Data augmentation techniques applied to the classification of multi-station seismic-volcanic signals

Franz Yupanqui Machaca[a,b,c], John Doe[a], and Jane Doe[c]

[a]Programa de Doctorado en Ingeniería Sustentable, Universidad Católica del Norte, Avenida Angamos 0610, 1270709 Antofagasta, Chile
[b]Millennium Institute on Volcanic Risk Research – Ckelar Volcanoes, Avenida Angamos 0610, 1270709 Antofagasta, Chile
[c]4D Perceptio, Bolivia
[d]Department of Another Thing, University of State and City, City, ST

***Abstract*—The abstract goes here.**

***Index Terms*—Deep Learning, Transfer Learning, Data Augmentation, seismic.**

I. Introduction

A. The problem of recognition

The seismic-volcanic signatures are dangerous events that happen unexpectedly, although these present some signals before they happen. In the world, these events are registered continuously by means of seismometers. Afterward, expert geophysicists take these data and classify these signals, task The seismic-volcanic activity has been studied for many years. Nowadays, this type of activity is registered by sensors that record signals in three dimensions. The seismic-volcanic pattern recognition process in Chile is typically done manually, although there are state-of-the-art studies that it makes automatically form **salazar2022multi**, **salazar2020deep**, **malfante2018automatic**, **bicego2012classification**, **curilem2016pattern**, **lara2020automatic**. These studies regularly consider one channel of three available, one station, and one volcano. In this context, applying Machine Learning (ML) and Deep Learning (DL) techniques to the automatic classification of volcanic seismic events has become an interesting approach compared to traditional methods.

B. A summary of methods used in classificaction of volcanic seismic events

In the analysis and treatment of seismic-volcanic time signals, transformations are regularly considered that allow us to represent the observations in the domains of: time, frequency and cepstral**malfante2018automatic**, **lara2020automatic**, which consider statistical measures, information theory, among others. Among the methods used for the analysis of this type of signals we have the Hidden Markov model (HMM)**bicego2012classification**, **beyreuther2012constructing**, **carniel2014characterization** with its improvements such as the Hidden Semi-Markov Models (HSMM), the Continuous Wavelet Transform (CWT)**beyreuther2012constructing** in the detection of singularities, Gaussian Mixture Model (GMM)**langer2009synopsis**, and others (HMM-based generative embedding)**bicego2012classification**. Currently, there are several works that propose the automatic classification of seismic-volcanic signals that consider supervised Machine Learning (ML) methods such as Support Vector Machines (SVM)**malfante2018automatic**, **curilem2016pattern**, **giacco2009support**, **langer2009synopsis**, **lara2020automatic**, **curilem2018improving**, **malfante2018machine**; Multilayer Perceptron(MLP)**giacco2009support**, **langer2009synopsis**, **lara2020automatic**; Random Forest**malfante2018automatic**, **lara2020automatic**, **titos2019classification**; Linear Discriminant Analysis**lara2020automatic**; Convolutional Neural Networks (CNN)**titos2019classification**, **salazar2020deep**, **curilem2018using**; Temporal Convolutional Neural Network(TCN)**titos2019classification**, **rodriguez2021bayesian**; Bayesian Neural Networks**bueno2019volcano**; Tensor Learning Framework with Multilinear Principal Component Analysis(MPCA) and Support Tensor Machine(STM)**peixoto2021tensor** techniques on multichannel and multistation data; Scalable Variational Inference for Gaussian Process (SVGP) and Deep Gaussian Processes (DGP)**lopez2020acontribution**; Bag of Words in seismic document classification as a set of seismic words**bicego2015volcano**. With unsupervised methods we have Kohonen Self-Organizing Map (SOM)**langer2009synopsis**, **esposito2008unsupervised**, Cluster Analysis(CA)**langer2009synopsis**, Balanced iterative reducing and clustering using hierarchies(BIRCH)**duque2020exploring**. Transfer Learning (TL) was applied in the classification of volcanic seismic events using CNN**mythesismaster**, **titos2019classification**. Deep Learning (DL) has been applied to seismic-volcanic signal classification, experimenting with different CNN**mythesismaster** and RNN architectures**salazar2020deep**, **ferreira2023deep**, **canario2020indepth**, and through Deep Gaussian Processes (DGP)**lopez2020acontribution**. Recurrent Neural Networks

F. Yupanqui Machaca, frayup@gmail.com
J. Doe frayup@gmail.com
J. Doe frayup@gmail.com
Manuscript received April 19, 2005; revised September 17, 2025.

(RNNs) provide real-time tracking capabilities of volcanic activity, even if the seismic sources change over time. It has been applied to seismic-volcanic signal classification, testing several architectures**salazar2020deep**. In detection and classification, the following RNN architectures have been applied: vanilla-RNN, Long Short-Term Memory (LSTM), and Gated-Recurrent Unit (GRU)**titos2018detection**; in **rodriguez2021bayesian** vanilla-RNN and LSTM, in addition to identifying the degree of specialization of the neurons. Autoencoder (AE) is an excellent unsupervised learning algorithm. AE is considered a nonlinear compression structure to reduce the signal size, dimensionality reduction, towards another latent dimension. In **mythesismaster** various AE architectures generated reconstructed and encoded seismic events, which were classified using different DL and TL model architectures. In **rodriguez2021bayesian** AE is applied to obtain the segmentation mask of the seismic event. Data Augmentation (DA), which basically deals with the generation of new data from existing data, following some rules and restrictions so that they are considered equally valid as the originals. Amplitude deformation, frequency deformation, noise addition, and horizontal shift were applied to seismic signals **curilem2018using**. In **salazar2022multi** the importance of DA to avoid biased models in the classification of seismic-volcanic events was shown. In reference to volcanic seismic data, most works consider a single station channel for the classification of volcanic seismic events, but there are already some works that consider several channels and stations **lara2020automatic**, **curilem2016pattern**, **maggi2017implementation**, **ferreira2023deep**. In addition, it is shown that the multi-station scheme is more precise than the single station **titos2019classification**, **ferreira2023deep**.

II. Methodology

Currently, Machine Learning is already used for the Recognition of seismic-volcanic signal patterns. In fact, our previous researches developed**mythesismaster**, **salazar2022multi** models that have implemented a new methodology for classifying these types of seismic signals by Deep Learning, Transfer Learning, and spectrograms. The rotation technique was used as a data augmentation technique for avoiding imbalanced data. The current research on recognizing seismic-volcanic signal patterns will consider data from the three Chilean volcanos. The Millennium Institute on Volcanic Risk Research – Ckelar Volcanoes[1] monitors many of these volcanos by seismic stations provided with triaxial sensors.

A. Lascar volcano

The Lascar volcano is located in the north of Chile (23°22' S, 67 °44' W; 5592 m a.s.l.), 270 km NE from Antofagasta and 70 km SE from San Pedro de Atacama. The Millennium Institute for Volcanic Risk Research (CKELAR) has been monitoring the area using a network of 11 three-component seismic stations. Short-period 2 Hz seismometers have been continuously monitoring at 200 Hz. The database of signals was built using only the Z channel.

[1]https://ckelar.org

B. Lascar's database

The Lascar database contains 6145 seismic events classified for geophysicist experts belonging to Ckelar. The data corresponds period between Mach to October 2018. In the next table, we show the data distribution.

III. Methods

We proposed a solution based on transfer learning because this helps to speed up training and improve the performance of the deep learning models **titos2019classification**, **bueno2019volcano**, in addition to having few examples to generate the model again. For the TL, the pre -trained AlexNet**krizhevsky2012imagenet** model was considered. The input data are generated spectrograms of the data labeled (waveforms) of each seismic event class. The processing sequence consists of five steps:

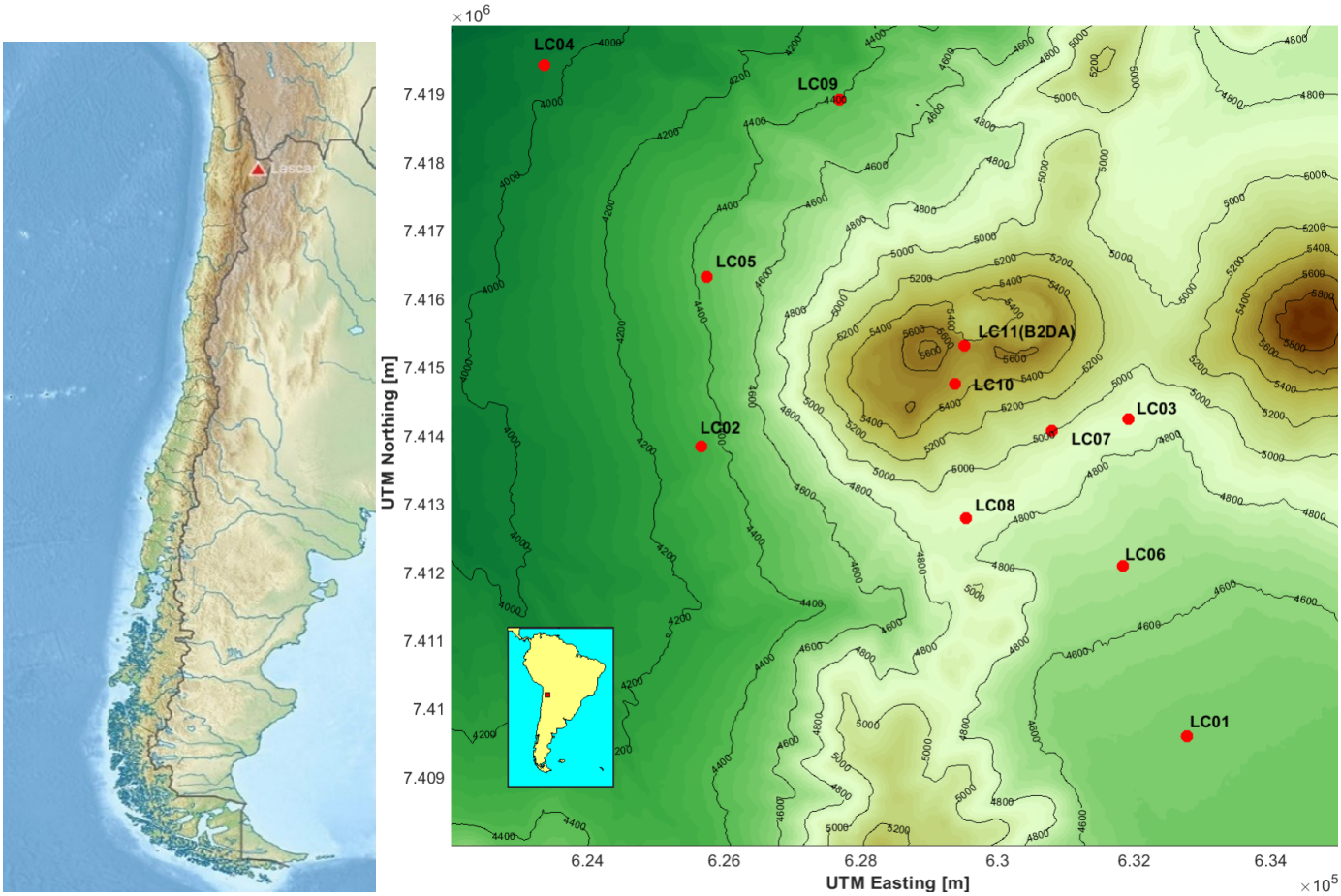
1) describe the pre-process that applies to the time series (earthquake);
2) detail the generation of spectrograms from the earthquakes labeled;
3) use the data augmentation technique to increase the amount of data in training and test data sets;
4) the construction of the prediction model; and, finally,
5) estimate the performance of the model.

