Math Geosci
https://doi.org/10.1007/s11004-022-10037-7

SPECIAL ISSUE

Ensemble and Self-supervised Learning for Improved Classification of Seismic Signals from the Åknes Rockslope

Daesoo Lee[1] · **Erlend Aune**[2] · **Nadège Langet**[3] · **Jo Eidsvik**[1]

Received: 6 April 2022 / Accepted: 30 October 2022

Abstract A case study with seismic geophone data from the unstable Åknes rock slope in Norway is considered. This rock slope is monitored because there is a risk of severe flooding if the massive-size rock falls into the fjord. The geophone data is highly valuable because it provides 1000 Hz sampling rates data which are streamed to a web resource for real-time analysis. The focus here is on building a classifier for these data to distinguish different types of microseismic events which are in turn indicative of the various processes occurring on the slope. There are 24 time series from eight 3-component geophone data for about 3500 events in total, and each of the event time series has a length of 16 s. For the classification task, novel machine learning methods such as deep convolutional neural networks are leveraged. Ensemble prediction is used to extract information from all time series, and this is seen to give large improvements compared with doing immediate aggregation of the data. Further, self-supervised learning is evaluated to give added value here, in particular for the case with very limited training data.

Keywords Classification · Ensemble prediction · Self-supervised learning · Geohazards · Åknes rockslope

✉ Daesoo Lee
daesoo.lee@ntnu.no

[1] Norwegian University of Science and Technology, Trondheim, Norway

[2] BI Norwegian Business School, Norwegian University of Science and Technology, Oslo, Norway

[3] Norwegian Seismic Array, Oslo, Norway

Published online: 30 November 2022

1 Introduction

With the advent of new sensor technology and current opportunities to stream data by wireless communication, there is an enormous potential for geohazards monitoring. Pressure on construction work such as roads and tunnels in remote areas as well as challenges with climate change leading to more rain and ice only increases the demand for monitoring tools. However, to extract useful knowledge from this kind of data one needs algorithms that can process the massive-size data and create meaningful classifications or predictions. Only then can the monitoring networks be applicable for decision support such as geohazard warning systems.

Such automatic prediction models can be built with machine learning methods. Bernardi et al. (2021) review approaches for satellite data in hazard warning systems. The automatic classification of microseismic signals was first developed in the field of volcano seismology via the use of different supervised machine learning methods, all relying on the extraction of features from the time series, such as neural networks (e.g. Falsaperla et al. 1996; Langer et al. 2003; Scarpetta et al. 2005; Langer et al. 2006; Ibs-von Seht 2008; Curilem et al. 2009), Hidden Markov Models (e.g. Benítez et al. 2006; Ibáñez et al. 2009), Support Vector Machine (e.g. Masotti et al. 2006; Langer et al. 2009; Malfante et al. 2018) or Random Forest (e.g. Hibert et al. 2017; Maggi et al. 2017; Malfante et al. 2018). Similar methods were used to classify seismic signals originating from unstable areas (e.g. Hammer et al. 2013; Dammeier et al. 2016; Provost et al. 2017; Vouillamoz et al. 2018; Feng et al. 2020).

The focus of this paper is on seismic geophone monitoring for rock slope and earthquake classification. A case study from the unstable Åknes rock slope is considered. Currently, the streamed geophone data from Åknes is processed on a computer which sends a message to an administrator if the amplitude of the seismic signal exceeds a threshold. The skilled administrator then looks at the signal and based on experience, one can very likely classify the event into a certain kind of rock fall or earthquake. In the future, one aims to go beyond this level and instead automatically classify events, so that efforts can be focused on what has happened on the slope, and why this might be the case from a geoscience perspective.

The main contribution of this work is to use and develop new statistical machine-learning methods for the seismic event classification problem. One of the most popular convolutional neural networks (CNNs), ResNet (He et al. 2016), is used here, and ensemble prediction is implemented over the 24-variate time series data instead of the standard aggregation to form a single spectrogram for each event. Ensemble prediction substantially improves on the state-of-the-art classification for this Åknes case study. Further, recently developed self-supervised learning (SSL) methods (Lee and Aune 2021; Tonekaboni et al. 2021) are tested on the geophone dataset, and promising results indicate that this can be valuable for such applications with limited labeled training data. The current work extends recent work by Langet and Silverberg (2022), who show extensive data analysis and classifying results of Åknes geophone data.

In Sect. 2 background on the Åknes rock slope case is provided. In Sect. 3 the terminology of event classification and multivariate time series data is presented. In Sect. 4 the current state of the art of deep machine learning to such events classification is outlined. In Sect. 5 the suggested statistical machine learning approach for improved classifying of events from multivariate non-stationary time series data is developed. In Sect. 6 results of the suggested method and others are shown for the Åknes geophone data case. In Sect. 7 closing remarks and points to further work are provided.

2 Background on the Åknes Rock Slope

The Åknes rock slope is located south of Stranda in Western Norway, see Fig. 1, by the fjord going in to Hellesylt and Geiranger.

The steep slope goes from the fjord up to nearly 1500 m altitude. The unstable rock slope is limited by strike-slip faults to the North-East and South-West, and a fracture/backscarp as the upper limit. The upper fracture has lately been widening at about 2 cm per year, and this has caused concern about rock fall geohazards.

The risk-prone rock slope volume is up to 50 million m^3. Even though three dimensional numerical modeling of the slope has been conducted (Gharti et al. 2012), it has shown difficult to map the subsurface properties of the slope due to large heterogeneities in the subsurface seismic velocity model. There are hence large uncertainties associated with the unstable volume, and how it will collapse into the fjord. But even with a much smaller volume, rock collapse would create a tsunami with severe impacts on the local communities (Harbitz et al. 2014), particularly considering its location in the narrow Norwegian fjords. The natural beauty of this fjord region has made it a UNESCO World Heritage Site. The area attracts nearly a million tourists per year and about 200 cruise ships pass by the Åknes rock slope every year.

Because of the widening of the upper crack at Åknes, the rock slope has been extensively monitored for several years. In these efforts one has gathered different data sources such as surface crack displacements sensors (Nordvik and Nyrnes 2009) and the displacements connections to meteorological data (Grøneng et al. 2011), InSAR data (Bardi et al. 2016) and seismic geophone data (Roth et al. 2006). Here, the focus is mainly on the geophone data.

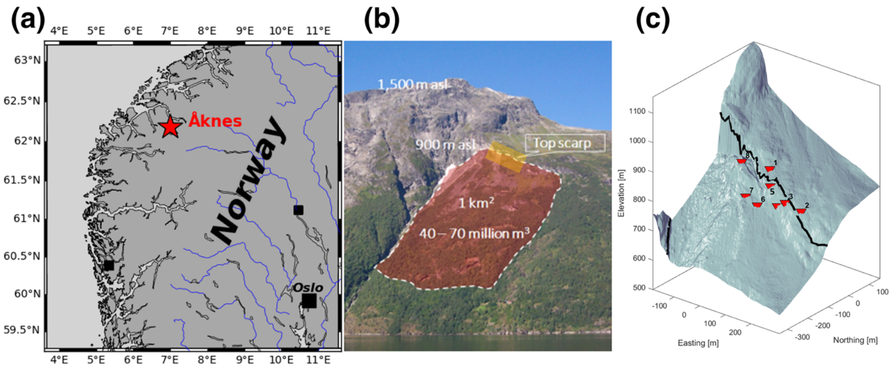


Fig. 1 **a** Location of the Åknes rockslope in Norway. **b** Photo of the slope taken from the opposite site of the fjord (Roth and Blikra 2009). The unstable area is highlighted by the red shade. **c** Digital elevation model corresponding approximately to the orange-shaded area in **b** and location of the eight geophones (red inverse triangles). The black line delineates the backscarp (Langet and Silverberg 2022)

Table 1 Class types with the corresponding numbers of samples in the dataset. The dataset is available within the source code. The data has been acquired over many years

Class name	Number of samples
Noise	8
Regional	292
Rockfall	215
Slope high-frequency (HF)	448
Slope low-frequency (LF)	218
Slope multi	207
Slope tremor	212
Spike	218
Unlabeled	1611

The data with the valid classes (i.e. Noise to Spike) had been acquired from 2007 to 2020 and the data with the Unlabeled class was acquired throughout 2021

Altogether, there are eight geophones in the slope, each of them with three-component sensors. Detailed locations of the geophones can be found in Fig. 1. The geophone data passively register seismic activity at Åknes. The data are streamed to a computer server and this hence provides a continuous-time monitoring system for detecting potential rock fall or movements on the rock slope. When there is a seismic signal above a pre-specified amplitude threshold, a window of 16 s of geophone data is stored. Based on expert opinion, every event is then labeled into one of eight classes, as described in Table 1. However, it should be noted that manual labeling is not straightforward for most of the events. Thus, manually defined classes should not be used as absolute ground truth and results should be interpreted with caution. Therefore, one should be aware that no classification model can achieve 100% classification accuracy on a test set. Discussing these processes is out of the scope of this work, but more details can be found in Langet and Silverberg (2022). There is by now a rich database of geophone observations and associated rock fall class interpretation for various seismic events at Åknes (Langet and Silverberg 2022). This provides an excellent test case to understand and try out new machine learning methods for classifying rock slope events, and by building a reliable encoder and classifier, one can achieve automatic classification of such seismic geophone data events in the future. This would provide a low-cost and highly valuable decision-support tool in the context of geohazard warning systems.

