Research Article GU J Sci 38(1): 75-97 (2025) DOI: 10.35378/gujs.1357317

Gazi University

Journal of Science

http://dergipark.gov.tr/gujs

A Deep Learning Approach based on Ensemble Classification Pipeline and Interpretable Logical Rules for Bilingual Fake Speech Recognition

Emre Beray BOZTEPE[1], Bahadir KARASULU[2, *]

[1]*Institute of Computer Science, Faculty of Mathematics and Computer Science, Wroclaw University, Fryderyka Joliot-Curie 15, 50-300 Wroclaw, Poland*
[2]*Department of Computer Engineering, Faculty of Engineering, Canakkale Onsekiz Mart University, Terzioglu Kampusu,17020, Canakkale, Türkiye*

Highlights

- The paper focuses on classification with deep learning for fake and real bilingual speech classes.
- An ensemble classification pipeline and logical rules infrastructure are combined in the study.
- A highly precise and more efficient classification accuracy were obtained.

Article Info

Received: 08 Sep 2023
Accepted: 12 Nov 2024

Keywords

Ensemble classifier
Machine learning
Deep learning
Speech recognition
Speech analysis

Abstract

The essential steps of our study are to quantify and classify the differences between real and fake speech signals. In this scope, the main aim is to use the salient feature learning ability of deep learning in our study. With the use of ensemble classification pipeline, the interpretable logical rules were used for generalized reasoning with the class activation maps to discriminate the different speech classes as correctly. Fake audio samples were generated by using Deep Convolutional Generative Adversarial Neural Network. Our experiments were conducted on three different language dataset such as Turkish, English languages and Bilingual. As a result of higher classification and recognition accuracy with the use of classification pipeline as compiled into a majority voting-based ensemble classifier, the experimental results were obtained for each individual language performance approximately as 90% for training and as 80.33% for testing stages for pipeline, and it reached as 73% for majority voting results considered together with the appropriate test cases as well. To extract semantically rich rules, an interpretable logical rules infrastructure was used to infer the correct fake speech from class activations of deep learning's generative model. Discussion and conclusion based on scientific findings are included in our study.

1. INTRODUCTION

Nowadays, speech processing is an essential research area for multimedia information retrieval. Fake speech recognition is a difficult task for both humans and machines, as it is usually due to the lack of understandable enough information obtained from the audio. Speech or audio forgery is an important issue that original audio signal can be easily changed by third party applications within different domains. Unless precautions are taken, these kinds of changes will raise some critical problems on the authentication (i.e., audio integrity verification) and forensics detection (i.e., forgery detection and localization) processes. These problems are caused by the audio splicing, copy-move forgery and insertion of another audio segment into the current audio signal, as well [1]. It is mandatory that maximum accuracy with minimum error in such a detection task is required to process non-stationary speech/audio signals which are needed for extracting salient temporal and auditory features to emphasize the dominant changes in a given stream of audio [2].

In the scope of signal analysis, the formant of a speech/voice is the resonant frequencies of a given filter. As a distinctive speech/voice fingerprint, a spectrogram is a basic time-frequency representation form which has a visualization of each formant value [3]. Mel is a unit of pitch such that equal distances in pitch

*Corresponding author, e-mail: bahadirkarasulu@comu.edu.tr

sounded equally distant to the listener. Mel-scale spectrogram is based on the spectrogram with frequencies which is calculated in Mel [4]. Every equal distance between these frequencies on the Mel-scale can be interpreted as they sound similarly different. The spectrogram is equipped with a special pattern or an energy level that varies at different times in the pronunciation of words, so that it can be analyzed and used to identify a speaking person's speech/voice [5]. A Mel-scale spectrogram represents values of the short-time Fourier transform (STFT) corresponding to the relevant part (i.e., window) of speech/audio sample [6]. In the manner of automatic feature learning capability of deep learning, feature extraction and purifying are made by given architectural model of deep infrastructure. In our experiments, deep layers are used to enrich the semantical meaning of given raw data. In the opposite side, the classical machine learning algorithms have been used in fake voice recognition and classification, to show the superiority of deep learning techniques in the manner of feature quality improvement by representation learning. In our study, the aim of the use of Mel-scale spectrograms is able to show signal characteristics for discriminating the different speech classes (i.e., real or fake) as correctly with feature learning (i.e., representation learning) ability of deep learning, and also, it was used with the class activation maps for generalized reasoning.