The next paragraphs, each step is explained.

1) Pre-processing: The seismometers register data at 200Hz, 500Hz or 100Hz. We resampled at 100Hz each event. Centered the data at mean. Apply the Butterworth-Highpass filter at 1Hz. Then we apply Butterworth-Bandpass filter at frequency between 1Hz to 10Hz. Next step is normalizing the data. This preprocessing task is realized by Obspy python toolbox **beyreuther2010obspy**.
2) The spectrograms were calculated applying a short-time Fourier transform with formula
$$S[x(t)](n,k) = \left| \sum_{m=0}^{N-1} x(m) \cdot w(m-n) \cdot e^{i2\pi mk} \right|^2$$
Where x(t) and y(t) represent the seismic signal and short-time Fourier transform sliding window. The sliding window size was set to 1 with a 95
3) Data augmentation techniques are considered to avoid unbalanced dataset, and for that reason to avoid biased models. Many data augmentation techniques were implemented, and these will be presented in the next section.
4) The pre-trained model AlexNet**krizhevsky2012imagenet** was considered for the being retrained over spectrograms, using the scheme of transfer learning.
5) The model performance is evaluated for metrics described in the next sections. The metrics is implemented on the TorchMetrics python toolbox[2].

Codes for reproducing our method are available at.[3]

[2]https://lightning.ai/docs/torchmetrics/stable
[3]https://github.com/franznet/daseismicsignals



(a) (b)

Fig. 1: a) Location map of the Láscar volcano. b) Red dots indicate the position of seismic stations.

TABLE I: Contribution of events from each station to the Lascar database.

Event\Station	LC01	LC02	LC03	LC04	LC05	LC06	LC07	LC08	LC09	LC10	B2DA
HY	0	0	0	7	45	4	0	3	12	136	6
LP	11	0	4	2	142	28	5	60	23	195	107
TC	9	0	11	97	284	36	157	127	144	1,171	162
TR	1	0	5	29	21	4	61	13	35	292	10
VT	2	0	18	52	84	6	206	14	41	2,249	14
TOTAL	23	0	38	187	576	78	429	217	255	4,043	299

TABLE II: Statistics of the events belonging to the Láscar volcano.

Events	Quantity (%)	Quantity	Maximum (s)	Minimum (s)	Mean (s)	Median (s)	Mode (s)	Standard Deviation (s)
HY	3%	213	265	10	57.2	59	40	35.3
LP	9%	577	600	9	82.3	67	25	63.5
TC	36%	2,198	670	9	78.5	65	20	56.2
TR	8%	471	216	9	46.2	31	20	33.8
VT	44%	2,686	189	5	19.6	17	10	12.3
TOTAL	100%	6,145	670	5	49.9	28.0	20	50.0

A. Data Similarity

Kullback–Leibler divergence (KL divergence) has been used to measure similarities between synthetic and real datasets, and is defined as

$$D_{KL}(P||Q) = \sum_i P(x_i) \cdot log\left(\frac{P(x_i)}{Q(x_i)}\right)$$

where P and Q are the probability distributions whose distance is calculated over the samples x_i of the distribution**iglesias2023data**. Kullback–Leibler divergence (KL divergence) is not a symmetric distance, so it can be symmetrized to give rise to the so-called Jensen–Shannon divergence (JSD), defined as

$$JSD(P||Q) = D_{KL}(P||(P+Q)/2) + D_{KL}(Q||(P+Q)/2)$$

A measurement to quantify the distance between the time series distribution. It is based on calculating the Wasserstein distance between time series data. The metric is defined by measuring the Wasserstein distance of the energy between frequencies. The probability distributions have a Wasserstein–Fourier distance which is calculated as follows**iglesias2023data**:

$$WF([x],[y]) = W_2(s_x, s_y)$$

where s_x and s_y are the normalised power spectral densities of the distributions.

B. Data Augmentation

1) Rotation: This transformation considers the rotation of the spectrogram around the time axis, within a given percentage range of signal length and at a given axis position. The axis position can be at the beginning, middle, or end of the spectrogram. When applying this technique, the combination of the percentage rotation value, the location of the rotation axis, and the spectrogram of the event to which it will be applied is obtained. When generating new instances, care is taken to ensure that these three values are not repeated, thus ensuring unique instances in the dataset.

2) Jittering: Jittering is a technique for data augmentation that is utilized in time series analysis**iwana2021empirical**, **iglesias2023data**. One of the simplest yet most effective transformation-based methods is how it is described. Jittering involves adding noise to a time series. Mathematically, an original time series $x = x_1, ..., x_t, ..., x_T$ is transformed into $x' = x_1 + \epsilon_1, ..., x_t + \epsilon_t, ..., x_T + \epsilon_T$ by adding noise at each time step t, where ϵ is typically Gaussian noise, distributed according to $N(\theta, \sigma^2)$. The standard deviation σ of the added noise is a hyperparameter that must be predetermined**iwana2021empirical**. Jittering assumes that the time series patterns of a particular dataset are typically noisy. Sensor, audio, or electroencephalogram (EEG) data could all be included in this statement. Jittering was used with wearable sensor data for Parkinson's disease monitoring.

3) Drifting: One issue with both online and offline settings is the drift, which is the change in the data distribution over time. Random walk is used to simulate data drift. In mathematics, a random walk is classified as a stochastic or random process that depicts a path that comprises a succession of random steps in a mathematical space, such as integers or real. The integer number line Z is a basic example of a random walk, which begins at θ, moves 1, or moves -1 with equal probability at each step. A random move of $+a$ or $-a$ with equal probability can be observed on any vector space, such as the real space. The real number $a \epsilon \mathrm{R}$ is the magnitude of the random walk[52]. Let's say d defines a path starting at position $d_1 = \phi(\tau)$. A random walk is modeled by the following expression:

$$d_t = d_{t-1} + \phi(\tau)$$

where ϕ is the random variable that describes the probability law for taking the next step and τ is the time interval between subsequent steps. Finally, an original time series $x = x_1, ..., x_t, ..., x_T$ is transformed into $x\prime = x_1 + d_1, ..., x_t + d_t, ..., x_T + d_T$. Drifting can be used to create more robust training data sets and make the model less sensitive or more resistant to drift when it occurs in real data.

4) Genetic Algorithms: Genetic Algorithms (GA) are described as a class within Evolutionary Algorithms (EA), inspired by the process of evolution by natural selection. Their main goal is to generate high quality solutions to problems, based on bio-inspired operators such as selection, crossover and mutation[53]. In [54], genetic algorithms have been applied in data augmentation on time series of centers of pressure for the detection of neuropathies. Inspired by GA operators, a seismogram data augmentation technique was developed, which works as follows:

- Selection: Involves selecting real seismograms of the same class, without repeating, to use as parents.
- Crossover: Combines genetic information from two or more parent seismograms to produce a new one. It involves exchanging time series segments between the parents' chromosomes.
- Mutation: Introducing small changes in a small proportion of the population. Here, a smoothing operation is applied to the scales of the exchanged segments.
- Validation: For validation, KL and JS divergences are applied to measure similarity between datasets. Values closer to zero for the KL and JS divergences indicate greater similarity in the frequency-domain power distribution between the two seismic signals.

5) SpecAugment: SpecAugment is a data augmentation technique designed for Automatic Speech Recognition (ASR). Its purpose is to help the network learn useful features that are robust to certain time warping, partial loss of frequency information and partial loss of small signal segments. It is a simple and computationally inexpensive method, since it acts directly on the log-mel spectrogram as if it were an image, which can be applied online during training[55]. SpecAugment policies consist of frequency masking, time masking and time warping[56][57]. We apply SpecAugment on spectrograms considering frequency and time masking policies, as well as temporal warping.

6) Interpolation AEs: Autoencoders can interpolate in certain situations by decoding the convex combination of the

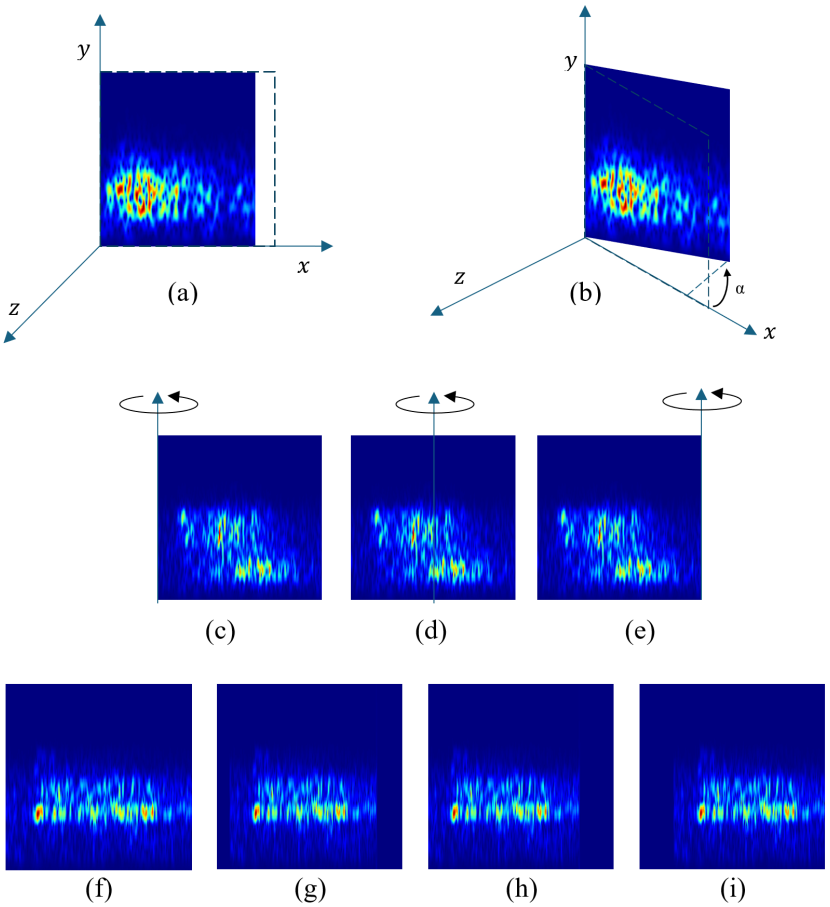


Fig. 2: Transformation by rotations, the time stretching (a) is produced when we apply the angle of rotation (b); The possible rotation axes are located at the beginning (c), center (d), and end (e) of the spectrogram, respectively. Application examples include: (f) original spectrogram, (g) 19% rotation around the central axis, (h) 23% rotation around the initial axis, and (i) 24% rotation around the final axis.