3 Seismic data and Methods for Time Series Events Classification

Examples of 16 s geophone data recordings are shown in Fig. 2. The sampling frequency is 1000 Hz, so the time series data consist of length-16, 000 vectors for each of the eight geophones with three (x, y, z) components (24 time series in total). These data represent non-stationary time series, where the amplitude and frequency content clearly change over the 16 s time interval. It is not obvious how to summarize such an

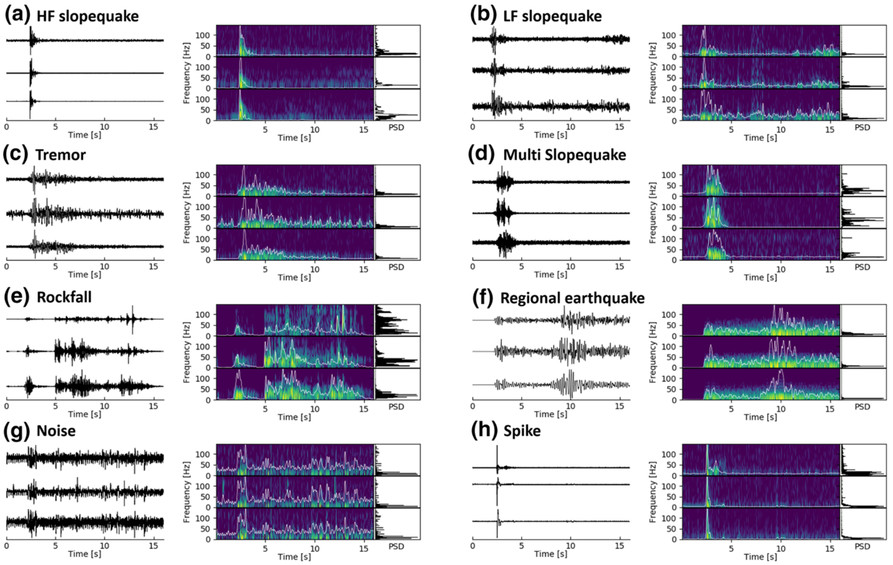


Fig. 2 Example time series (left) associated with their spectrograms (right) for each class (**a**–**h**). Vertical component records of the same event at three chosen geophones are shown for each class. White lines on spectrograms delineate signal envelopes while plots on the right show their Power Spectral Density (PSD)

image by simple statistics such as mean, variance, correlation, dominating frequency, or other measures.

A common technique of analyzing such data is to compute spectrograms (Cohen 1989). This essentially entails taking a rolling window short-time Fourier transformation of the data. The magnitudes for each frequency and time will then indicate the core frequency contents of the signal as a function of the 16 s time interval. Figure 2 shows example spectrograms associated with the time series.

There are apparent differences in the signals. The most obvious one is probably the signal duration which can last less than 3 s (Fig. 2a, b, d, h) or, on the contrary, last longer or even span the full duration of the record (Fig. 2c, e, f, g). The shape of the signals, represented by their envelopes, is also an important characteristic and displays either a main peak of energy (e.g. Fig. 2a, h) or several peaks of energy distributed over time (e.g. Fig. 2c, e, f). Lastly, the frequency content of the signals exhibits clear differences from one type of event to another, with some classes reaching high frequencies up to 80 Hz or even more (e.g. Fig. 2a, d, e, h) while some others are confined to lower frequencies (< 20 Hz, e.g. Fig. 2b, c, f). Moreover, variability in the time series, and consequently in the spectrograms, can be observed depending on the sensor which recorded the event. This is particularly visible for the rockfall event (Fig. 2e) In this example, the relative amplitudes of the different bursts in the time series are variable. Such variability from a geophone to another is mostly due to the event source location and its distance to the recording stations. Seismic waves are indeed affected by the properties of the medium in which they propagate.

Since each geophone is composed of 3 components (one vertical component and two horizontal components) there is complementary information on the recorded waves. For example, P-waves are polarised in the vertical plane while S-waves are polarised in the horizontal plane. However, in practice at Åknes, the events are very often so short that it is difficult to really distinguish P- and S-waves.

With the 24 time series data for each event, a natural question is then ’how one can use all this information?’ And in doing so, one aims to provide an improved model for classifying the various seismic events. There must clearly be added value in the multivariate time series responses. A first attempt is likely to align the time series and conduct some kind of average summary statistic or spectrogram for all time series associated with an event. Another way forward is to test ensemble prediction where the classes are predicted separately over geophones and components. Then, 24 predictions can be made from the time series. The predictions can be averaged to obtain a final prediction. With this approach, not only all the information can be maximally utilized as there is no information loss caused by pre-aggregation or filtering methods, but also the better uncertainty assessment can be captured in the final prediction as a nature of the ensemble approach. Some other way for the maximal use of the information lies in effective dimension reduction in a neural network encoder. From a perspective of an encoder that projects an input onto a lower dimension space, it plays a role of a dimension reduction model. Then, the dimension reduction should be conducted such that the vector elements in the reduced dimension capture all the meaningful information while discarding unnecessary information. To achieve that, decorrelation between each vector element and maintaining a certain variance within each vector element turn out to be effective (Bardes et al. 2021; Lee and Aune 2021).

4 Related Work in Deep Learning

Two main items are addressed here: (i) CNN architecture with a focus on the promising ResNet architecture. Key elements of ResNet and related methods are described in Sect. 4.1. (ii) SSL which is an intermediate supervised and unsupervised learning, relying on a pre-trained model, which often returns good performance even with small amounts of labeled data. Several mainstream SSL methods of relevance are introduced and explained in Sect. 4.2.

4.1 ResNet

In the past decade, there have been numerous architectural advances for CNN models since the first suggested architecture LeNet (LeCun et al. 1998). As ResNet is the main architecture in this study, it is briefly explained in the following subsection.

AlexNet (Krizhevsky et al. 2017) was one of the first deep CNN architectures that largely contributed to the starting era of CNN in computer vision. AlexNet competed in the ImageNet competition (ImageNet Large Scale Visual Recognition Challenge) in 2012 and won the competition with a large margin from the second best. One of its main features is a larger depth of the model which was essential for its high performance. The model processes an image by downsizing the height and width of

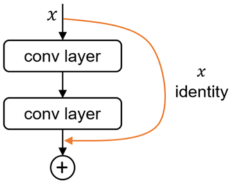


Fig. 3 Skip connection is highlighted in orange. It is the key component in ResNet compared with AlexNet, and it helps with the challenge of vanishing gradients in deep models

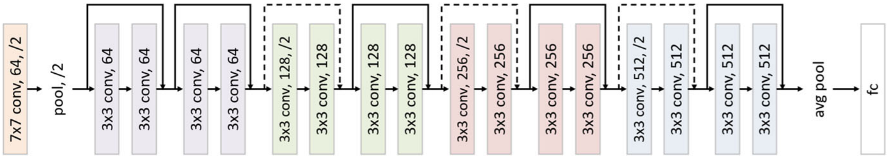


Fig. 4 Architecture of ResNet18. ($n \times n$ conv, m) denotes a convolutional layer with the kernel size of $n \times n$ and output channel size of m, /2 denotes downsampling by 2, pool denotes a max pooling layer, avg pool denotes a global average pooling layer, and fc denotes a fully connected layer

an input image while increasing the channel dimension, and this allows one to capture richer features of the image. Many CNN model architectures have been proposed since AlexNet. Some mainstream CNN models are Inception (GoogLeNet) (Szegedy et al. 2015), VGGNet (Simonyan and Zisserman 2014), and ResNet (He et al. 2016). In the latter, it is shown that ResNet outperforms both VGG and Inception.

ResNet is one of the most commonly used CNN architectures in computer vision at the moment (Chen et al. 2020; Grill et al. 2020; Zbontar et al. 2021; Bardes et al. 2021; Lee and Aune 2021; Liu et al. 2022). Its overall convolutional architectural modeling is similar to AlexNet but key difference is the use of skip connection shown in Fig. 3. The architecture of ResNet18 is presented in Fig. 4. The skip connection prevents a vanishing gradient problem connected to deep models. ResNet hence enables scalable models and faster training. He et al. (2016) showed that ResNet could achieve a major performance improvement compared to state-of-the-art models at that time.