Deep learning is a popular research area within machine learning scope which automatically learns semantically rich information from data to bridge the semantic gap between low level and high level features. One of the main differences between classical machine learning and deep learning is the abstraction of features in deep learning instead of manually created (i.e. handcrafted) features for classical machine learning. Therefore, as the number of layers increases, the depth increases and a more refined set of features is created through filtering, and representation learning is performed [7]. For deep learning, some models in the literature have unsupervised strategies for mapping low-dimensional latent vectors to high-dimensional data [8]. In our study, Deep Convolutional Generative Adversarial Neural Network (DCGAN) [9] was chosen to extract salient temporal features from Mel-scale spectrogram images with its convolutional layers, and so, it was possible compiling the fake speech samples with the use of the generator and discriminator sub-models of this DCGAN network. Its class activation maps from appropriate layers were used in the interpretable logical rule based generalized reasoning. At a glance, Generative Adversarial Networks (GAN) has two sub-models to challenge the learning problem, such as DCGAN [9]. In a adversarial manner, these two sub-models are trained together in a zero-sum game that the first model (i.e., generator model) is trained to generate new samples, and also, the second model (i.e., discriminator model) is trained to classify samples as either real one from input domain or fake one (i.e., generated) [8].

In the literature, there are a bunch of studies to synthesize synthetic speech/voice with the use of GANs. For considering the synthesizing the raw speech/audio, ciwGAN (Categorical InfoWaveGAN) and fiwGAN (Featural InfoWaveGAN) [10], the WaveGAN [11], InfoGAN's Q-network [12] models are used as contradicted to two-dimensional visual data in DCGAN, the main difference between DCGAN and WaveGAN is the ability to construct an infrastructure where a one-dimensional vector corresponding to the time series data is produced as the raw acoustic output by the generator, and thus one-dimensional acoustic data is treated as the input to the discriminator. In addition, MelGAN [13] is a non-autoregressive feed-forward convolutional architectural model. The HiFi-GAN [14] consists of one generator and two discriminators.

The main goal of this paper is to quantify and classify the differences between real and fake speech signal (i.e., audio samples) classes as given as Mel-scale spectrograms with the help of an ensemble classification pipeline based on the deep learning feature learning. The proposed classification pipeline covers two main steps. The first step is the feature extraction. It transforms the audio of the speech sample into an image using a Mel-scale spectrogram, and the second step is representation learning using this classification pipeline. In this scope, the pipeline is enriched with meaningful and salient features to improve the discriminative power of ensemble classifiers for real and fake classes that the classifier is semantically rich. In addition, the use of an interpretable logical rule based reasoning for classification scoping was also chosen to generalize and improve the recognition ability based on bilingual speech/audio samples.

The main contribution of the study is to be able to recognise the fake speech as spoken content with the conversation, regardless of which language it is in such a bilingual form (i.e. Turkish or English). To prove the main goal of our study, we used deep learning-based features to feed an ensemble classification pipeline

and a mining process of interpretable logical rules for generalized classification using deep learning's generative model's class activations. As a first contribution, our experimental results with higher classification and recognition accuracy were obtained for each individual language performance approximately as 90% for training and as 80.33% for testing stages with classification pipeline as compiled into a majority voting-based ensemble classifier, and it reached as 73% for majority voting results considered together with the appropriate test cases as well. There is a difference between the average training and test accuracies that the main reason of this difference in which there are so many different classifiers and datasets used in our experiments. This difference is not so high because of morphological infrastructural differences between single language data and bilingual data. The model reliability is shown with the known metrics from literature such as Precision, Recall, Accuracy, *F*1 score and etc. This classifier pipeline was built on the *k*-Nearest Neighbor Classifier (*k*-NN) [15], support vector classifier (SVC) [16], Gaussian process classifier [17] and decision tree (DT) classifier [18] components. In addition, the second main contribution is to be able to construct an interpretable logical rules infrastructure to extract meaningful and semantically rich rules for a case-based reasoning system which infer the correct fake speech from class activations of deep learning's generative model. It has also achieved a combined best average accuracy of 73% for bilingual fake speech recognition in the test phase of our case-based reasoning infrastructure.