latent codes of two data points and producing an output that semantically blends the features of the data points[58]. First, an input $x \epsilon \mathrm{R}^{d_x}$ is passed through an "encoder" $z = f_\theta(x)$ parametrized by θ to obtain a latent code $z \epsilon \mathrm{R}^{d_z}$. The latent code is then passed through a "decoder" $\hat{x} = g_\phi(z)$ parametrized by ϕ to produce an approximate reconstruction $\hat{x} \epsilon \mathrm{R}^{d_x}$ of the input x. We consider the case where f_θ and g_ϕ are implemented as multi-layer neural networks. The encoder and decoder are trained simultaneously (i.e. with respect to θ and ϕ) to minimize some notion of distance between the input x and the output $\hat{x}$, for example the squared L_2 distance $\|x - \hat{x}\|^2$. Interpolating using an autoencoder describes the process of using the decoder g_ϕ to decode a mixture of two latent codes. Typically, the latent codes are combined via a convex combination, so that interpolation amounts to computing $x_\alpha = g_\phi(\alpha z_1 + (1 - \alpha) z_2)$ for some $\alpha \epsilon [0, 1]$ where $z_1 = f_\theta(x_1)$ and $z_2 = f_\theta(x_2)$ are the latent codes corresponding to data points x_1 and x_2.

C. Model performance - Metrics

Machine Learning metrics allow you to quantify the performance of a machine learning model once it is already trained, in addition to comparing it with generated Machine Learning models and the current situation. Evaluating the Machine Learning algorithm is an essential part of any project. Most of the time we use classification accuracy to measure the performance of our model, however, it is not enough to truly judge the model, which is why different types of evaluation metrics are considered. Among the different classification metrics, we can mention:

TABLE III: Confusion matrix.

		Predicted	
		Positive (P)	Negative (N)
Actual	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

- Confusion Matrix: Tabular display of truth labels versus model predictions. It's not exactly a performance metric, but rather a kind of basis on which other metrics evaluate results. Where TP are true positives, TN are true negatives, FP are false positives and FN are false negatives.
- Recall (Sensitivity): Calculates the real quantity that the model is able to identify.

$$Recall = \frac{TP}{TP + FN}$$

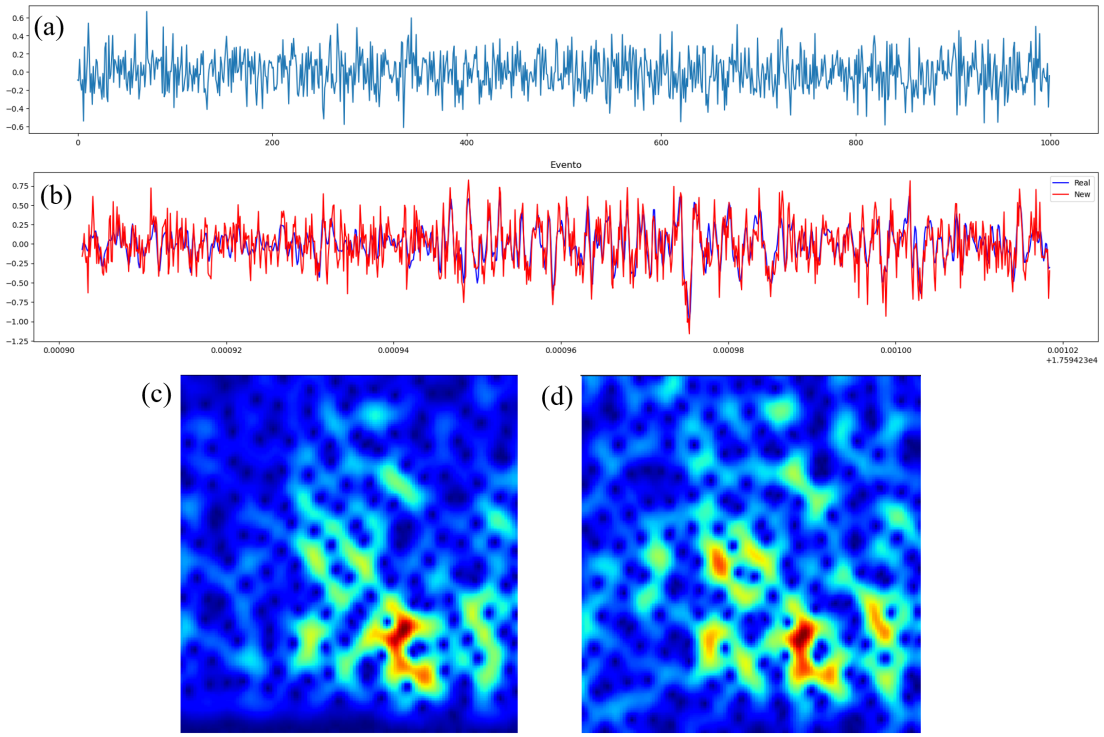


Fig. 3: Generating new signal with Jittering. (a) Gaussian noise with $N(\mu = 0, \sigma = 0.2)$ distribution. (b) Original (blue) and generated (red) signal. (c) Spectrogram of the original signal, and (d) of the generated signal with jittering.

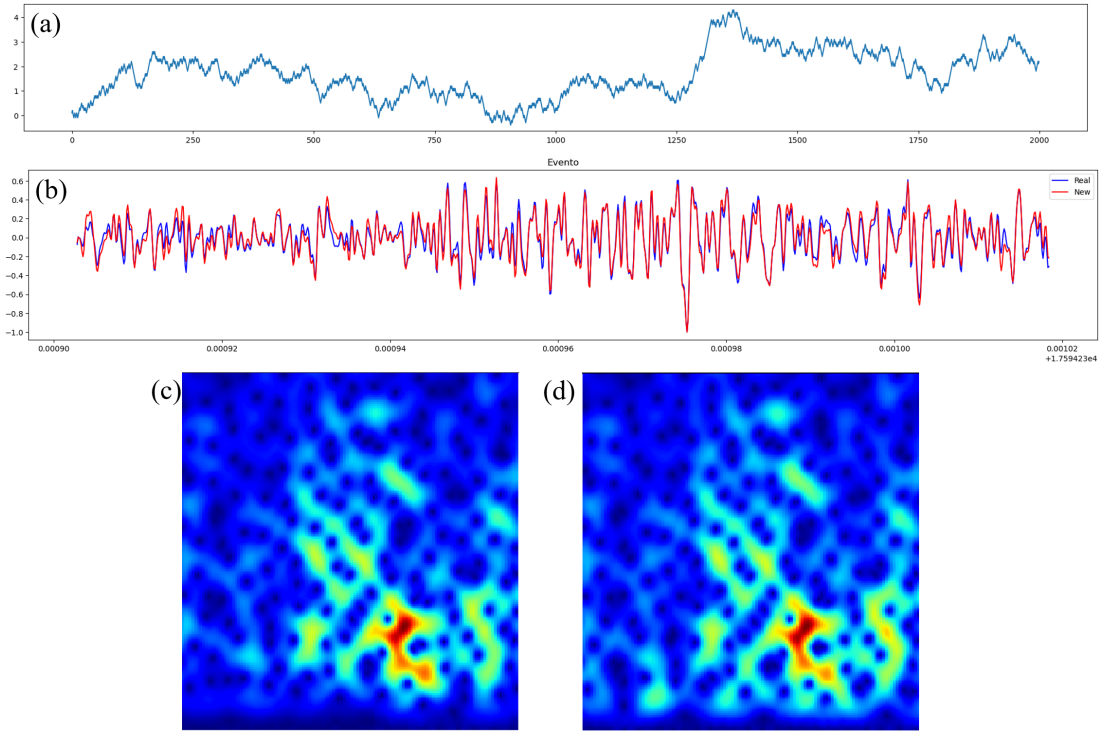


Fig. 4: Generating new signal with Drifting. (a) Drifting signal with $\tau = 0.2$. (b) Original (blue) and generated (red) signal. (c) Spectrogram of the original signal, and (d) of the generated signal with drifting.

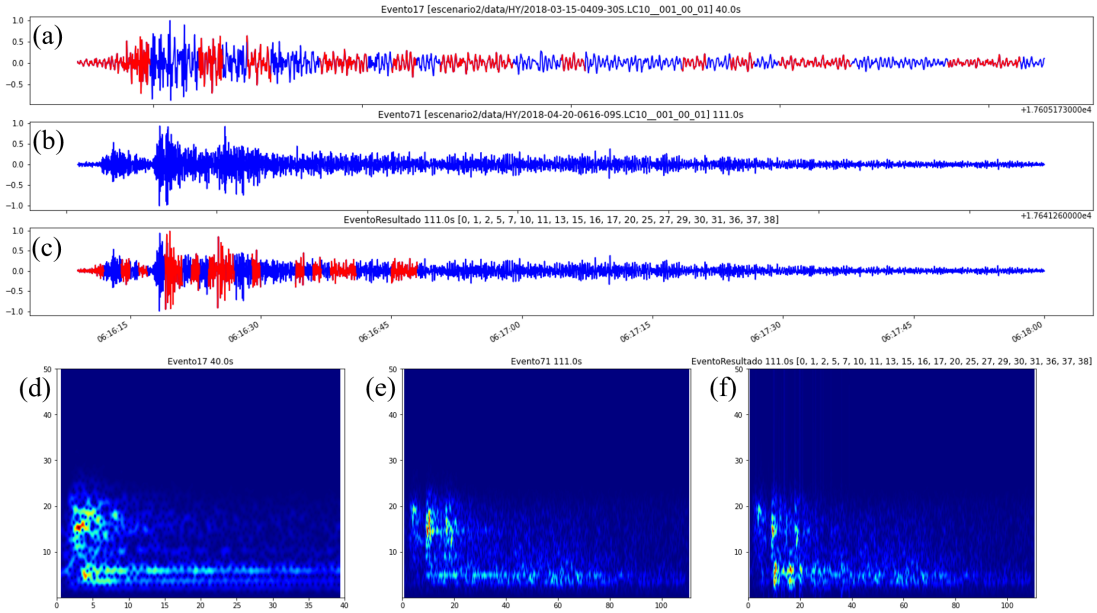


Fig. 5: Generating new signal with Genetic Algorithm without time fitting. (a) and (b) are parent signals, red color indicates crossing segments. (c) Generated signal with Genetic Algorithm. (d) and (e) are spectrogram of parent signals. (f) Spectrogram of the generated signal with Genetic Algorithm.