4.2 Self-supervised Learning

The three main approaches for representation learning are shown in Fig. 5. In SSL, a single image is augmented into two images, and representations of the two augmented images, z_0 and z_1, are pulled together in the embedding space. Intuitively, it can be thought that although the absolute pixel values are different in the two augmented images, they carry the same semantics of the dog. Hence, the two representations are pulled in the embedding space. The representation typically refers to an output vector after an encoder, and learning process of the representation is termed representation learning. A goal of the representation learning is to train the representation such that it can capture informative features of an input. The three main approaches are supervised learning, unsupervised learning (e.g. auto-encoder style), and SSL frameworks. But, the supervised learning approach is limited as it requires a labeled dataset. Although

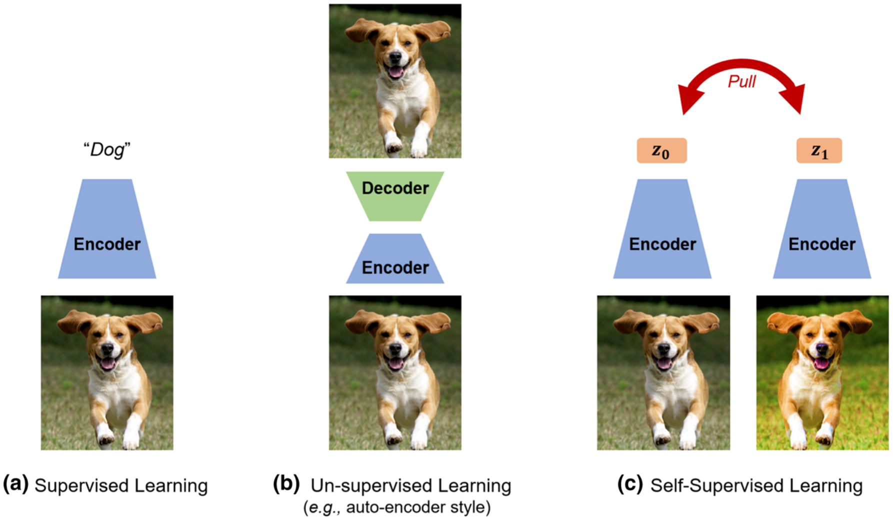

"Dog"
Encoder
Decoder
Encoder
Pull
z_0
z_1
Encoder
Encoder
(a) Supervised Learning
(b) Un-supervised Learning
(e.g., auto-encoder style)
(c) Self-Supervised Learning

Fig. 5 The three main approaches for representation learning: **a** supervised learning, **b** unsupervised learning (e.g. auto-encoder style), and **c** SSL frameworks

an auto-encoder can learn the representation without a labeled dataset by reconstructing a given input, as its task is specific to reconstruction, the quality of the learned representation is often not so universal that it can be used for various downstream tasks. SSL takes two augmented data from a single input and pulls the representations of the two augmented data in the embedding space so that the representations are learned explicitly in the embedding space. Another advantage is that one can utilize domain expertise to design an augmentation method so that semantically more-meaningful representations can be learned. Generally, SSL results in better-quality representations than an auto-encoder, and it is an actively studied research area. After an encoder is (pre-)trained by SSL, the pre-trained encoder can be used for various downstream tasks. In Figs. 5 and 6, images are used as input for easy understanding of representation learning and SSL, but other forms of input can be used, such as time series and spectrograms as are analyzed in the results here.

The recent mainstream SSL frameworks can be divided into two main categories: (i) contrastive learning, (ii) non-contrastive learning. Some well-known contrastive learning methods are MoCo (He et al. 2020) and SimCLR (Chen et al. 2020). In those methods, there are a reference sample, a positive sample, and a negative sample. The reference and positive samples form a positive pair, and the reference and negative samples form a negative pair. Then, those contrastive methods learn representations by pulling the representations of the positive pairs together and pushing those of the negative pairs apart. However, these methods require a large number of negative pairs per positive pair to learn representations effectively. To eliminate the need for negative pairs, non-contrastive learning methods such as BYOL (Grill et al. 2020), SimSiam (Chen and He 2021), Barlow Twins (Zbontar et al. 2021), VICReg (Bardes et al. 2021), and VIbCReg (Lee and Aune 2021) have been proposed. The non-contrastive learning methods use positive pairs only, and training of the networks could be simplified. The

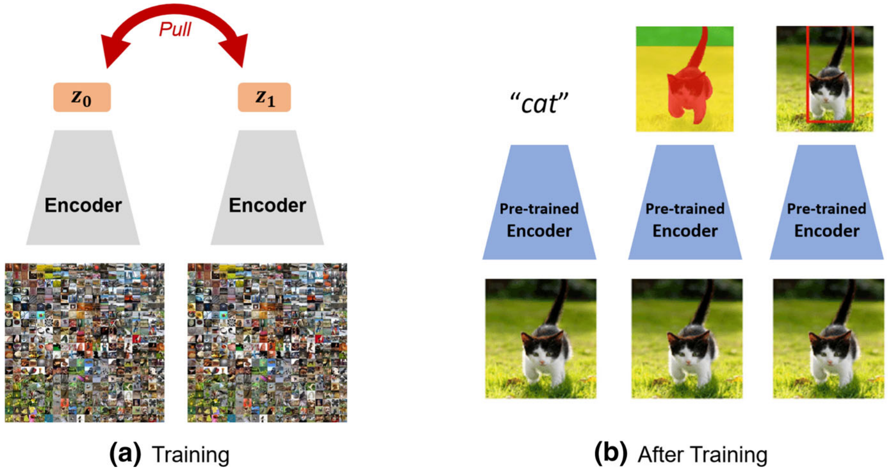


Fig. 6 Illustration of SSL. **a** Fig. 5 showed SSL on a single sample and here it shows SSL on a dataset. **b** After pre-training the encoder by SSL, the pre-trained encoder can understand various kinds of visual concepts. Thus, it can be used for various downstream tasks such as classification, segmentation, and object detection with fine-tuning

non-contrastive learning methods were able to outperform the existing contrastive learning methods in terms of quality of learned representations. In particular, VICReg and VIbCReg outperform other competing SSL methods by having effective feature decorrelation (Bardes et al. 2021; Lee and Aune 2021). Simplified illustrations of the introduced SSL methods are presented in Fig. 7. It should be noted that time series can be used as an input instead of an image while the entire methodological framework remains intact.

VICReg encodes two different views of the same input with an encoder. The different views are created through data augmentation methods. Then, the two outputs from the encoder are further projected into a higher dimension by a projector. The loss function of VICReg forms on the two outputs from the projector. The loss function consists of variance, invariance, and covariance losses. The variance loss keeps variance of the output vector's each component to be larger than 1.0 along the batch dimension. The invariance loss minimizes the Euclidean distance between the two output vectors. The covariance loss decorrelates between components of the output vector. By minimizing the loss function, VICReg can effectively learn the representations. VIbCReg improves VICReg by introducing better covariance with an iterative normalization layer (Huang et al. 2019) and normalized covariance loss.

Temporal Neighborhood Coding (TNC) (Tonekaboni et al. 2021) is a recent SSL method to learn representations for non-stationary time series. It learns time series representations by ensuring that a distribution of signals from the same neighborhood is distinguishable from a distribution of non-neighboring signals. It was developed to address time series in the medical field, where modeling the dynamic nature of time series data is important. An overview of the TNC framework is presented in Fig. 8. While the figure shows time series as input, it should be noted that a spectrogram can be used as input instead of time series, which is the case in our study.