The rest of the paper is organized as follows. In section 2, a concise literature review and the comparison of related studies with the use of a short table are given in details. In section 3, methods, materials and infrastructure used in classification processes within our approach for fake speech recognition are introduced. In addition, a mining process of interpretable logical rules for classification scope were investigated. In section 4, the experiments conducted with English, Turkish and bilingual speech samples are explained with their details as related case studies. Furthermore, a discussion about these empirical results from experiments are given. Section 5 concluded the scientific findings from experiments, performance results, and discussed further opinions about the study.

2. LITERATURE REVIEW

In the literature, there are some studies about the use of GANs. The ciwGAN and fiwGAN [10] are two neural network architectural models that proposed in the related study have substantially more reduced latent representations (from 5 to 13 variables total). In the study [10], the models' output are unique lexical items for each unique code and reach accuracy that ranges from 98% to 26% in the five-word model. In addition, the WaveGAN [11] architecture was modified by Beguš [10], and also, InfoGAN's Q-network [12] was added onto it. It is used to computationally simulate lexical learning from raw acoustic data. As contradicted to two-dimensional visual data in DCGAN, the main difference between DCGAN and WaveGAN is able to construct an infrastructure that a one-dimensional vector corresponding to time series data as raw acoustic output is produced by the Generator, and hence, one-dimensional acoustic data is treated as Discriminator's input. The WaveGAN model achieves 93% accuracy on the test set. The Q-network is trained on retrieving information from the Generator's output with ciwGAN model. In addition, the categorical variables or features are determined to retrieve information by the use of the Q-network in fiwGAN model [10]. The InfoGAN achieves 5% error rate in classifying MNIST digits by matching each category in the experiment of its study.

MelGAN [13] is a non-autoregressive feed-forward convolutional architectural model. In a GAN setup, it is used to achieve audio waveform generation without additional distillation or perceptual loss functions. In the MelGAN architectural model, a stack of transposed convolutional layers was used to upsample the input sequence. Using the layered structure, each of the transposed convolutional layers was followed by a stack of residual blocks with dilated convolutions. In addition, MelGAN's window-based discriminator's learning style is constructed to classify between distributions of small audio chunks. The MelGAN model trained on LJ Speech Dataset [13] and achieved Mean Opinion Score (MOS) metric as 3.61 with the confidence interval of 95%. The HiFi-GAN [14] consists of one generator and two discriminators. These discriminators are used as multi-scale and multi-period discriminators. Therefore, HiFi-GAN generates synthesized speech from Mel-scale spectrograms. The generator uses transposed convolutions for upsampling of Mel-scale spectrograms to generate speech/audio samples. The HiFi-GAN model trained on LJ Speech Dataset and achieved MOS metric as 4.36 with the confidence interval of 95%. These studies

used some different datasets and different performance metrics to evaluate their experimental results in the scope. Table 1 is given for a fair comparison of these studies in the literature in manner of overview of the state-of-the-art. For more information about used datasets in these studies, one can refer to the related studies' citing references in details as given as below.

Table 1. *Overview of the related works*

Reference Study with Its Model Name	Highlights of Reference Study	Dataset	Performance Result
ciwGAN and fiwGAN [10]	More reduced latent representations, feature learning	TIMIT	Accuracy that ranges from 98% to 26%
WaveGAN [11]	With the same number of parameters and numerical operations of DCGAN as its two-dimensional analog	Speech Commands Zero Through Nine (SC09)	93% accuracy
InfoGAN [12]	InfoGAN learns interpretable representations	MNIST	5% error rate
MelGAN [13]	Achieve audio waveform generation without additional distillation or perceptual loss functions	LJ Speech Dataset	MOS metric as 3.61 with the confidence interval of 95%
HiFi-GAN [14]	Multi-scale and multi-period discriminators and using transposed convolutions for upsampling of Mel-scale spectrograms	LJ Speech Dataset	MOS metric as 4.36 with the confidence interval of 95%

3. MATERIAL METHOD

In this section, the methods used in our classification infrastructure to recognize fake audio parts of given speech are introduced with their details. Audio processing, related audio feature extraction process and Mel-scale spectrogram presentation of audio are given in detail. In our classification, a deep learning based data generation process was used to create a Mel-scale spectrogram of fake audio samples. In addition, deep neural network's (Deep Convolutional Generative Adversarial Neural Network, DCGAN) trained layers were used to construct extracted features for further machine learning based ensemble classification. These deep neural network layers' activations were presented with the use of Gradient Class Activation Maps (Grad-CAM) as heat maps as well. At the end, for a fair generalization of the bilingual fake speech recognition process, these heat maps used in a mining process of interpretable logical rules for classification scope were investigated to extract appropriate rules. In the manner of binary classification, the examined infrastructure with these rules was used to classify English, Turkish (i.e., monolingual) and Bilingual (i.e., English and Turkish together) audio samples (i.e., real or fake speech classes). In this section, these methods and related infrastructure are given in detail.