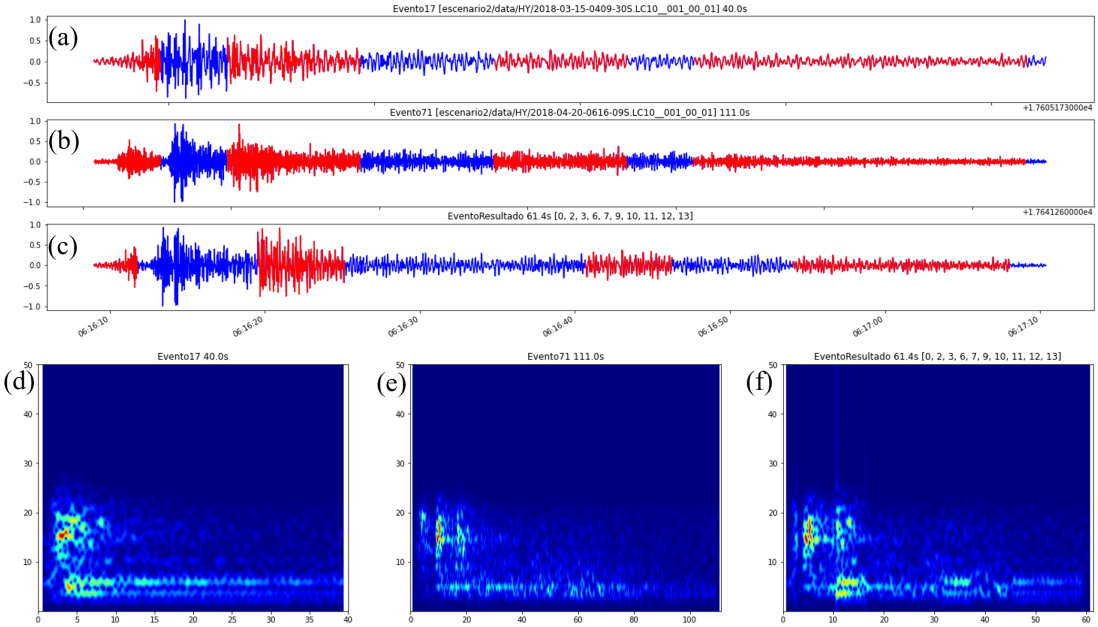


Fig. 6: Generating new signal with Genetic Algorithm with time fitting. (a) and (b) are parent signals, red color indicates crossing segments. (c) Generated signal with Genetic Algorithm. (d) and (e) are spectrogram of parent signals. (f) Spectrogram of the generated signal with Genetic Algorithm.

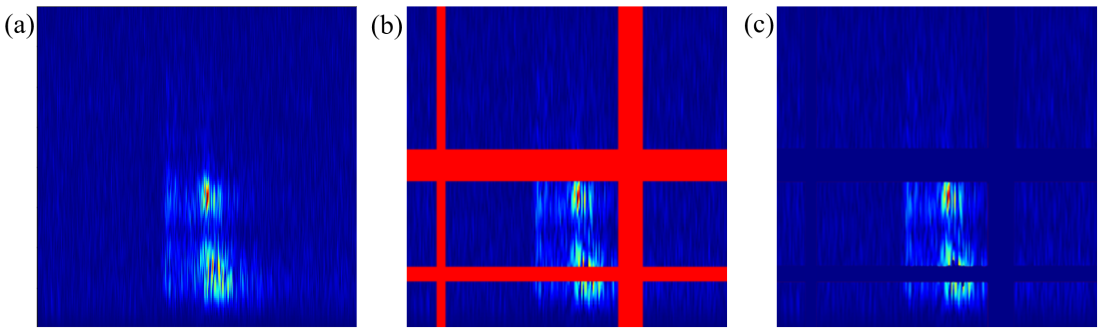


Fig. 7: Generating new signal with SpecAugment. (a) Original spectrogram. Spectrogram with two time-frequency distortions. The distortions are shown in red (b), and with white noise in (c).

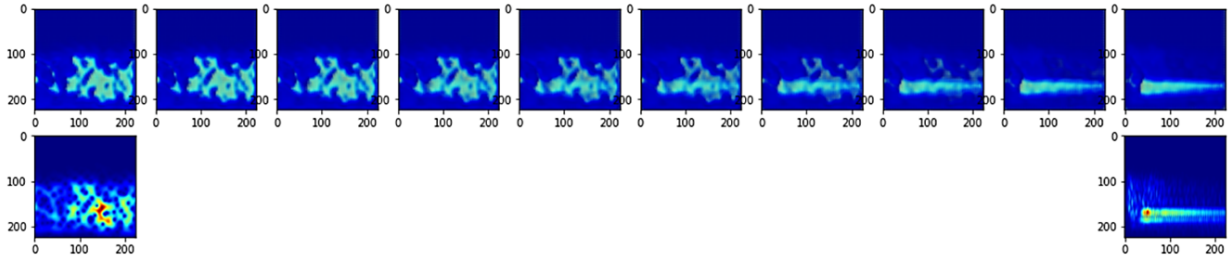


Fig. 8: Event interpolation (HY) at different intensity levels generates more diversity.

- Precision: Calculates the quality of the model in classification tasks. What cases identified as real are.

$$Precision = \frac{TP}{TP + FP}$$

- F_β: Combines precision and recall measures into a single value, making it easier to compare performance between several models. β is a real positive factor, which gives more importance to precision than to recall if it is greater than 1. This is a metric widely used in problems in which the data set to be analyzed is unbalanced.

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall}$$

- Acurracy: Measures the percentage of correct cases. It usually has problems with unbalanced datasets.

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

- Specificity: The number of genuinely negative examples that the model identified divided by the total number of negative examples in the data set.

$$Specificity = \frac{TN}{TN + FP}$$

D. Evaluation metrics for data augmentation

Data augmentation techniques have become a widely strategy adopted for improve the generalization capacity of deep neural networks. However, a very important aspect is the concepts of similarity and diversity in the data [36]. The similarity captures the resemblance between the original and augmented data, and the diversity measure the variation in complexity between the original and augmented data.

1) Fréchet Inception Distance (FID): The Fréchet Inception Distance is a primary metric for evaluating the quality of images generated by Generative Adversarial Networks (GANs)[37]. The main objective of the FID is to quantify the similarity between generated and real images[38]. It is considered an improvement over previous metrics such as the Inception Score (IS), as the FID uses statistics from real world samples for comparison, unlike the IS, which does not[39]. The key idea behind the FID is to embed real and generated images into a vision-relevant feature space and then calculate a distance between the distributions of these two embeddings. In practice, this feature space is typically the penultimate layer (pool3, with 2048 features) of an Inception-V3 classifier network pre-trained on ImageNet[39].

$$FID(\mu_r, \Sigma_r, \mu_g, \Sigma_g) = \|\mu_r - \mu_g\|_2^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{\frac{1}{2}})$$

where (μ_r, Σ_r) and (μ_g, Σ_g) denote the sample main and covariance of the embeddings of the real and generated data, respectively, and $Tr(\cdot)$ indicates the matrix trace[39]. The theoretical range of the FID is $[0, \infty)$, since it is a distance measure. Lower FID scores indicate better image quality[42]. Theoretically, an FID of 0 means that the feature distributions of the generated and real images are identical (in terms of mean and covariance according to the Gaussian approximation)[38]. FID has been found to correlate reasonably well with human judgments of the fidelity of generated images[38].

2) Kernel Inception Distance (KID): The squared Maximum Mean Discrepancy (MMD) measurement is used by KID to determine the difference between the feature representations of real and generated images. The features are derived from the Inception network. Unlike the Fréchet Inception Distance[38], KID's unbiased estimator increases its reliability, particularly

when there are less test images than the dimensions of the inception features. The lower KID signifies that there are more visual similarities between real and generated images[43]. KID use a polynomial kernel $k(x, y) = (\frac{1}{d}x^T \cdot y+1)^3$ where d is the dimension of the Inception representation vector, and x and y are two feature vectors. The kernel $K(x, y) = k(\phi(x), \phi(y))$ can be used as an MMD for input images, and images can be converted to Inception representations through ϕ mapping[44]. Since it's a distance metric, a KID value of 0 indicates a perfect match between the two feature distributions being compared (the one from the real images and the one from the generated ones). Theoretically, if the distributions are identical, the distance is zero. A key mathematical advantage of the KID is that it has a simple, unbiased estimator, unlike the FID[44].

3) Multi-Scale Structural Similarity (MS-SSIM): MS-SSIM is a multi-scale improve version of SSIM that tries to eliminate aspects of an image that are not crucial for human perception[47][48]. The SSIM calculation (the basis of MS-SSIM) compares three principal components: luminance (intensity), contrast, and structure. It uses simple statistical moments such as the mean and standard deviation in local windows to obtain a similarity score.

$$MS{-}SSIM(x, y) = [l_M(x, y)]^{\alpha_M} \cdot \prod_{j=1}^{M} [c_j(x, y)]^{\beta_j} \cdot [s_j(x, y)]^{\gamma_j}$$

Where $l_M(x, y)$ represents the luminance component between images x and y, $c_j(x, y)$ denotes the contrast component at scale j, and $s_j(x, y)$ reflects the structural similarity in scale j. The parameters α_M, β_j, and γ_j define the relative importance of the luminance, contrast, and structure components, respectively. Lastly, M signifies the number of scales employed in the MS-SSIM calculation[46]. The range of MS-SSIM values is between 0 and 1, and images with higher MS-SSIM values are perceived as more similar[45], with 1 indicating perfect similarity. Lower MS-SSIM scores indicate higher diversity[40][42]. Unlike metrics such as FID (Fréchet Inception Distance) or KID (Kernel Inception Distance), the MS-SSIM does not require a pre-trained Inception network[49]. For this reason, is not recommended to use the IS, KID or FID because they already use weights from the inception network, which are valid for images similar to those in the ImageNet dataset. These metrics cannot be used as over fingerprint datasets because are not part of the ImageNet dataset[49]. FID indicate better image quality. Lower MS-SSIM scores indicate higher diversity[40]. Smaller FID and KID values indicate more realistic images on image translation[41]. Because our spectrograms dataset is not part of ImageNet dataset, we used the MS-SSIM score.

IV. Results

We had unbalanced data, with 3% (HY), 9% (LP), 36% (TC), 8% (TR) and 44% (VT), see Table 2. We generate new data means, data augmentation techniques such as Rotation, Jittering, Drifting, Scaling, Flipping, Genetic Algorithms, and SpecAugment. We designed a lot of experiments based on each data augmentation method (Table 3). Each new balanced dataset was considered for Transfer Learning with the AlexNet pre-trained model. On the other hand, we trained several models of Deep Learning, ad hoc, with new balanced datasets.