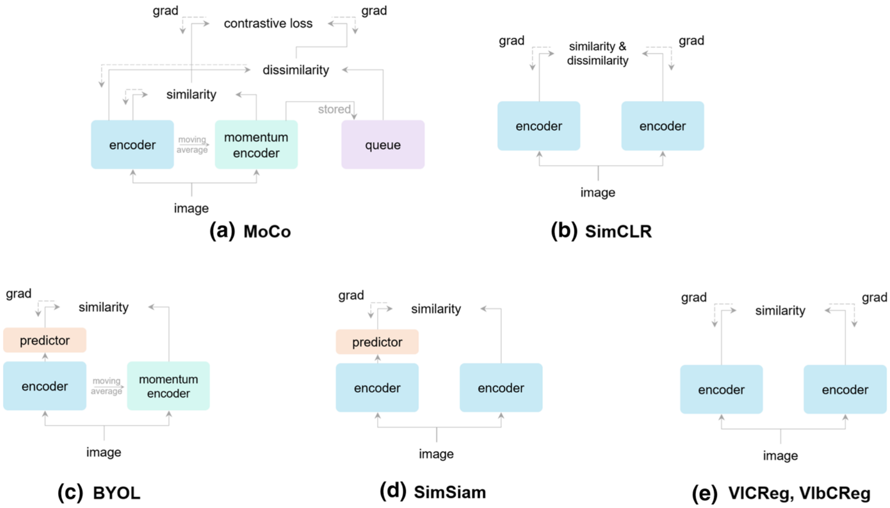


Fig. 7 Simplified illustrations of SSL methods. grad denotes gradient and similarity denotes similarity between two output vectors from two encoders that share their weights and the similarity is optimized to be reduced between the two augmented images. The dashed lines indicate the gradient propagation flow. Therefore, the lack of a dashed line denotes stop-gradient

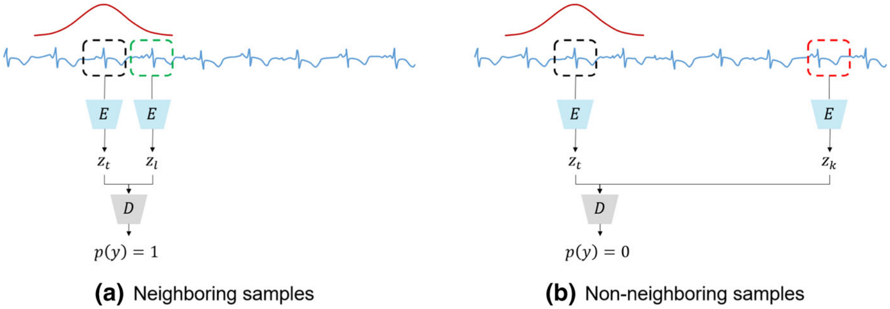


Fig. 8 Overview of the TNC framework components. E, z, and D denote an encoder, representation, and discriminator, respectively, where the discriminator is a shallow neural network for binary classification. TNC encodes the distinguishable distributions between neighboring samples and non-neighboring samples (**a**) by training the discriminator to predict 1 for representations of neighboring samples (denoted as z_t and z_l) and (**b**) 0 for representations of non-neighboring samples (denoted as z_t and z_k)

5 Proposed Method

There are two main components in this paper: (i) Ensemble prediction, (ii) VNIbCReg (Variance Neighboring-Invariance better-Covariance Regularization). The ensemble prediction is a method that forms separate classifications for each subset of data. In the current context, this means 24 time series (8 geophones with 3 axes). An average of the 24 predictions is used as a final result. Our experiments show that it brings a significant improvement in classification performance. VNIbCReg is an SSL method proposed for the event classification in this paper. Its framework is based on VIbCReg and TNC. By combining VNIbCReg with TNC, it is shown that VNIbCReg learns quality representations in an unsupervised learning manner, and therefore a model can be effectively pre-trained with VNIbCReg.

Ensemble Prediction The current motivation behind the ensemble prediction is as follows: (i) First assumption: Each seismic time series should carry sufficient information to be able to make a reasonable prediction. Then, by employing the notion of the ensemble, classification performance can be improved. (ii) Second assumption: If spectrograms of the 24 time series are aggregated, useful information on some of the 24 spectrograms might be weakened. To highlight each spectrogram's information content, each one is processed by the model separately, as shown in the pseudocode.

A big advantage of ensemble prediction is to get a bigger dataset: As each spectrogram of the 24 time series is treated as an individual sample in the ensemble prediction's training process, the dataset with the ensemble prediction is practically 24 time larger. Unlike early approaches in signal processing (Ovanger 2021; Langet and Silverberg 2022), which take averages early on to improve the signal-to-noise ratio, ensemble methods postpone the averaging and rather attempt to extract as much information as possible from each data sample. In principle, the approach could be used over several models as well as for different parts of the dataset (Hastie et al. 2009).

Pseudocode for the ensemble prediction is presented in Algorithm 1. It mainly consists of two parts: training and testing. While most of the steps are similar, the main difference is that during training, a single x_i is randomly picked and its corresponding $\hat{y}$ is used as the prediction, and during testing, the average of $\hat{y}$-s from all x_i-s in X is used as the prediction.

Algorithm 1 Pseudocode for the ensemble prediction

```
while Training do
    Get $X$ and $y$                                  ▷ mini-batch; $X = \{x_1, x_2, \cdots, x_{24}\}$; $x$ is time series
    Randomly pick $x_i$ from $X$
    $x_i \leftarrow$ Convert to spectrogram$(x_i)$    ▷ Now, $x$ is a spectrogram
    $\hat{y} \leftarrow f_C(f_E(x_i))$               ▷ $f_E$ and $f_C$ denote an encoder and a classifier
    Optimize $L(y, \hat{y})$                         ▷ $L$ is a cross-entropy loss
end while

while Testing do
    Get $X$ and $y$
    $X \leftarrow$ Convert to spectrogram$(X)$
    $\bar{\hat{y}} \leftarrow 0$
    for $x_i$ from $X$ do
        $\hat{y} \leftarrow f_C(f_E(x_i))$
        $\bar{\hat{y}} \leftarrow \hat{y} + \hat{y}$
    end for
    $\bar{\hat{y}} = \bar{\hat{y}}/24$
end while
```

VNIbCReg: Variance Neighboring-Invariance better-Covariance Regularization As mentioned in Sect. 4.2, VNIbCReg can be viewed as VIbCReg with TNC. VIbCReg is

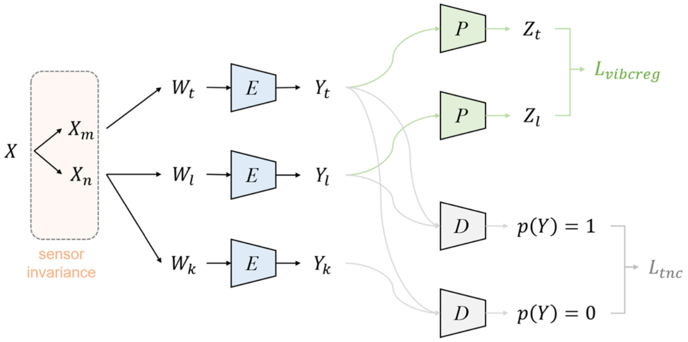


Fig. 9 Overall framework of VNIbCReg

effective in representation learning because samples with different classes can be well separated in the embedding space. Despite the effectiveness of VIbCReg, it was not designed to encode temporal transition in the (signal) representation. The necessity of encoding the temporal transition can be observed given several samples from our dataset in Fig. 2. To compensate for the lack of ability to encode the temporal transition in VIbCReg, TNC is adopted into VIbCReg, resulting in VNIbCReg.

The overall framework of VNIbCReg is presented in Fig. 9. Notation is clarified next, along with the required loss functions. Here, X denotes an input of the spectrograms where $X \in \mathbb{R}^{B \times 24 \times H \times W}$ (B: batch, H: height, and W: width), and m and n denote randomly-selected indices for a single time series among the 24 time series. Therefore, X_m and X_n have dimension $\mathbb{B}^{B \times H \times W}$. A cropped window is denoted by W, where W_t and W_l are in the same neighborhood while W_t and W_k are non-neighboring. It should be noted that W_t, W_l, and W_k are randomly chosen while keeping their spatial relations to each other. An illustration of W_t, W_l, and W_k is presented in Fig. 10. Further, E, P, and D denote an encoder, the projector in VIbCReg, and the discriminator in TNC, respectively. Loss functions are denoted by L. Note that an iterative normalization layer in VIbCReg is omitted for simplicity in the figure and the same coloring for models such as E, P, and D represents the shared weights between the same colored models. As for the discriminator's input, two Y-s are concatenated and used as input. As for Y and Z in the loss functions, the following notation is used: $Y = [y_1, \dots, y_B]^T \in \mathbb{R}^{B \times F_y}$ and $Z = [z_1, \dots, z_B]^T \in \mathbb{R}^{B \times F_z}$, where F_y and F_z denote feature size of Y and Z, respectively.

