying complex objects by leveraging the following observation: Although it may be hard to precisely describe complex objects as geometric points, it is often relatively easy to precisely compute the distance between any two of them. FastMapSVM uses the distance function to construct a low-dimensional Euclidean embedding of the objects. It then invokes the full power of SVMs. The low-dimensional Euclidean embedding also facilitates a perspicuous visualization of the classification boundaries.

Implementing FastMapSVM. We have implemented FastMapSVM and have made it publicly accessible in a Python package available at: `https://github.com/malcolmw/FastMapSVM`. The most expensive computations, i.e., evaluations of the distance function, are parallelized using Python's built-in `multiprocessing` module, which allows for the concurrent execution of multiple threads on a single host. FastMapSVM requires as input (a) the labeled training data set, (b) the distance function, and (c) a location to store the resulting trained model. We used the `scikit-learn` SVM implementation and conducted a grid search for the optimal SVM hyperparameters.

Conclusions and Future Work

In this paper, we advance FastMapSVM—an interpretable ML framework that combines the complementary strengths of FastMap and SVMs—as an advantageous alternative to existing methods, such as NNs, for classifying complex objects. FastMapSVM offers several advantages. First, it enables domain experts to incorporate their domain knowledge using a distance function. This avoids relying on complex ML models to infer the underlying structure in the data entirely. Second, because the distance function encapsulates domain knowledge, FastMapSVM naturally facilitates interpretability and explainability. In fact, it even provides a perspicuous visualization of the objects and the classification boundaries between them. Third, FastMapSVM uses significantly smaller amounts of time and data for model training compared to other ML algorithms. Fourth, it extends the applicability of SVMs and kernel methods to domains with complex objects.

We demonstrated the efficiency and effectiveness of FastMapSVM in the context of classifying seismograms. On the STEAD data set, we showed that FastMapSVM performs comparably to state-of-the-art NN models in terms of precision, recall, and accuracy. It also uses significantly smaller amounts of time and data for model training compared to other methods. On the

Ridgecrest data set, we first demonstrated the robustness of FastMapSVM against noisy perturbations. We then demonstrated its ability to reliably detect new microseisms that are otherwise difficult to detect.

In future work, we expect FastMapSVM to be viable for classification tasks in many other real-world domains. In Earthquake Science, we will apply FastMapSVM to analyze and learn from data obtained during temporary deployments of large-N nodal arrays and distributed acoustic sensing. In Computational Astrophysics, we anticipate the use of FastMapSVM for identifying galaxy clusters based on cosmological observations. In general, the efficiency and effectiveness of FastMapSVM also make it suitable for real-time deployment in dynamic environments.

Our implementation of FastMapSVM is publicly available at: `https://github.com/malcolmw/FastMapSVM`.

Acknowledgments

This work at the University of Southern California is supported by DARPA under grant number HR001120C0157 and by NSF under grant number 2112533. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the sponsoring organizations, agencies, or the U.S. Government.

The authors declare that they have no competing interests.

MW and TKSK conceived the general concept of combining FastMap with SVMs and kernel methods for object classification, independent of [4]. MW also refined the concept in the Earthquake Science domain, implemented the FastMapSVM method presented here, conducted the experiments, and drafted the manuscript. KS and AL conducted various experiments using FastMapSVM in support of those presented here. NN and TKSK provided critical guidance and oversight to the project. AL, TKSK, NN, and KS contributed significantly to manuscript

revision.

STEAD data are publicly available at `https://github.com/smousavi05/STEAD`. Ridgecrest data are publicly available at `https://scedc.caltech.edu`.

Supplementary Material

Figures S1, S2, S3, and S4.