A. Real

The original unbalanced dataset of 6,145 events is considered, which was divided into the proportions of 80/20, for training and testing the model, respectively. This experiment will show the influence of an unbalanced dataset on the accuracy of the probabilistic model.

B. Rotation

The original unbalanced dataset of 6,145 events is considered, which is balanced considering rotation as a given technique, so that the number of examples of the underrepresented classes is increased, obtaining a dataset of 1.3430 events. This data set will show us the effectiveness of this technique of increasing data in the construction of the classification model. In addition, another balanced dataset of 13,430 purely artificial events were generated, 2,686 examples for each event, for comparison purposes. The parameters used for the rotation were in the rotation range of 5% to 25%, and with random rotation axis (start, center, end), without repetition.

C. Jittering

The original unbalanced dataset of 6,145 events is considered, and data augmentation is applied to the dataset to balance it by increasing examples from underrepresented classes, resulting in a dataset of 13,430 events. The parameters used are Gaussian noise with a distribution of $N(\mu = 0, \sigma = 0.2)$. A balanced dataset of 13,430 events was also generated entirely from artificial data.

D. Drifting

The original unbalanced dataset of 6,145 events is considered, to which the data augmentation technique is applied in such a way that the dataset is balanced by increasing examples from underrepresented classes, resulting in a dataset of 13,430 events. The parameters used are random values from defined real intervals $[a, b]$. A balanced dataset of 13,430 events was also generated entirely from artificial data.

E. Genetic Algorithm)

Using the data augmentation technique of Genetic Algorithms, a balanced dataset of 13,430 artificial events was generated, 2,686 examples for each event. Different values of the signal proportion parameters of the parent events, as well as the number of crossing segments in the technique, are considered for the generation of the new dataset. This dataset will show the effectiveness of this data augmentation technique in building the model. In this type of data augmentation technique, there are two variants:

- Without time adjustment: It is verified that the parents can be segmented into the number of crossover segments and time; the crossover occurs at the defined times. The

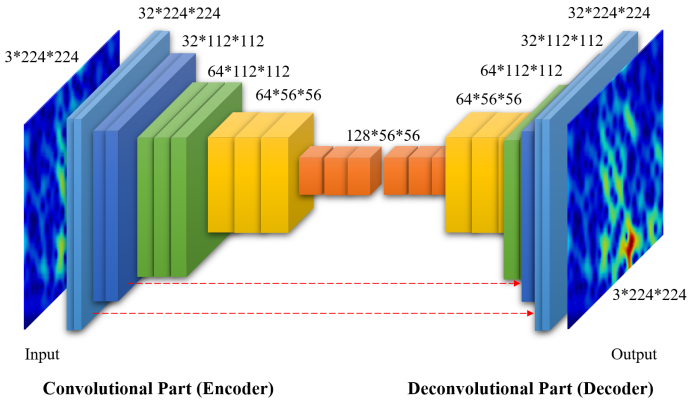


Fig. 9: Convolutional AutoEncoder of spectrograms.

parameters used are three crossover segments with a time of 5 seconds each.

- With time adjustment: The number of crossover segments and the signal proportion to be used for the crossover are defined. The segment times are calculated proportionally for both parents.

F. SpecAugment

Using SpecAugment's data augmentation technique, a balanced dataset of 13,430 artificial events was generated, 2,686 examples for each event. This dataset will show the effectiveness of this data augmentation technique in building the model. To generate the new data set, different values of the maximum percentage parameters of each time and frequency mask were considered, in addition to the number of masks for each of them. The parameters used are two time and frequency masks, with 10% of the signal in each mask.

G. Interpolation AEs

This technique was used to generate a balanced dataset of 13,430 purely artificial events. For this technique, the randomly selected parameter $\alpha \epsilon [0.4, 0.6]$ was used, with a designed ad hoc convolutional autoencoder to reconstruct the interpolated signal, see Figure 9.

The experimental evidence showed the data augmentation techniques' importance in the precision of deep learning models since models trained without considering data augmentation techniques presented low precision (Table 4). For comparing our models with the state-of-art, we consider performance metrics for accuracy, precision, recall, specificity, F1.

V. Discussion

The various experiments in this research show the performance of the probabilistic model on unbalanced datasets. Specifically, dataset (i) produced a model biased towards the LP, TC and VT events (Table 4). It can be seen that balanced datasets generate higher performance models, since they expand the examples of the event feature space. The models were then tested on real data, showing competitive levels of classification in most cases. Transfer learning allowed deep learning models to be generated in less time (Table 4). In Table 4 and Table 5. we can see that drifting data augmentation technique achieves best result in almost all of pre trained models. This can be because drifting technique covers best the data space. The best case is drifting (xi) with a drift range 0.0001 - 0.001 with an accuracy of 97.6%, model trained over entirely artificial dataset. The next interesting result is the rotation (iii) with a range of 5% - 25% that achieves an accuracy of 94.2% over another artificial dataset. In both previous cases, the F1 score metrics are similar in their accuracy respective. Although both models were generated from entirely artificial data, they show good performance in addition to the tests on real data, with accuracies of 97.2% for (xi) and 93.7% for (iii) (Table 5), although the highest accuracy value is recorded in (x) with 98.3% with real and artificial data. It can be observed that the dataset composed of real and artificial data (x) generated by drifting in the range of 0.0001-0.001 reaches an accuracy of 93.8%. Then the dataset of artificial data (ix) generated by drifting in range 0.001-0.01, reaches an accuracy of 93.5%. Then we have the artificial data dataset (xxv) generated by Autoencoder Interpolation in the range 40%-60%, it obtains an accuracy of 93.1% (Table 4). These models also show good performance in tests with real data, with accuracies of 98.3% for (x), 93.7% for (ix) and 80.7% for (xxv) (Table 5). Only the last model has a large distance between training and test accuracy on real data. To obtain a generalized dataset, data from several stations were considered to generate a multi-station generalized probabilistic model. Furthermore, it was observed that models trained with completely artificial data achieved high recognition accuracy with real data in most cases (Table 5). Figure 11 shows that drifting, rotation, interpolation, and specification have slightly better accuracy values than the other techniques. In general, the trained models perform better when tested on real data than during training. Regarding the similarity and diversity metrics, it can be seen that the MS-SSIM values for the rotation and drifting techniques are similar, although the former shows more diversity, depending on the generation parameters (Table 6). It can also be seen that Jittering shows higher diversity in (v) and (iv), with MS-SSIM values of 0.26 and 0.29, respectively. The most similar are Interpolation (xxv) and Drifting (vi), with MS-SSIM values of 0.42 and 0.41, respectively. Figure 12 shows the concentration of points in terms of the value of the MS-SSIM similarity metric vs. the accuracy of the model training. Only two distant points are seen, the dataset with only real data (i), which is distant due to having a low accuracy because it is an unbalanced dataset. Another distant point is the drifting dataset (vii) with a range of 0.01-0.1 because it shows more similarity. Zooming in on Figure 13, it can be seen that almost all the datasets have the same level of diversity, including the real dataset, but the artificial Jittering dataset (v) presents more diversity. Comparing artificial datasets with real and artificial datasets, within the same DA method, it can be seen that the artificial datasets present more diversity and training performance with Jittering, SpecAugment (xx)(xxi), and Rotation. In Figure 14, the most artificial real datasets present better diversity and testing accuracy values than the artificial datasets in the same

TABLE IV: Statistics related to the transfer learning over AlexNet pre-trained model.

Dataset	Data type	Data Augmentation	Balanced dataset	Epochs	Total data	Train data	Test data	Training time (min.)
(i)	Real	No	NO	10	6,145	4,916	1,229	4.2
(ii)	Real + Artificial	Rotation(5%-25%)	YES	20	13,430	10,744	2,686	18.7
(iii)	Artificial	Rotation(5%-25%)	YES	20	13,430	10,744	2,686	20.9
(iv)	Real + artificial	Jittering(0.2)	YES	20	13,430	10,744	2,686	16.8
(v)	Artificiales	Jittering(0.2)	YES	20	13,430	10,744	2,686	17.2
(vi)	Real + Artificial	Drifting Drift(0.01-0.1)	YES	20	13,430	10,744	2,686	17.9
(vii)	Artificial	Drifting Drift(0.01-0.1)	YES	20	13,430	10,744	2,686	20.3
(viii)	Real + Artificial	Drifting Drift(0.001-0.01)	YES	20	13,430	10,744	2,686	21.7
(ix)	Artificial	Drifting Drift(0.001-0.01)	YES	20	13,430	10,744	2,686	21.0
(x)	Real + Artificial	Drifting Drift(0.0001-0.001)	YES	20	13,430	10,744	2,686	18.1
(xi)	Artificial	Drifting Drift(0.0001-0.001)	YES	20	13,430	10,744	2,686	20.9
(xii)	Real + Artificial	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	YES	20	13,430	10,744	2,686	16.7
(xiii)	Artificial	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	YES	20	13,430	10,744	2,686	17.0
(xiv)	Real + Artificial	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	YES	20	13,430	10,744	2,686	16.6
(xv)	Artificial	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	YES	20	13,430	10,744	2,686	16.9
(xvi)	Real + Artificial	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	YES	20	13,430	10,744	2,686	16.6
(xvii)	Artificial	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	YES	20	13,430	10,744	2,686	17.3
(xviii)	Real + Artificial	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	YES	20	13,430	10,744	2,686	16.5
(xix)	Artificial	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	YES	20	13,430	10,744	2,686	17.1
(xx)	Real + Artificial	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	YES	20	13,430	10,744	2,686	16.5
(xxi)	Artificial	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	YES	20	13,430	10,744	2,686	17.0
(xxii)	Real + Artificial	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	YES	20	13,430	10,744	2,686	16.6
(xxiii)	Artificial	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	YES	20	13,430	10,744	2,686	16.9
(xxiv)	Real + Artificial	Interpolation AE Signal percent(40%-60%)	YES	20	13,430	10,744	2,686	16.6
(xxv)	Artificial	Interpolation AE Signal percent(40%-60%)	YES	20	13,430	10,744	2,686	17.9

TABLE V: Performance of transfer learning models trained to 1-20 [Hz].