3.1. Audio Processing and Features

For audio event detection and voice/speech recognition, the audio features obtained from the audio samples assigned to the same class are correlated with each other. The audio analysis is based on obtaining audio features from the audio signal to represent the audio. Audio features are frequently used in studies as vector data type. These features include signal energy, frequency distribution which are given as numerical values that show the change of the signal over time [19]. The frequency domain or time-frequency representation of the signal energy is important. Most features are created by dividing the audio signal into frames, windows and performing spectrum analysis or obtaining statistical information of the signal by looking at

the size of the frequency components as well. The process of framing is such that the audio signal changes into fixed length frames with a certain time step (e.g., 50 milliseconds) [19, 20]. The process of windowing refers to separating the input audio signal into temporal audio segments. As incongruent to the real-world audio signal, obtained borders of these segments are then visible as discontinuities in signal. The windowing functions are smooth functions which go to zero at the borders, and thus, the discontinuity at the border becomes invisible with an effect as minimum statistical change on the signal [21]. In this way, spectrum analysis can be done more accurately. A widely used audio data representation format in audio analysis is the spectrogram. The spectrogram of an audio signal is defined as a feature matrix in the time-frequency domain. In the frequency domain, this matrix is constructed by joining together the feature vectors end-to-end for consecutive time frames in an audio recording [22].

In our study, in order to increase data diversity for gaining fake audios, there are a bunch of audio pre-processing methods that are applied such as pitch shifting [23], audio stretching [24] and audio trimming [25]. Librosa [26] audio processing library was used in order to apply these methods to audio samples given in our study. For having used mono audios, sampling rate is chosen as 8000 Hz for applying these audio enhancement methods.

Pitch shifting method helps to modify the pitch of an audio with the given step number without modifying its speed. In addition, the pitch shift involves an STFT [6], shift-usually of an importance spectrum along the frequency axis, after which sign reconstruction using the algorithm of Griffin and Lim [27]. While moving the magnitude spectrum by ignoring the phase, In their study, Griffin and Lim tried to find an affordable strategy to find the right section while reconstructing the time domain sign. Furthermore, pitch shifting was applied in four half steps in our experiments as well.

In our study, audio stretching [24] was used to modify the duration of that audio without modifying its frequencies. Three different time rates are used to enhance which equals 0.25, 0.50 and 1.25 value to have slower and faster audios. It basically transforms the signal using STFT [6], stretches it using Phase Vocoder [28] and reconstructs it to an audio signal using inverse STFT. Audio trimming [25] was used to cut the silent parts of an audio signal given in our study. It basically just eliminates the low-level sounds in an audio range. After abovementioned audio preprocessing stages, some well-known audio features are extracted to obtain useful information about audio segments. Such details are given below.

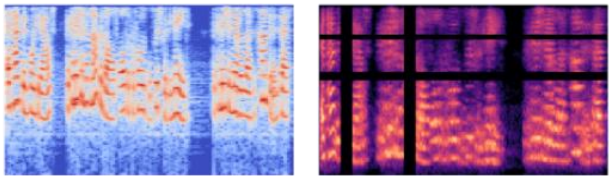
The human ear is perceptually more sensitive to low frequency variations of the signal. The Mel scale is a nonlinear frequency scale. In accordance with this scale, the mel-spectrogram consists of vectors of mel band energy features as a matrix [22]. Mel filter bank expands bandwidth with center frequencies for various filters which is a bank with triangular shaped bands arranged on the mel frequency scale.