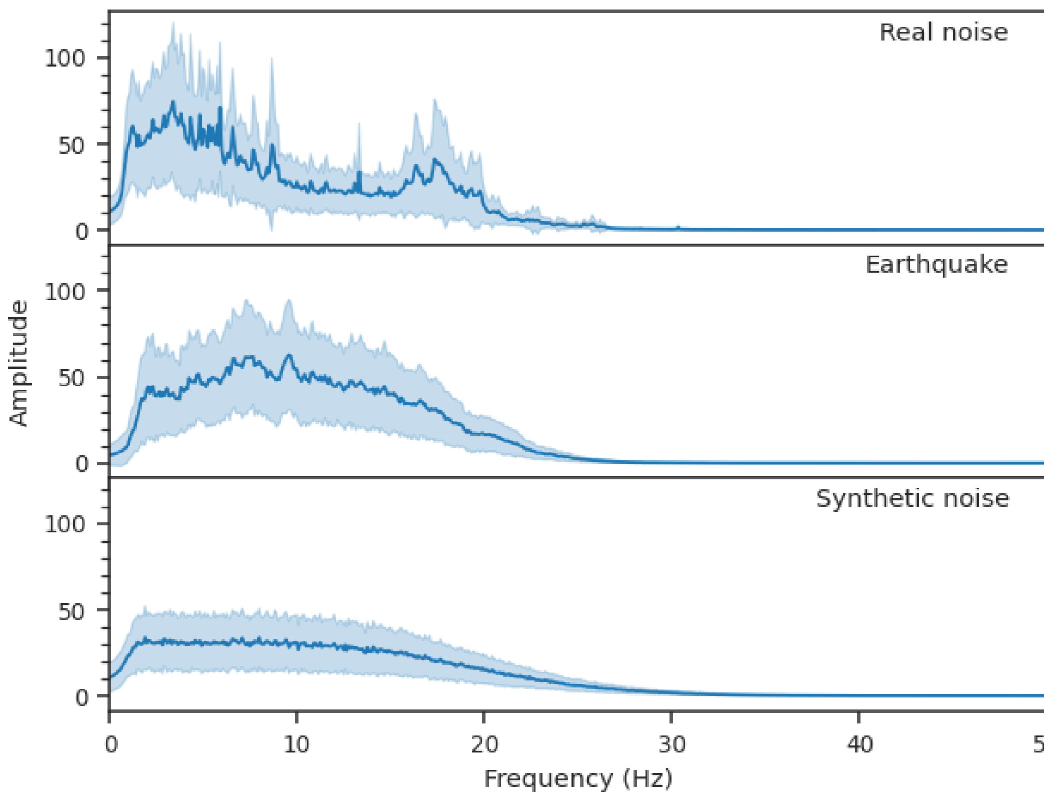

Fig. S1. Average seismogram frequency spectra. Shows the typical frequency spectra of real noise, earthquake signals, and added synthetic noise.