Dataset	Data augmentation	Accuracy	F1	Recall	Precision	Cohen Kappa	Average Acurracy	Class Accuracy				
								HY	LP	TC	TR	VT
(i)	No	62.8	63.9	62.8	66.7	67.6	62.8	38.9	74.8	73.5	33.3	93.4
(ii)	Rotation(5%-25%)	91.5	91.4	91.5	91.4	89.2	91.5	98.4	92.4	83.5	93.3	89.6
(iii)	Rotation(5%-25%)	94.2	94.0	94.2	94.0	92.6	94.2	99.8	96.3	83.1	96.1	95.8
(iv)	Jittering(0.2)	90.2	90.0	90.2	90.2	87.5	90.2	94.0	93.6	77.9	95.9	89.4
(v)	Jittering(0.2)	91.1	91.1	91.1	91.2	88.8	91.1	97.1	95.3	85.2	93.3	84.3
(vi)	Drifting Drift(0.01-0.1)	84.1	83.5	84.1	83.7	79.6	84.1	94.4	90.7	62.1	82.4	90.8
(vii)	Drifting Drift(0.01-0.1)	87.7	87.6	87.7	87.9	84.6	87.7	98.7	88.5	78.1	91.3	81.7
(viii)	Drifting Drift(0.001-0.01)	90.8	90.6	90.8	90.6	88.3	90.8	96.9	95.7	77.7	90.2	93.6
(ix)	Drifting Drift(0.001-0.01)	93.5	93.4	93.5	93.5	91.8	93.5	98.9	95.9	83.0	97.0	92.8
(x)	Drifting Drift(0.0001-0.001)	93.8	93.5	93.8	93.7	92.0	93.8	99.8	100.0	81.0	98.0	90.0
(xi)	Drifting Drift(0.0001-0.001)	97.6	97.6	97.6	97.6	97.0	97.6	100.0	99.0	93.4	100.0	95.6
(xii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	87.8	87.6	87.8	87.7	84.6	87.8	95.8	91.5	73.6	91.7	86.7
(xiii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	86.4	86.2	86.4	86.3	82.8	86.4	92.6	90.7	74.6	85.6	88.6
(xiv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	86.4	86.1	86.4	86.2	82.7	86.4	94.6	88.5	72.0	86.0	91.0
(xv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	85.8	85.5	85.8	85.9	81.9	85.8	90.4	94.4	75.1	77.3	91.8
(xvi)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	84.9	84.7	84.9	84.7	80.8	84.9	88.1	91.5	74.1	80.0	90.8
(xvii)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	83.0	82.9	83.0	83.0	78.6	83.0	89.2	83.1	77.2	73.9	91.8
(xviii)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	83.7	83.4	83.7	83.8	79.3	83.7	94.0	84.3	75.0	73.2	91.8
(xix)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	83.2	83.2	83.2	83.3	78.8	83.2	89.5	86.0	75.3	76.9	88.2
(xx)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	90.5	90.2	90.5	90.2	87.9	90.5	98.2	95.5	77.9	90.2	90.4
(xxi)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	91.7	91.5	91.7	91.5	89.4	91.7	97.5	94.8	83.5	89.5	93.2
(xxii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	88.6	88.6	88.6	88.5	85.7	88.6	94.9	86.8	80.5	89.5	91.4
(xxiii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	86.6	86.4	86.6	86.4	83.1	86.6	96.0	85.6	75.8	87.2	88.2
(xxiv)	Interpolation AE Signal percent(40%-60%)	87.5	87.4	87.5	87.6	84.1	87.5	89.3	90.9	76.9	89.3	91.0
(xxv)	Interpolation AE Signal percent(40%-60%)	93.1	93.1	93.1	93.3	91.3	93.1	97.6	96.5	83.7	96.5	91.4

TABLE VI: Test of trained models over the real dataset of 6,145 events.

Dataset	Data augmentation	Accuracy	F1	Recall	Precision	Cohen Kappa	Average Acurracy	Class Accuracy				
								HY	LP	TC	TR	VT
(i)	No	-	-	-	-	-	-	-	-	-	-	-
(ii)	Rotation(5%-25%)	96.6	96.2	96.6	95.9	95.7	96.6	97.7	95.3	97.0	94.9	98.1
(iii)	Rotation(5%-25%)	93.7	88.9	93.7	85.5	86.9	93.7	100.0	96.0	82.5	93.8	96.0
(iv)	Jittering(0.2)	96.2	96.0	96.2	95.8	95.0	96.2	94.8	96.7	95.8	96.2	97.5
(v)	Jittering(0.2)	93.9	87.3	93.9	82.9	86.2	93.9	99.5	97.1	86.0	95.8	91.3
(vi)	Drifting Drift(0.01-0.1)	91.5	89.4	91.5	88.1	90.0	91.5	93.0	95.5	90.0	81.5	97.6
(vii)	Drifting Drift(0.01-0.1)	66.6	53.5	66.6	55.2	48.3	66.6	69.5	96.5	42.0	56.5	68.4
(viii)	Drifting Drift(0.001-0.01)	95.0	95.4	95.0	95.8	94.7	95.0	93.0	97.1	94.9	91.3	98.8
(ix)	Drifting Drift(0.001-0.01)	93.7	86.7	93.7	82.4	85.5	93.7	98.1	97.4	83.4	97.7	91.8
(x)	Drifting Drift(0.0001-0.001)	98.3	96.7	98.3	95.3	96.0	98.3	99.5	100.0	95.5	98.5	97.8
(xi)	Drifting Drift(0.0001-0.001)	97.2	93.8	97.2	91.0	92.9	97.2	100.0	98.3	93.3	99.6	94.9
(xii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	95.9	93.7	95.9	91.8	93.8	95.9	96.7	97.6	94.1	94.3	97.0
(xiii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	82.7	73.6	82.7	70.0	71.8	82.7	86.9	93.4	65.2	78.3	89.8
(xiv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	95.2	93.6	95.2	92.2	93.6	95.2	95.3	96.7	93.2	92.4	98.3
(xv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	75.2	67.3	75.2	64.9	64.3	75.2	77.5	91.2	57.3	61.4	88.8
(xvi)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	93.3	94.0	93.3	94.9	93.8	93.3	90.6	98.1	95.5	84.1	98.3
(xvii)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	81.4	74.2	81.4	70.7	72.4	81.4	85.4	83.9	67.9	78.6	91.1
(xviii)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	94.0	94.0	94.0	94.2	93.8	94.0	95.3	94.8	95.1	86.4	98.5
(xix)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	80.1	68.1	80.1	64.9	65.8	80.1	88.7	92.5	56.9	76.6	85.9
(xx)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	96.7	96.4	96.7	96.1	95.4	96.7	98.6	98.6	95.8	92.4	98.2
(xxi)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	95.4	91.1	95.4	87.8	89.8	95.4	100.0	97.2	88.1	96.6	95.1
(xxii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	95.5	95.7	95.5	95.9	94.9	95.5	97.7	90.5	96.3	94.9	98.4
(xxiii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	93.6	87.5	93.6	83.7	85.2	93.6	100.0	95.1	82.8	97.5	92.4
(xxiv)	Interpolation AE Signal percent(40%-60%)	90.4	91.8	90.4	93.4	92.0	90.4	82.2	90.5	94.7	86.4	98.1
(xxv)	Interpolation AE Signal percent(40%-60%)	80.7	71.0	80.7	67.5	67.8	80.7	88.7	85.8	60.4	80.7	88.1

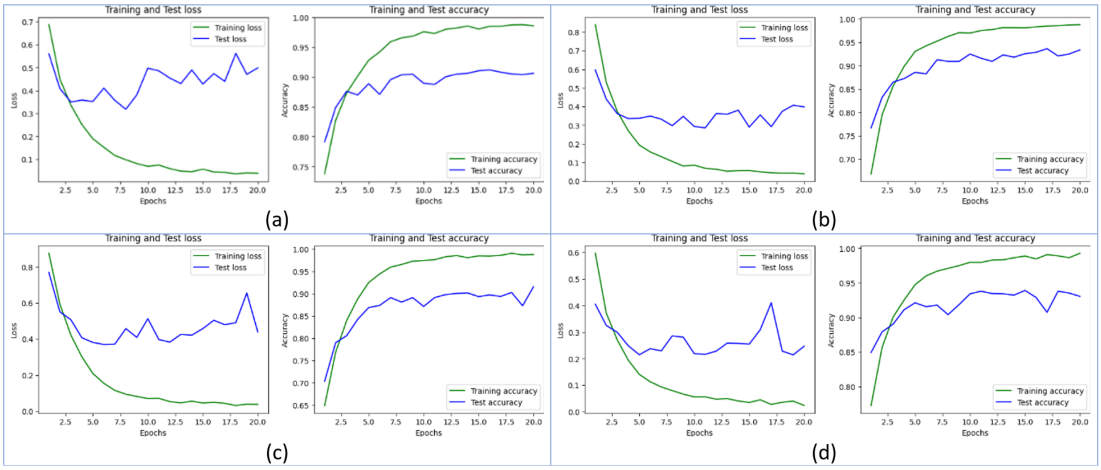


Fig. 10: Training, and validation performance scores of the transfer learning with AlexNet for the experiment: a) training and test loss for the experiment (viii), b) (ix), c) (xxi), and d) (xxv).

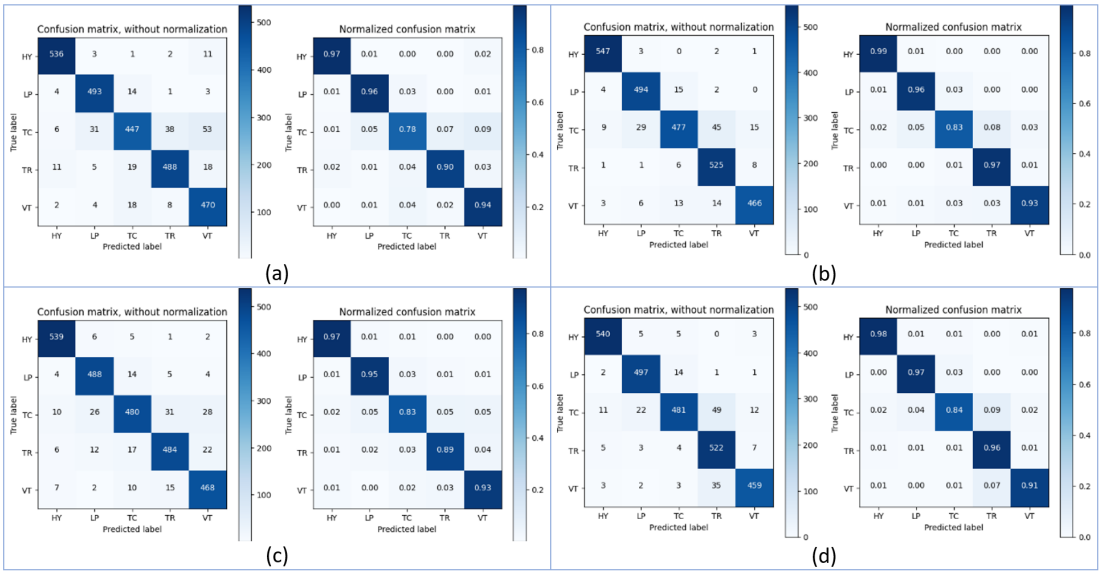


Fig. 11: Confusion matrix without normalization and normalized for experiments a) (viii), b) (ix), c) (xxi) and d) (xxv).