The additive loss is used for the two parts

$$L_{vnibcreg} = L_{vibcreg} + L_{tnc}. \tag{1}$$

Here, the first part on the right side of Eq. (1) consists of

$$L_{vibcreg} = \lambda s(Z_t, Z_l) + \mu\{v(Z_t) + v(Z_l)\} + \nu\{c(Z_t) + c(Z_l)\}, \tag{2}$$

$$s(Z_t, Z_l) = \frac{1}{B} \sum_{b=1}^{B} \| Z_{t_b} - Z_{l_b} \|_2^2, \tag{3}$$

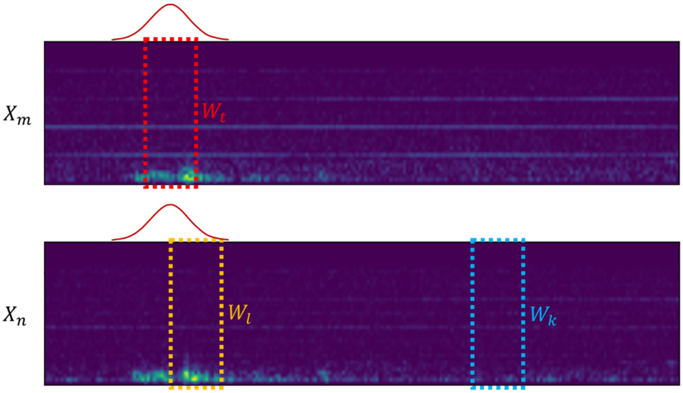


Fig. 10 Illustration of W_t, W_l, and W_k. Each denotes a reference window, a neighboring window, and a non-neighboring window. The red curve of a normal distribution shape represents the neighborhood

$$v(Z) = \frac{1}{F_z} \sum_{f=1}^{F_z} \mathrm{ReLU}\left(\gamma - \sqrt{\mathrm{Var}\left(Z_f\right) + \epsilon}\right), \tag{4}$$

$$c(Z) = \frac{1}{F_z^2} \sum_{i \neq j} C(Z)_{i,j}^2, \tag{5}$$

$$C(Z) = \left(\frac{Z - \bar{Z}}{\|Z - \bar{Z}\|_2}\right)^T \left(\frac{Z - \bar{Z}}{\|Z - \bar{Z}\|_2}\right) \text{ where } \bar{Z} = \frac{1}{B} \sum_{b=1}^{B} Z_b. \tag{6}$$

Here, $s(..)$ denotes the invariance term, $v(..)$ variance term and $c(..)$ the covariance term. Moreover, Var denotes a variance estimator, γ is a target value for the standard deviation, fixed to 1 as in the original implementation, ϵ is a small scalar (i.e. 0.0001) to prevent numerical instability, $\sum_{i \neq j}$ denotes summation of off-diagonal terms in a 2-dimensional matrix, and λ, μ, and ν are hyper-parameters to control the importance of each term. It should be noted that the notation for $L_{vibcreg}$ largely follows that of the original paper.

The second term on the right in Eq. (1) is defined as follows

$$L_{tnc} = \rho \Big(\{-\log \left(D(Y_t, Y_l)\right) + \log \left(1 - D(Y_t, Y_k)\right)\} + \{(1 - c_p(Y_t, Y_l))^2 + c_n(Y_t, Y_k)^2\} \Big), \tag{7}$$

$$c_p(Y_t, Y_l) = \frac{1}{F_y} \sum_{i=j} \left(\frac{Y_t - \bar{Y}_t}{\|Y_t - \bar{Y}_t\|}\right)^T \left(\frac{Y_l - \bar{Y}_l}{\|Y_l - \bar{Y}_l\|}\right) \text{ where } \bar{Y} = \frac{1}{B} \sum_{b=1}^{B} Y_b, \tag{8}$$

$$c_n(Y_t, Y_k) = \frac{1}{F_y} \sum_{i=j} \left(\frac{Y_t - \bar{Y}_t}{\|Y_t - \bar{Y}_t\|}\right)^T \left(\frac{Y_k - \bar{Y}_k}{\|Y_k - \bar{Y}_k\|}\right). \tag{9}$$

The original TNC loss function is defined as $-\log(D(Y_t, Y_l)) + \log(1 - D(Y_t, Y_k))$ given that its regularization weighting hyper-parameter is set to zero. In Eq. (7), there is an additional term, $(1 - c_p(Y_t, Y_l))^2 + c_n(Y_t, Y_k)^2$, which is proposed added. This addition improves quality of representation learning on top of the original TNC loss by correlating Y_t and Y_l and de-correlating Y_t and Y_k. Here, $\sum_{i=j}$ denotes summation of diagonal terms in a 2-dimensional matrix and ρ is a hyper-parameter for weighting L_{tnc}. In the experiments, λ, μ, ν, and ρ are empirically set to 10, 10, 10, and 13, respectively, based on suggested hyperparameters in (Lee and Aune 2021) for λ, μ, and ν and using a grid-search method for ρ.

Lastly, a quirky component in the VNIbCReg framework is the sensor invariance which is proposed for the following reason: A single sample consists of the 24 time series in our dataset. Although the 24 time series are different to some extent due to the sensors' location, functionality, and axial directions, they are semantically the same in the sense that they belong to the same class. The sensor invariance is what allows representations of the spectrograms of the different time series within the same sample to be pulled together in the embedding space by both VIbCReg and TNC.

After pre-training by an SSL method, only the encoder is typically kept while discarding the rest of the parts such as the projector and the discriminator. Then, the encoder can be used for various downstream tasks of classification with some fine-tuning.

6 Results

Experimental results relevant to the following items are presented in this section: (i) supervised learning with the aggregated spectrogram with respect to the model architecture (i.e. AlexNet vs. ResNet) and the class weight. The class weight is a method commonly used when a dataset is unbalanced for its classes. It gives higher weight to a class with a small number of samples while giving lower weight to a class with a large number of samples, (ii) supervised learning with respect to the ensemble prediction, (iii) Linear and fine-tuning evaluations on TNC, VIbCReg, VNIbCReg, and intermediate variants between VIbCReg and VNIbCReg. Before presenting the results, an experimental setup and the linear and fine-tuning evaluations are described.

6.1 Experimental Setup

Training and Test Datasets For naive supervised learning and linear evaluation, the dataset is split by stratified random sampling into 80% for training and 20% for testing. For the fine-tuning evaluation, the dataset is split into n% (where n is specified in the fine-tuning result table) and 20% of the dataset for training and test datasets, respectively. For the naive supervised learning, the linear evaluation, and the fine-tuning evaluation, samples with valid classes (i.e. noise, regional, rockfall, slope HF, slope LF, slope tremor, slope multi, and spike) are used as a dataset. For the SSL, a dataset consists of all the samples including the unlabeled-class samples. Given the randomness of the dataset split, all the experimental cases are run three times with different random seeds.

Preprocessing First, a time series input is converted into a spectrogram using a spectrogram function from `SciPy` with a 0.08 s-long sliding window with a 12.5% overlap (Virtanen et al. 2020). Then, it is scaled by log and min-max scaling, and resized such that the height is adjusted to 128 and the width is adjusted proportionally to the increase ratio of the height.

The preprocessing used in our experiments is somewhat different from that of (Langet and Silverberg 2022) in which spectrograms are computed using a 1 sec sliding window with a 95% overlap, and then converted into an RGB image with the size of $(3 \times 224 \times 224)$. The sliding window length and overlap give monotonous spectrograms i.e. low-resolution spectrogram, and it leads to an overfitting which give lower accuracy in our experiments. Also, the spectrograms do not have to be converted into square-shaped RGB images. Spectrograms originally have dimension of $(1 \times H \times W)$ as 1 channel real-valued matrix, where H and W denote height and width, respectively. Thus, a naive form of spectrograms can be already viewed as an 1-dimensional real-valued image data. In addition, because a convolutional layer can process rectangle image shapes as well as square shapes, the rectangle shapes of the naive form of spectrograms can be processed by a CNN model without squishing the rectangular shape into a square, which can actually lead to some information loss and performance drop.

The aggregation of the spectrograms of the 24 time series is only used for cases where the ensemble prediction is not used. While Langet and Silverberg (2022) aggregate the spectrograms over those that are not so noisy based on a user-defined threshold, Ovanger (2021) uses a weighted aggregation based on the inverse of the signal-to-noise ratio such that noisy time series gets a smaller weight. As the intuitive weighted aggregation does not require any user-specific threshold it is used in this study.

Augmentation In our experiments, there are two types of augmentations: Neighboring Crop (NC) and Random Crop (RC). The NC takes a spectrogram and outputs three different cropped windows of the spectrogram, W_t, W_l, and W_k, as shown in Fig. 9. In contrast, the RC outputs three different cropped windows of the spectrogram. But the cropped windows in the RC do not have any spatial relation between each other as the three cropped windows are selected at random. In our experiments, height of the crop is set to the same height as the input spectrogram and the width of the crop is set to 10% of width of the input spectrogram for both the NC and the RC. Although additional augmentation methods can likely gain extra performance improvement, no other method is used here to clearly compare the performance between the RC and the NC in the linear and fine-tuning evaluations.