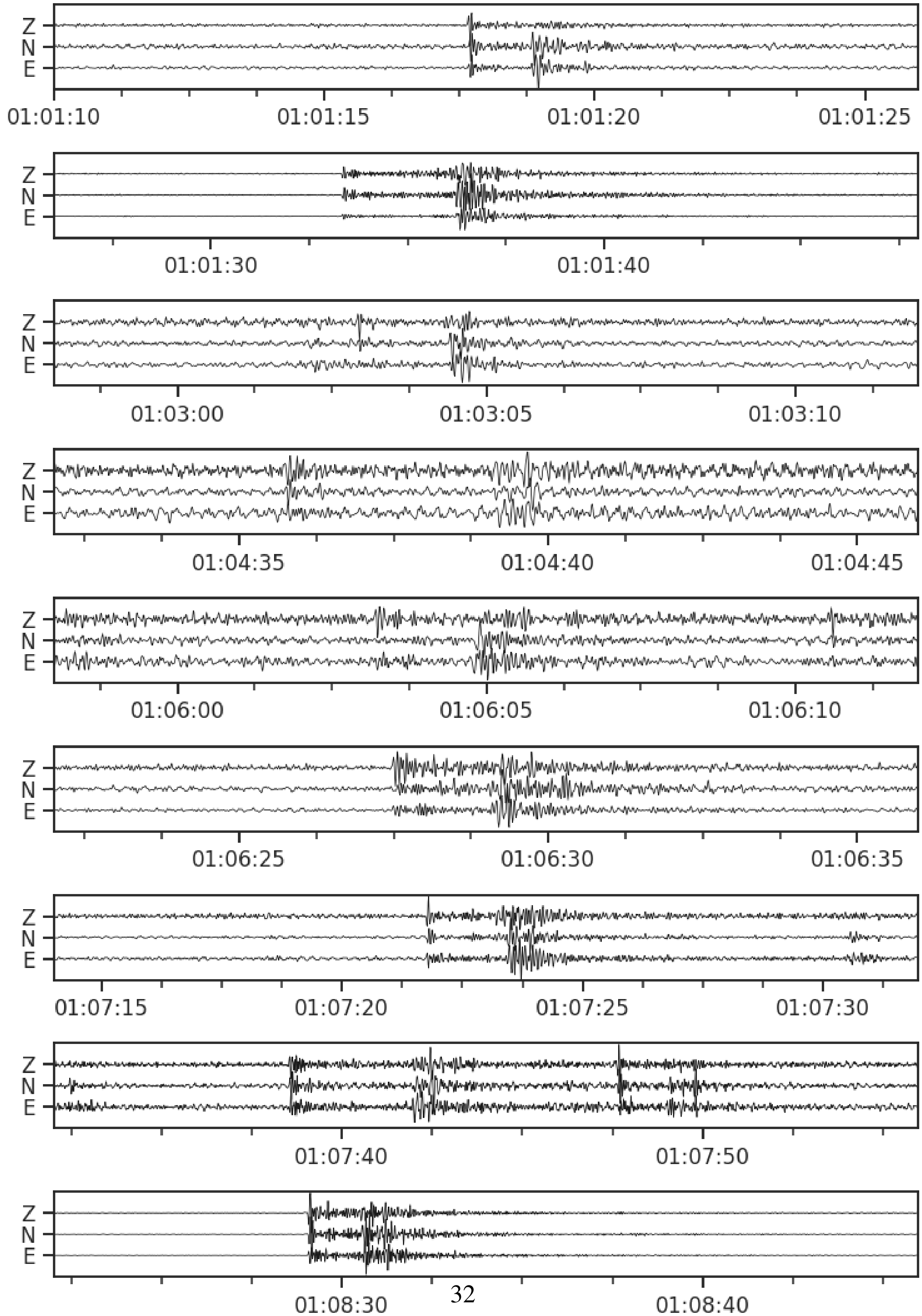

Fig. S2. Earthquakes identified by automatic FastMapSVM scan for earthquakes. Shows easily discernible earthquake signals.