TABLE VII: Similarity and Diversity metrics for experimental datasets.

Dataset	Data augmentation	Fréchet Inception Distance (FID)						Kernel Inception Distance (KID)						Multiscale Structural Similarity Index (MS-SSIM)					
		HY	LP	TC	TR	VT	FID	HY	LP	TC	TR	VT	KID	HY	LP	TC	TR	VT	MS
(i)	No	0.16	0.10	0.02	0.05	0.02	0.03	0.14	-0.75	3.78	0.31	-0.23	1.21	0.34	0.41	0.33	0.29	0.34	0.34
(ii)	Rotation(5%-25%)	0.70	0.61	0.50	0.45		0.56	4.81	-1.40	4.41	2.56		2.59	0.38	0.42	0.36	0.29		0.36
(iii)	Rotation(5%-25%)	0.03	0.01	0.04	0.02	0.03	0.02	1.62	0.15	1.16	-1.03	2.34	0.85	0.36	0.42	0.36	0.33	0.35	0.37
(iv)	Jittering(0.2)	10.09	5.46	7.65	8.88		8.02	56.99	18.80	40.98	48.72		41.37	0.28	0.36	0.27	0.26		0.29
(v)	Jittering(0.2)	0.01	0.01	0.02	0.02	0.03	0.02	-2.50	-2.35	3.38	0.24	-2.77	-0.80	0.26	0.33	0.26	0.21	0.23	0.26
(vi)	Drifting Drift(0.01-0.1)	34.80	30.48	35.93	30.01		32.81	102.53	71.83	89.02	83.60		86.75	0.40	0.46	0.42	0.35		0.41
(vii)	Drifting Drift(0.01-0.1)	0.01	0.02	0.01	0.01	0.01	0.01	-0.06	-0.10	0.29	0.51	-0.11	0.11	0.80	0.81	0.81	0.76	0.71	0.78
(viii)	Drifting Drift(0.001-0.01)	1.50	1.23	2.69	0.84		1.56	3.25	3.61	3.84	1.01		2.93	0.37	0.42	0.38	0.31		0.37
(ix)	Drifting Drift(0.001-0.01)	0.03	0.01	0.02	0.03	0.01	0.02	1.53	-1.02	0.30	1.14	-0.80	0.23	0.42	0.48	0.43	0.37	0.36	0.41
(x)	Drifting Drift(0.0001-0.001)	0.02	0.02	0.03	0.02		0.02	-4.42	4.68	-3.96	0.79		-0.73	0.35	0.40	0.33	0.29		0.34
(xi)	Drifting Drift(0.0001-0.001)	0.01	0.01	0.02	0.01	0.01	0.01	-5.60	-1.65	2.42	3.00	1.74	-0.02	0.35	0.40	0.32	0.31	0.34	0.34
(xii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	0.89	0.41	0.39	0.28		0.49	2.36	2.98	-1.94	2.55		1.49	0.38	0.40	0.35	0.30		0.36
(xiii)	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	0.02	0.01	0.00	0.01	0.01	0.01	0.45	-2.49	-0.40	-2.06	2.54	-0.39	0.35	0.40	0.37	0.30	0.36	0.36
(xiv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	1.65	0.83	0.76	1.03		1.07	1.51	4.59	2.90	1.29		2.57	0.35	0.41	0.35	0.30		0.35
(xv)	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	0.02	0.01	0.01	0.02	0.01	0.01	1.06	-0.48	-0.32	-2.37	-1.77	-0.77	0.35	0.40	0.37	0.33	0.40	0.37
(xvi)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	0.90	0.63	0.55	0.41		0.63	3.99	1.91	0.08	4.25		2.56	0.36	0.41	0.35	0.30		0.35
(xvii)	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	0.02	0.02	0.04	0.01	0.01	0.02	-0.28	-3.37	-0.14	1.87	-0.70	-0.52	0.34	0.41	0.32	0.29	0.34	0.34
(xviii)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	0.97	0.67	0.73	0.36		0.68	4.43	-0.28	3.54	0.97		2.16	0.35	0.41	0.33	0.30		0.35
(xix)	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	0.01	0.00	0.00	0.01	0.01	0.01	0.91	0.75	-0.73	2.75	0.86	0.91	0.31	0.41	0.35	0.30	0.34	0.34
(xx)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	1.53	1.48	1.85	1.21		1.52	6.23	4.41	6.33	2.06		4.76	0.37	0.41	0.33	0.30		0.35
(xxi)	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	0.00	0.01	0.02	0.01	0.01	0.01	-1.21	2.46	-0.33	-1.04	1.03	0.18	0.34	0.40	0.35	0.31	0.34	0.35
(xxii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	3.84	3.45	3.08	3.22		3.40	16.73	11.34	7.82	6.15		10.51	0.35	0.41	0.33	0.31		0.35
(xxiii)	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	0.01	0.01	0.01	0.00	0.01	0.01	0.31	1.29	0.47	0.72	-0.61	0.44	0.35	0.40	0.36	0.32	0.39	0.37
(xxiv)	Interpolation AE Signal percent(40%-60%)	2.31	2.46	2.18	1.88		2.21	9.48	3.04	4.03	7.12		5.92	0.38	0.44	0.36	0.33		0.38
(xxv)	Interpolation AE Signal percent(40%-60%)	0.01	0.01	0.01	0.01	0.01	0.01	-0.22	0.62	-0.89	-0.44	0.03	-0.18	0.43	0.48	0.41	0.37	0.41	0.42

TABLE VIII: Relationship between Data Augmentation and Dataset.

Code	Data augmentation	Dataset
N	Real dataset	i
R	Rotation (5%-25%)	ii, iii
J	Jittering (0.2)	iv, v
D	Drifting (0.01-0.1)	vi, vii
DD	Drifting (0.001-0.01)	viii, ix
DDD	Drifting (0.0001-0.001)	x, xi
G	Genetic Algorithm Segment Size(5s) Crossover Segments(3)	xii, xiii
GG	Genetic Algorithm Segment Size(5s) Crossover Segments(5)	xiv, xv
H	Genetic Algorithm Signal Percent(40%) Crossover Segments(10)	xvi, xvii
HH	Genetic Algorithm Signal Percent(45%) Crossover Segments(10)	xviii, xix
S	SpecAugment Frequency Percent(0%-10%) Frequency Masks(2) Time Percent(0%-10%) Time Marks(2)	xx, xxi
SS	SpecAugment Frequency Percent(0%-15%) Frequency Masks(2) Time Percent(0%-15%) Time Marks(2)	xxii, xxiii
I	Interpolation AE Signal percent(40%-60%)	xxiv, xv

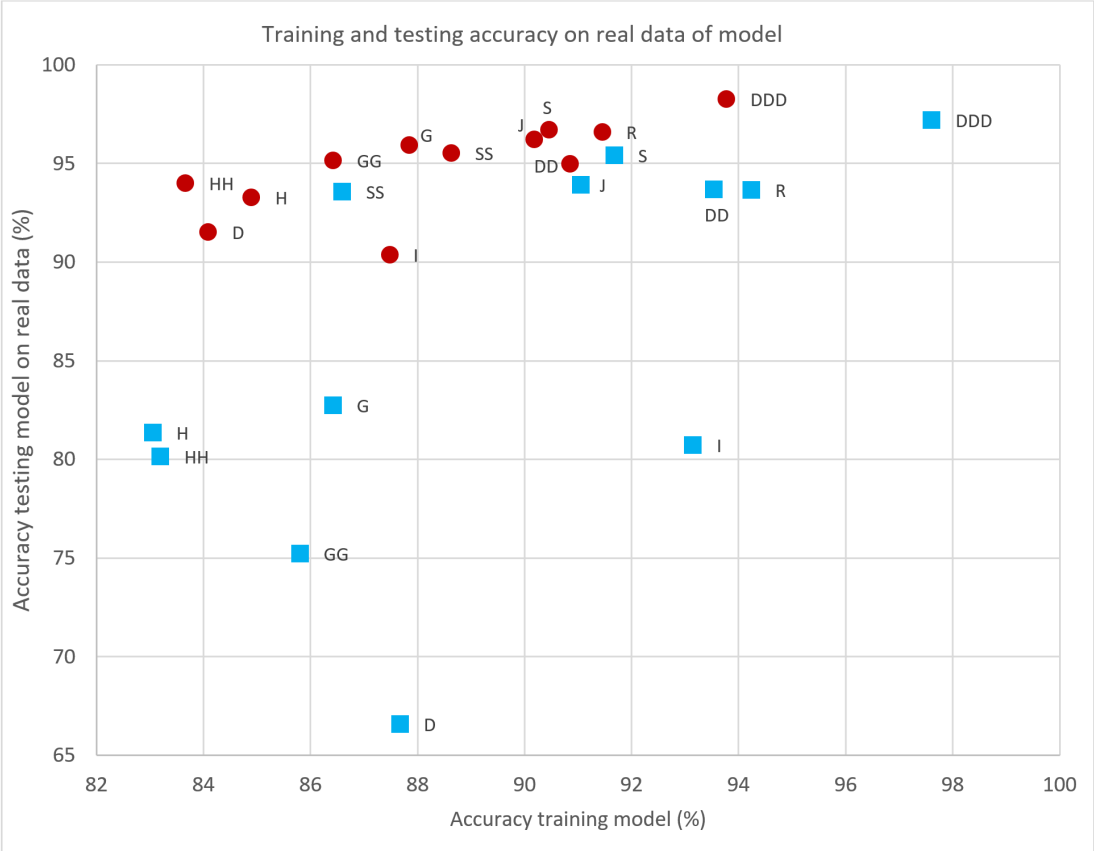


Fig. 12: Accuracy of model training and testing on real data. **Green** shape indicates real dataset with 62.8% of accuracy, **red** circles indicate datasets with real and artificial data, and **turquoise** squares are entirely artificial datasets. The codes are described in Table VIII.

DA method, and they are Drifting (x)(xi), rotation (ii)(iii), SpecAugment (xvii)(xviii), Drifting (viii)(ix).