Architecture In our experiments, AlexNet and ResNet34 are compared. AlexNet is the standard CNN model used in the relevant previous studies (Langet and Silverberg 2022) and ResNet is a very promising CNN model. ResNet34 is specifically chosen due to its representation size as 512 which is the same as in VIbCReg and has bigger model capability than ResNet18. It should be noted that the numbers of parameters for AlexNet and ResNet34 are around 57 million and 21 million, respectively. Hence, ResNet34 is almost three times lighter than AlexNet. The original implementation of AlexNet and ResNet takes RGB images (i.e. $C \times H \times W$, where C is 3) as input.

Therefore, the input channel size is 3 for the first convolutional layers in both models. In our experiments, the input spectrogram has a dimension of $(1 \times H \times W)$, therefore, the first convolutional layers in both models are modified to receive an input with one channel. Except that, both model architectures remain the same as in (Langet and Silverberg 2022). The same holds for the projector and the discriminator.

Optimizer Adam (Kingma and Ba 2014) is used with a cosine learning rate scheduler. Its initial learning rate for the encoder and the classifier is set to 0.001 for the SSL, naive supervised learning, and the linear evaluation. For the fine-tuning evaluation on the SSL methods, the initial learning rates are set to 0.0005 and 0.001 for the encoder and the classifier, respectively. The batch size is 128 for the SSL and 64 for naive supervised learning, the linear evaluation, and the fine-tuning evaluation. A number of epochs is 300 for the self-supervised learning and 100 for naive supervised learning, the linear evaluation, and the fine-tuning evaluation. The used deep learning library is `PyTorch` (Paszke et al. 2019).

Criteria Two criteria are used: accuracy and confusion matrix. The accuracy is defined as Eq. (10) where TP, FP, TN, and FN denote True-Positive, False-Positive, True-Negative, and False-Negative, respectively. The confusion matrix shows how many predicted labels are correctly predicted per each class in a matrix form.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{10}$$

6.2 Linear and Fine-tuning Evaluations

The linear and fine-tuning evaluations are conventional evaluations methods for SSL (Chen et al. 2020; Grill et al. 2020; Zbontar et al. 2021; Bardes et al. 2021; Lee and Aune 2021). The linear evaluation protocol is as follows: After the model training by a SSL method, only the encoder is kept and its weights are frozen (i.e. set to be non-trainable). Next, the frozen encoder is topped with a linear layer which is fitted on a training dataset. A goal of the linear evaluation is to see how linearly separable the learned representations are for classification. If good classification performance can be achieved by the linear layer on the learned representations, the quality of the learned representations is good.

The fine-tuning evaluation protocol is as follows: After the model training by a SSL method, similarly to the linear evaluation, only the encoder is kept. But the encoder remains trainable and it is topped with a linear layer and fitted on a subset of a training dataset (e.g. 5% of a training dataset). This is to see how well the model is generalized by a SSL method such that the model would perform decently even with a small dataset. The fine-tuning evaluation is particularly good at revealing the effectiveness of a SSL method in a situation with a small amount of labeled data and a large amount of unlabeled data. In the fine-tuning evaluation with a small subset of a training dataset (i.e. 5% and 10%), the averaged Batch Normalization (BN)'s statistics obtained during training are used during fine-tuning to utilize BN statistics obtained from a relatively

Table 2 Test accuracy of supervised learning with the aggregated spectrograms with respect to the class weight and the model architecture

Class weight	AlexNet	ResNet34
Used	0.867 (0.005)	**0.878 (0.012)**
Not used	0.864 (0.01)	**0.881 (0.012)**

Mean accuracy along with standard deviation is reported, where the standard deviation is reported in the parenthesis. The bold font denotes the higher accuracy

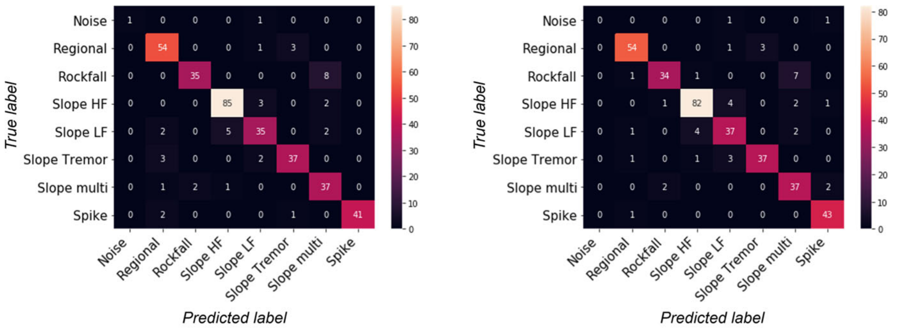
larger dataset. This is a common practice in fine-tuning with a small dataset, which can easily be achieved by specifying `model.eval()` in a training step in PyTorch.

6.3 Results

Supervised Learning with the Aggregated Spectrogram with respect to Architecture and Class Weight First, comparison between AlexNet and ResNet is made while using the aggregated spectrogram as input. Its experimental result is shown in Table 2. The class weight is often used when a dataset is unbalanced to balance the classification accuracy between different classes. The experimental results with respect to the use of the class weight are also shown in Table 2. It should be noted that the class weight is calculated among classes except for the noise class. Samples with the noise class do not have common patterns (i.e. they are just a collection of samples with uncertain classes without common/shared patterns). Hence, it is difficult to train a model to do a decent classification on the noise class anyways. Also, the lack of samples for the noise-class (i.e. 8 samples only), gives another reason for excluding the noise-class in the class weight. The optimizer settings mentioned in Sect. 6.1 are a bit different in this experiment only. The number of epochs is 50 instead of 100 and the weight decay of 0.0001 is used to prevent severe overfitting.

Table 2 shows that ResNet outperforms AlexNet in terms of accuracy, while it remains almost the same with respect to the use of the class weight. The confusion matrix in Fig. 11 shows the accuracy difference between classes, and that this is influenced by the class weight. Although it does not pose a significant change, still it shows that the correct classification is slightly more distributed over different classes when the class weight is used. The class weight is employed in all the following experiments.

Supervised Learning with respect to Ensemble Prediction The experimental results with respect to the ensemble prediction and the model architecture are shown in Table 3. The noticeable test accuracy improvement compared with Table 2 is seen for both AlexNet and ResNet34. ResNet34 with the ensemble prediction performs the best in terms of accuracy. The confusion matrix of ResNet34 with respect to the ensemble prediction is presented in Fig. 12. To further investigate the ensemble prediction, examples of individual predictions within the ensemble prediction on some samples are presented in Fig. 13. With the ensemble prediction, each spectrogram of the 24 time



(a) Without the class weight

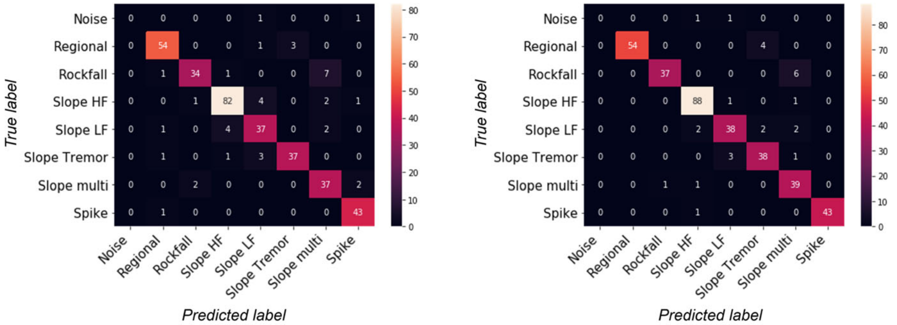
(b) With the class weight

Fig. 11 Confusion matrix of ResNet34 with the aggregated spectrogram with respect to the class weight. The color intensity denotes a number of classified samples (the brighter, the higher)

Table 3 Test accuracy of supervised learning with respect to the use of the ensemble prediction and the model architecture

Ensemble prediction	AlexNet	ResNet34
Used	<u>0.916 (0.005)</u>	**0.928 (0.005)**
Not used	0.867 (0.005)	0.878 (0.012)

Note that if the ensemble prediction is not used, that is equivalent to the use of the aggregated spectrograms as input. The bold font and the underline denote the highest accuracy and the second highest accuracy, respectively



(a) Without the ensemble prediction

(b) With the ensemble prediction

Fig. 12 Confusion matrix of ResNet34 with respect to the ensemble prediction

series can maximally be utilized without information loss due to some aggregation process and better classification can be achieved with the ensemble approach over the 24 predictions.