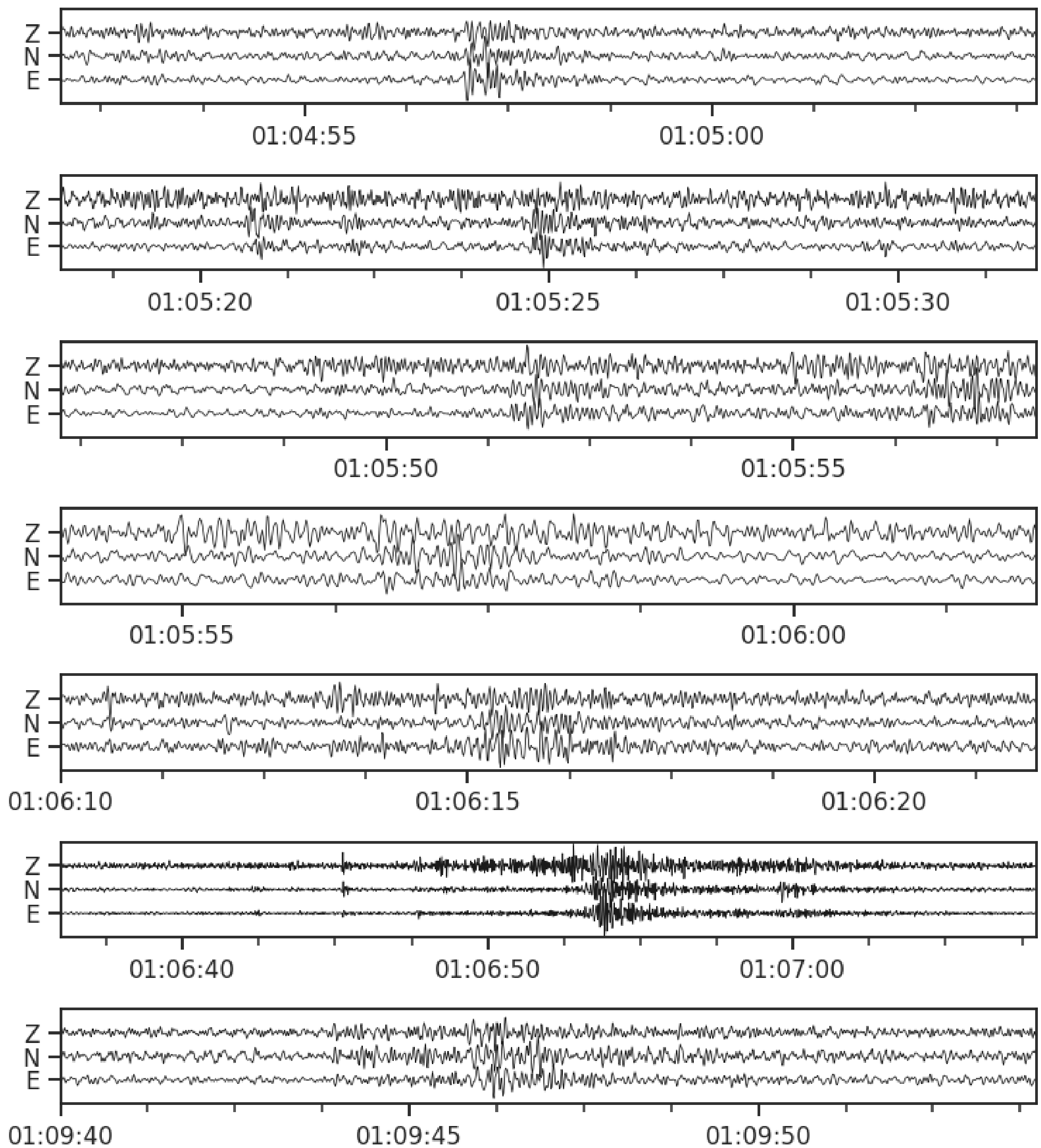

Fig. S3. Potential earthquakes identified by automatic FastMapSVM scan for earthquakes. Shows earthquake signals with low signal-to-noise ratio.

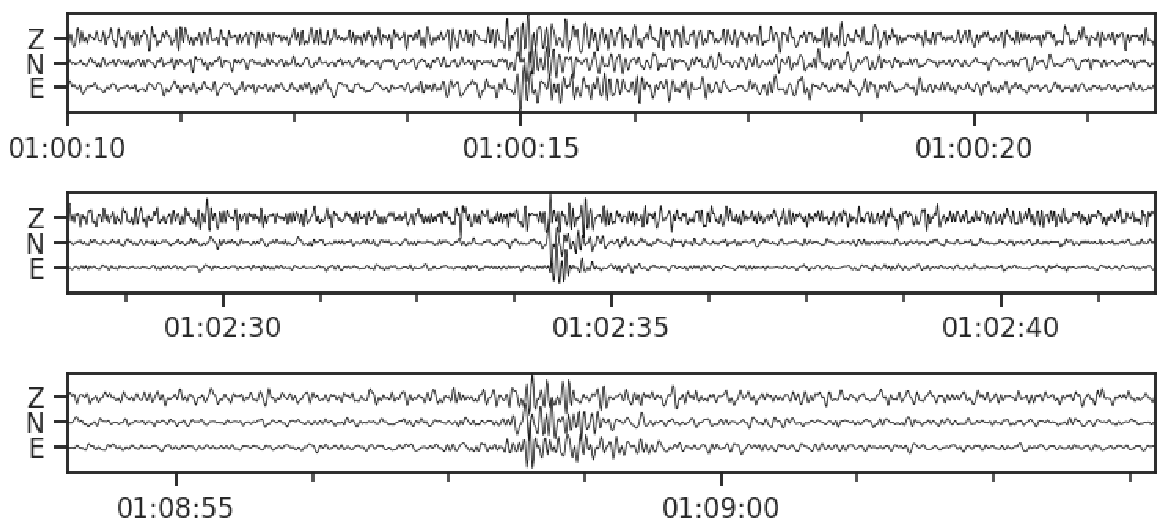

Fig. S4. Ambiguous signals identified as earthquakes by automatic FastMapSVM scan. Shows ambiguous signals that may or may not be from an earthquake source.