VI. Conclusion

The current situation was reviewed in relation to the characterization and recognition of seismic-volcanic signal patterns, by reading scientific articles. Regarding the specific objectives of this work, in this study of the current situation in the recognition of seismic-volcanic signal patterns has been developed, especially with regard to deep learning, transfer learning and augmentation techniques. data. Many Data augmentation methods have been evaluated and developed to balance the data deposition of volcanic seismic signals. Taking all this into account, it can be stated that the specific objectives of this work have been effective, and, therefore, the general objective of this research has also been effective. It can be seen that the developed models and data augmentation methods are competitive with the baseline presented by scientific articles on the current situation. In addition to the following results:

- Data from multiple stations at Lascar volcano allowed us to build more generalized probabilistic models, avoiding bias when considering a single station.
- Data augmentation techniques are important to improve the performance metrics of deep learning models, as they

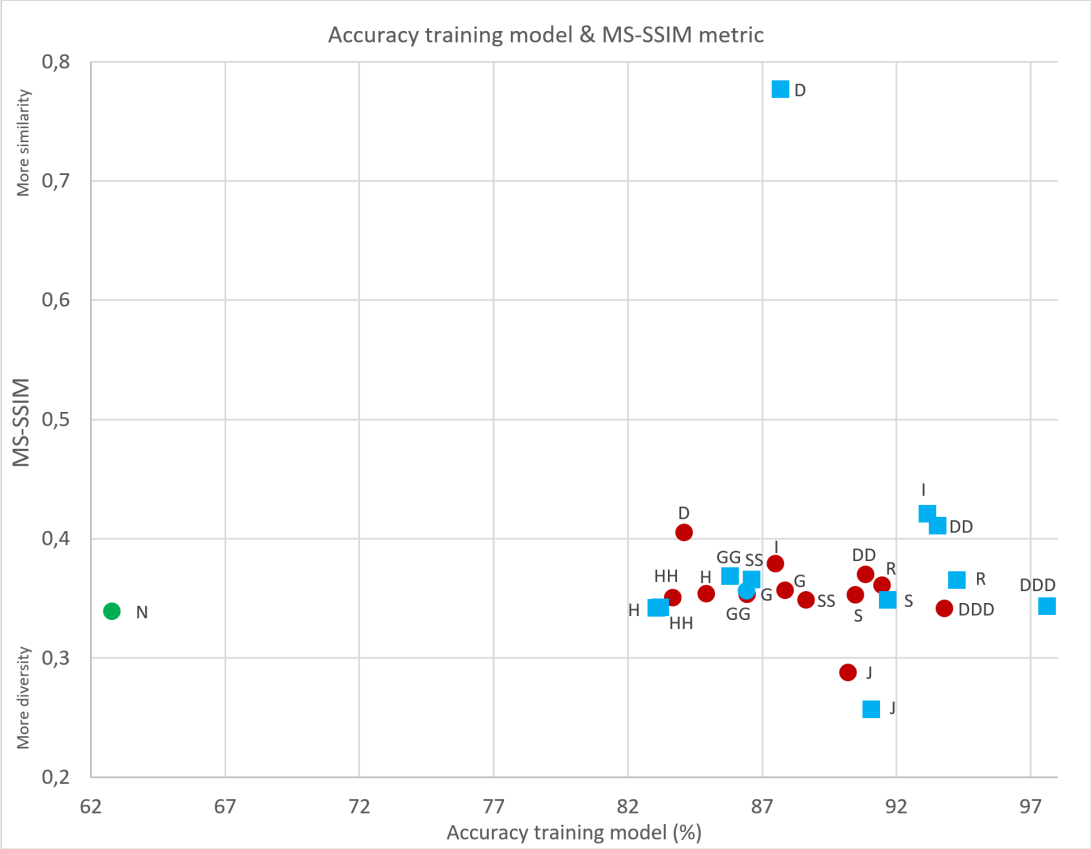


Fig. 13: Accuracy of model training and MS-SSIM metric. **Green** shape indicates real dataset, **red** circles indicate datasets with real and artificial data, and **turquoise** squares are entirely artificial datasets. The codes are described in Table VIII.

allow more examples to be generated within the feature space of seismic events.
- Similarity and diversity metrics allowed us to analyze the characteristics of the various techniques implemented.
- Transfer learning helped generate models much faster than training them from scratch.

A repository on GitHub is available to verify data at: https://github.com/franznet/daseismicsignals.

APPENDIX A
PROOF OF THE FIRST ZONKLAR EQUATION

Appendix one text goes here.

APPENDIX B

Appendix two text goes here.

ACKNOWLEDGMENT

The authors would like to thank...

Franz Yupanqui Machaca earned his Master's degree in Computer Engineering from the Universidad Católica del Norte in Chile in 2023. He is currently pursuing his Ph.D. in Sustainable Engineering at the Universidad Católica del Norte, focusing on the development of advanced algorithms for seismic-volcanic signal recognition. His research interests include deep learning techniques, transfer learning, data augmentation, computational intelligence in signals, data analytics, and more.

John Doe Biography text here.

Jane Doe Biography text here.

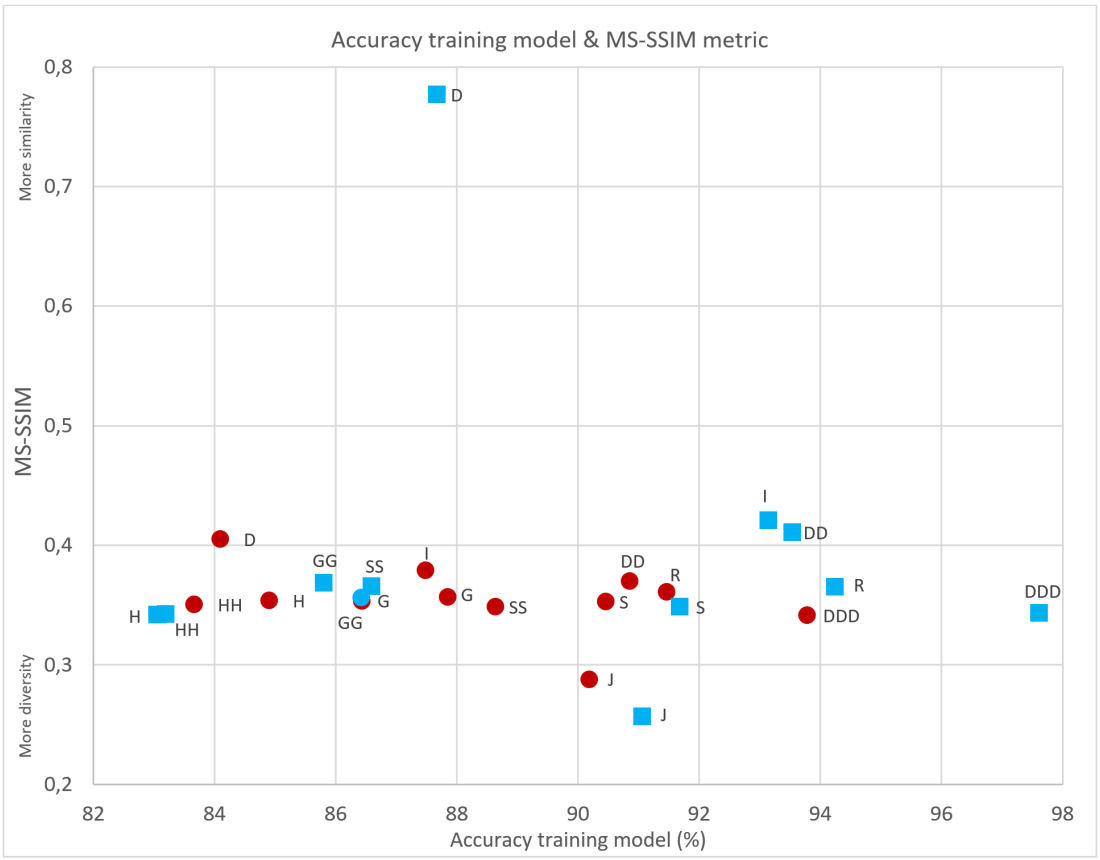


Fig. 14: Zooming at figure of model training accuracy and MS-SSIM metric. **Green** shape indicates real dataset, **red** circles indicate datasets with real and artificial data, and **turquoise** squares are entirely artificial datasets. The codes are described in Table VIII.

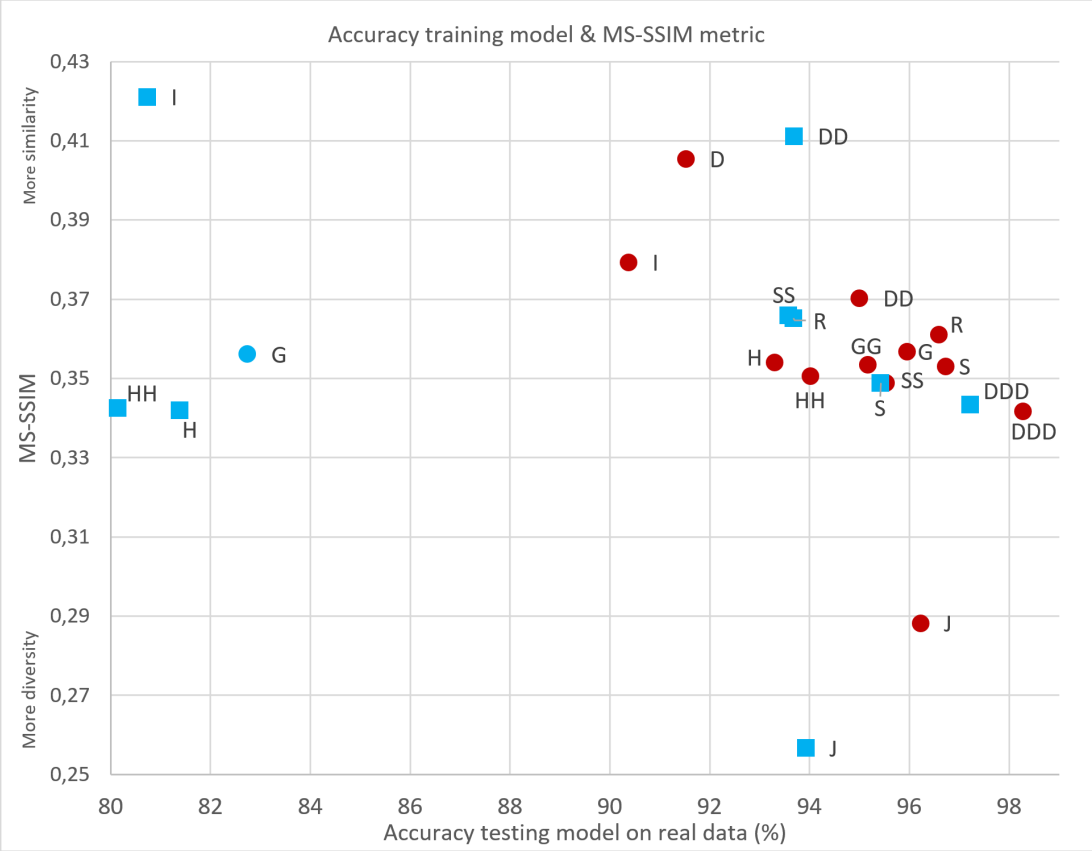


Fig. 15: Accuracy of model testing and MS-SSIM metric. **Green** shape indicates real dataset, **red** circles indicate datasets with real and artificial data, and **turquoise** squares are entirely artificial datasets. The codes are described in Table VIII.