Linear and Fine-tuning Evaluations on SSL Methods The linear and fine-tuning evaluations are conducted on VIbCReg, VNIbCReg, and intermediate variants between VIbCReg and VNIbCReg. The evaluation on the intermediate variants is conducted

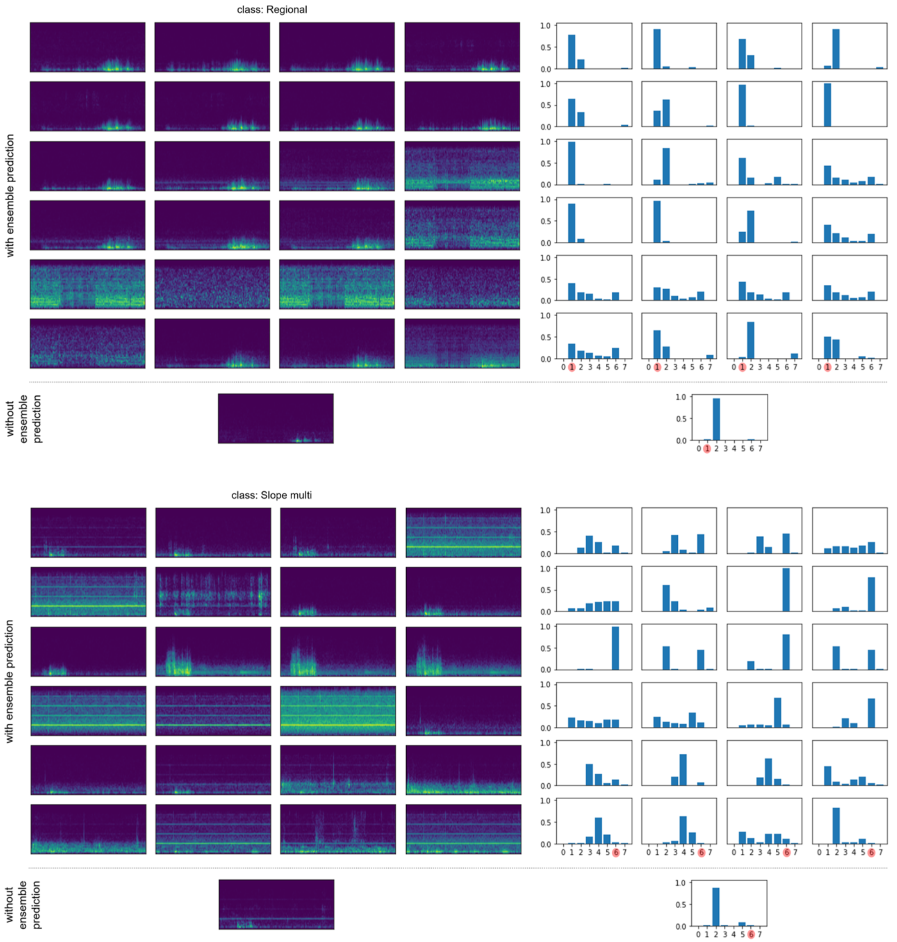


Fig. 13 Examples of the ensemble prediction with two different classes (top: Regional, bottom: Slope multi) where a correct label is predicted with the ensemble prediction while an incorrect label is predicted without the ensemble prediction. In each sub-figure, there are 6 × 4 displays of the 24 time series. The left displays show each of the 24 spectrograms. The right display shows the softmax classification prediction on each of these. The single-display below the tables shows the aggregated spectrogram and softmax classifications. The true label is colored with a small red circle. Note that the final prediction is the averaged prediction of the 24 predictions for the ensemble prediction. The class indices refer to the following: 0: noise, 1: regional, 2: rockfall, 3: slope HF, 4: slope LF, 5: slope tremor, 6: slope multi, 7: spike

to find out how much contribution each component that is added on VIbCReg to form VNIbCReg makes.

The linear evaluation results are presented in Table 4. VIbCReg does not perform so well here, meaning that the learned representations by VIbCReg are not so linearly-separable by class. By introducing main components from the original TNC (i.e. NC and $L^{\dagger}_{tnc}$), a significant improvement is achieved in the linear evaluation. This indicates that the ability to encode the temporal transition improves quality of learned representations, especially for non-stationary time series with apparent temporal transition.

Table 4 Linear evaluation result. RC and NC refer to the random crop and neighboring crop, respectively. $L_{tnc}^{\dagger}$ denotes the original TNC loss and L_{tnc} denotes the modified TNC loss proposed in this work.

Base SSL method	Augmentation for cropping	$L_{tnc}^{\dagger}$	L_{tnc}	Test acc.	Remarks
RandInit	x	x	x	0.642 (0.026)	
TNC	NC	o	x	0.499 (0.011)	= TNC
VIbCReg	RC	x	x	0.476 (0.02)	= VIbCReg
VIbCReg	NC	x	x	0.578 (0.005)	
VIbCReg	NC	o	x	0.698 (0.023)	
VIbCReg	NC	x	o	**0.716 (0.025)**	**= VNIbCReg**

o and x denote used and not-used. Note that VNIbCReg corresponds to the based SSL method of VIbCReg with the NC and L_{tnc}. RandInit refers to a case where an encoder is randomly-initialized and frozen without any pre-training and used for the linear evaluation

Table 5 Fine-tuning evaluation result. The notation is the same as in the linear evaluation result

Base SSL method	Augmentation for cropping	$L_{tnc}^{\dagger}$	L_{tnc}	Test acc. (fine-tuned on $n\%$ of the dataset)		
				$n = 5(\%)$	$n = 10(\%)$	$n = 80(\%)$
RandInit	x	x	x	0.185 (0.036)	0.528 (0.038)	**0.928 (0.005)**
TNC	NC	o	x	0.5 (0.029)	0.715 (0.033)	0.915 (0.01)
VIbCReg	RC	x	x	0.504 (0.036)	0.628 (0.024)	0.912 (0.002)
VIbCReg	NC	x	x	0.619 (0.036)	0.747 (0.016)	0.912 (0.003)
VIbCReg	NC	o	x	0.642 (0.019)	0.799 (0.029)	0.92 (0.002)
VIbCReg	NC	x	o	**0.717 (0.037)**	**0.824 (0.03)**	0.919 (0.005)

RandInit refers to a case where an encoder is randomly-initialized and frozen without any pre-training and used for the fine-tuning evaluation (i.e. naive supervised learning). Note that the accuracy is much higher with a small labeled dataset when pretrained by VNIbCReg

The effectiveness of L_{tnc} is shown by further improving the performance. The fine-tuning evaluation result is presented in Table 5. The performance ranking order of the fine-tuning evaluation between VIbCReg, VNIbCReg, and the intermediate variants remains the same as in the linear evaluation. One of the noticeable points in the result is the high test accuracy in a small dataset regime (i.e. n of 5% and 10%) for models that are pretrained by an SSL method. Especially, VNIbCReg results in the highest test accuracy in the small dataset regime, indicating that the model is well generalized by VNIbCReg. As the geophone sensor data is collected unlabeled, there is naturally a large unlabeled dataset while the labeled dataset is much smaller because proper labeling can only be done manually by an expert. Given that circumstance, VNIbCReg is expected to provide robust performance improvement when a large amount of an unlabeled dataset is utilized.

7 Conclusion

In this paper new machine-learning approaches were implemented and tested for classifying seismic geophones events at an unstable rock slope in Norway. Improved algorithms for classification of microseismic events are important to develop better decision support tools for this case.

A version of ensemble learning in which the 24 time series from all geophone components were treated separately and then combined via ensemble prediction showed to get very high performance. The lesson learned from this is that early aggregation to improve the signal-to-noise ratio is not necessarily beneficial. Instead, the ensemble learning approach, which consists in computing one spectrogram for each time series recorded by each component of each geophone, appeared more useful here, giving substantial accuracy gains in the classification task in the end. Self-supervised learning methods also gave good performance for our case, particularly so when labeled training data lacks.

The classification methods outlined here, do not only apply to geophone data related to rock hazards applications. The same statistical machine learning techniques can also be applied to microseismic events in reservoirs or with other data such as distributed acoustic sensing data which can potentially complement seismic data (Binder and Tura 2020). This can be particularly important during CO2 injection projects in the future, where seismic events can indicate potential leakage but can also be a result of other kinds of activity.

Acknowledgements We acknowledge support from the Norwegian Research Council project ML4ITS Grant 312062 and the SFI Centre for Geophysical Forecasting Grant 309960.

Funding Information Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital)

Data availability The source code is available at https://github.com/ML4ITS/Aknes_clf with the Åknes dataset included. The dataset is registered in (Lee et al. 2022). For the Åknes dataset, events were classified by visual inspection of a large number of waveforms recorded in different years, and proper labeling was ensured by cross-checking with a reviewed seismic bulletin (NORSAR 1971). Therefore, this dataset is encouraged to be used as a benchmark dataset so that different classification methods can be developed and fairly compared to each other.

Declarations

Conflict of interest Not applicable.

References

Bardes A, Ponce J, LeCun Y (2021) Vicreg: Variance-invariance-covariance regularization for self-supervised learning. arXiv preprint arXiv:2105.04906

Bardi F, Raspini F, Ciampalini A, Kristensen L, Rouyet L, Lauknes TR, Frauenfelder R, Casagli N (2016) Space-borne and ground-based InSAR data integration: the Åknes test site. Remote Sens 8(3):237

Benítez MC, Ramírez J, Segura JC, Ibanez JM, Almendros J, García-Yeguas A, Cortes G (2006) Continuous hmm-based seismic-event classification at deception island, antarctica. IEEE Trans Geosci Remote Sens 45(1):138–146

Bernardi MS, Africa PC, De Falco C, Formaggia L, Menafoglio A, Vantini S (2021) On the use of interferometric synthetic aperture radar data for monitoring and forecasting natural hazards. Math Geosci 53(8):1781–1812

Binder G, Tura A (2020) Convolutional neural networks for automated microseismic detection in downhole distributed acoustic sensing data and comparison to a surface geophone array. Geophys Prospect 68(9):2770–2782

Chen T, Kornblith S, Norouzi M, Hinton G (2020) A simple framework for contrastive learning of visual representations. In: International conference on machine learning, PMLR, pp 1597–1607

Chen X, He K (2021) Exploring simple siamese representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 15750–15758

Cohen L (1989) Time-frequency distributions-a review. Proc IEEE 77(7):941–981

Curilem G, Vergara J, Fuentealba G, Acuña G, Chacón M (2009) Classification of seismic signals at Villarrica volcano (Chile) using neural networks and genetic algorithms. J Volcanol Geoth Res 180:1–8

Dammeier F, Moore J, Hammer C, Haslinger F, Lowe S (2016) Automatic detection of alpine rockslides in continuous seismic data using hidden markov models. J Geophys Res Earth Surf 121:351–371

Falsaperla S, Graziani S, Nunnari G, Spampinato (1996) Automatic classification of volcanic earthquakes by using multi-layered neural networks. Natural Hazards 13:205–228

Feng L, Pazzi V, Intrieri E, Gracchi T, Gigli G (2020) Joint detection and classification of rockfalls in a microseismic monitoring network. Geophys J Int 222:2108–2120

Gharti HN, Komatitsch D, Oye V, Martin R, Tromp J (2012) Application of an elastoplastic spectral-element method to 3D slope stability analysis. Int J Numer Meth Eng 91(1):1–26

Grill JB, Strub F, Altché F, Tallec C, Richemond P, Buchatskaya E, Doersch C, Avila Pires B, Guo Z, Gheshlaghi Azar M, Piot B, Kavukcuoglu K, Munos R, Valko M (2020) Bootstrap your own latent-a new approach to self-supervised learning. Adv Neural Inf Process Syst 33:21271–21284

Grøneng G, Christiansen HH, Nilsen B, Blikra LH (2011) Meteorological effects on seasonal displacements of the åknes rockslide, western Norway. Landslides 8(1):1–15

Hammer C, Ohrnberger M, Fäh D (2013) Classifying seismic waveforms from scratch: a case study in the alpine environment. Geophys J Int 192:425–439

Harbitz C, Glimsdal S, Løvholt F, Kveldsvik V, Pedersen G, Jensen A (2014) Rockslide tsunamis in complex fjords: From an unstable rock slope at åkerneset to tsunami risk in western Norway. Coast Eng 88:101–122

Hastie T, Tibshirani R, Friedman JH, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction, vol 2. Springer, Berlin

He K, Fan H, Wu Y, Xie S, Girshick R (2020) Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9729–9738

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778

Hibert C, Provost F, Malet JP, Maggi A, Stumpf A, Ferrazzini V (2017) Automatic identification of rockfalls and volcano-tectonic earthquakes at the Piton de la Fournaise volcano using a Random Forest algorithm. J Volcanol Geoth Res 340:130–142

Huang L, Zhou Y, Zhu F, Liu L, Shao L (2019) Iterative normalization: Beyond standardization towards efficient whitening. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 4874–4883

Ibáñez J, Benítez C, Gutiérrez L, Cortés G, García-Yeguas A, Alguacil G (2009) The classification of seismo-volcanic signals using Hidden Markov Models as applied to the Stromboli and Etna volcanoes. J Volcanol Geoth Res 187:218–226

Ibs-von Seht M (2008) Detection and identification of seismic signals recorded at Krakatau volcano (Indonesia) using artificial neural networks. J Volcanol Geoth Res 176:448–456

Kingma D P, Ba J (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980

Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60(6):84–90

Langer H, Falsaperla S, Masotti M, Campanini R, Spampinato S, Messina A (2009) Synopsis of supervised and unsupervised pattern classification techniques applied to volcanic tremor data at Mt Etna, Italy. Geophys J Int 178:1132–1144

Langer H, Falsaperla S, Powell T, Thompson G (2006) Automatic classification and a-posteriori analysis of seismic event identification at Soufrière Hills volcano, Montserrat. J Volcanol Geoth Res 153:1–10

Langer H, Falsaperla S, Thompson G (2003) Application of artificial neural networks for the classification of the seismic transients at Soufrière Hills volcano, Montserrat. Geophys Res Lett 30(21)

Langet N, Silverberg F (2022) Automated classification of seismic signals recorded on the Åknes rockslope, Western Norway, using a convolutional neural network. Earth Surf Dyn

LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324

Lee D, Aune E (2021) VIbCReg: Variance-invariance-better-covariance regularization for self-supervised learning on time series. arXiv preprint arXiv:2109.00783

Lee D, Aune E, Langet N, Eidsvik J (2022) Seismic signal dataset from the Åknes Rockslope. https://doi.org/10.6084/m9.figshare.21340101.v1

Liu Z, Mao H, Wu C Y, Feichtenhofer C, Darrell T, Xie S (2022) A convnet for the 2020s. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11976–11986

Maggi A, Ferrazzini V, Hibert C, Beauducel F, Boissier P, Amemoutou A (2017) Implementation of a multistation approach for automated event classification at piton de la fournaise volcano. Seismol Res Lett 88:878–891

Malfante M, Dalla Mura M, Mars J, Métaxian JP, Macedo O, Inza A (2018) Automatic classification of volcano seismic signatures. J Geophys Res Solid Earth 123:10,645-10,658

Masotti M, Falsaperla S, Langer H, Spampinato S, Campanini R (2006) Application of support vector machine to the classification of volcanic tremor at Etna, Italy. Geophy Res Lett 33

Nordvik T, Nyrnes E (2009) Statistical analysis of surface displacements-an example from the åknes rockslide, western Norway. Nat Hazard 9(3):713–724

NORSAR (1971) NORSAR: NORSAR Seismic Bulletins. https://www.norsar.no/seismic-bulletins/, https://doi.org/10.21348/b.0001

Ovanger O (2021) Graph gaussian process classifier with anchor graph and label propagation. Master's thesis, NTNU

Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S (2019) PyTorch: An imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A, d' Alché-Buc F, Fox E, Garnett R, (eds), Advances in neural information processing systems 32, Curran Associates, Inc., 8024–8035

Provost F, Hibert C, Malet JP (2017) Automatic classification of endogenous landslide seismicity using the Random Forest supervised classifier. Geophys Res Lett 44:113–120

Roth M, Blikra L (2009) Seismic monitoring of the unstable rock slope at Aaknes, Norway. Geophys Res Abstr 11. https://doi.org/10.4133/1.2923645

Roth M, Dietrich M, Blikra L H, Lecomte I (2006) Seismic monitoring of the unstable rock slope site at åknes, norway. In: Symposium on the application of geophysics to engineering and environmental problems 2006, Society of Exploration Geophysicists, pp 184–192

Scarpetta S, Giudicepietro F, Ezin E, Petrosino S, Del Pezzo S, Martini M, Marinaro M (2005) Automatic classification of seismic signals at Mt. Vesuvius Volcano, Italy, using neural networks. Bull Seismol Soc Am 95:185–196

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 1–9

Tonekaboni S, Eytan D, Goldenberg A (2021) Unsupervised representation learning for time series with temporal neighborhood coding. arXiv preprint arXiv:2106.00750

Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat İ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P, SciPy 10 Contributors, (2020) SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods 17:261–272

Vouillamoz N, Rothmund S, Joswig M (2018) Characterizing the complexity of microseismic signals at slow-moving clay-rich debris slides: the Super-Sauze (southeastern France) and Pechgraben (Upper Austria) case studies. Earth Suface Dyn 6:525–550

Zbontar J, Jing L, Misra I, LeCun Y, Deny S (2021) Barlow twins: Self-supervised learning via redundancy reduction. In: International conference on machine learning, PMLR, pp 12310–12320