The Librosa [26] audio and music processing library is used in our study to get spectrograms with audio samples with a frequency of 8000 Hz (i.e., mono) and using their log-mel features. It is applied to all of the audio files in a given dataset which are original samplings and fake samplings such as pitch shifted audios, time stretched audios and trimmed audios. In order to increase the diversity for fake data, the Multiple Masking [29] method is applied to augment the data to get mixed data. In this way, this method is used to apply frequency and time masks from random parts in both time and frequency zones by chosen number of masks and consecutive time frames and frequency bins. In our study, the parameter of time frames is especially chosen as 24 and the parameter of frequency bin is chosen as 36 which means this randomly selected part will be between 24 frames maximum for time frames and 36 bins maximum for frequency bins. In other words, it creates silent parts in spectrograms to increase the diversity [30]. This method is only used in stretched audios because of having extended times. In addition, in our study, fake audio samples are generated using the same approach used in the Multiple Masking. Total three different parts are randomly selected in the frequency zone and cropped from this spectrogram to be added to the end or start of that spectrogram. To increase the diversity with cropping more seconds from the audio portion, the parameter of frequency bins is chosen as 150 and Gaussian noise (background noise) [31] is applied to that randomly cropped part. The Gaussian noise method is used for adding a random signal as white noise to the background of an audio which is known as sizzle. After applying this method, new spectrograms are gained from fake audio samples and these fake audio samples are reconstructed as an audio file. In order to

increase the diversity for real data, Mel-frequency cepstral coefficient (MFCC) [19] and Chroma Energy Normalized Statistics (CENS) type extraction methods [32] are applied. MFCC features extraction is one of the most popular methods used to process and to augment audio data which have coefficients that are less-affected by changes. These coefficients are a set of Discrete Cosine Transform (DCT) decorrelated parameters. Through an appropriate transformation, they are computed based on the logarithmically compressed filter-output energies. To process the Discrete Fourier Transformed (DFT) speech signal, it uses a perceptually spaced triangular filter bank [33].

Chroma Energy Normalized Statistics (CENS) feature extraction method is often used to perform audio matching and similarity [34]. It basically smooths the local deviations in an audio such as tempo, pace and so on. In our experiments, in order to achieve a balanced data of real samplings, which consist of original samplings, obtained by applying MFCC and Chroma CENS, related data are splitted by 80% train and 20% test samplings for DCGAN [9] training in our study as well.

In Figure 1, a raw audio sample [35] (a) Mel-scale spectrogram, and (b) its pre-processed audio sample by applying stretching by the rate as 1.25 value, and applying multiple masking Mel-scale spectrograms are shown.



(a) (b)

Figure 1. *Mel-scale spectrograms; a) Raw audio sample, b) Pre-processed audio sample*

3.2. Deep Convolutional Generative Adversarial Neural Network

In our study as deep learning libraries, the Tensorflow backend and Keras frontend [36, 37] are used to build a DCGAN model as our deep learning based automatic feature learning infrastructure. As mentioned before, two different sub-models are generated for DCGAN as usually with the use of Keras as well. In this infrastructure, the generator is used to generate a random spectrogram from the given input, and a discriminator is used to classify the generated spectrogram as well.

If we take a closer look at our automatic feature learning infrastructure, for the generator sub-model, the Input layer takes spectrograms as inputs and gives them into a Dense layer with 8192 neural nodes. Then, a Reshape layer takes place to reshape the inputs to make the next layers suitable. Three Transposed Convolutional layers (i.e., Deconv) are used with their filter counts equal to 128, 256, and 512 value, respectively. In addition, the LeakyReLU (Leaky Rectified Linear Unit) [38] activation function is used after every Transposed Convolutional layer. Leaky version of a Rectified Linear Unit (LeakyReLU) [39] is used as an activation function for mostly hidden layers which returns the value itself if the value is positive which corresponds to a flat slope while it determines the value with a small slope for negative values. The coefficient of this slope which helps to determine the function result for negative values is given before the training process. And last, three filters involved by a Convolutional layer [40] are added.

For the discriminator sub-model, an Input layer with the shape of the output of the generator model is used. Three Convolutional layers (i.e., Conv) with their filter counts are equal to 64, 128 and 128, respectively. The LeakyReLU as activation function is used. A Flatten layer is used to get a one dimensional (1D) vector and a Dropout [39] process is applied as a layer to prevent possible over-fitting problems. Finally, a Dense layer with one neural node is used to classify the input as "*Fake*" or "*Real*" with Sigmoid [41] activation function which is used mostly for the output layer to classify the result with a threshold value.

In our study, for a fair generalization, the DCGAN [9] model is trained for 100 epochs with the generation 10 images per epoch for Turkish data while 52 images are generated per epoch for English. The Adaptive Moment Estimation (ADAM) [42] optimization method is used for training. It helps to keep an

exponentially decreased average past gradient to the similar value to compute adaptive learning rate. In this way, it is ensured that the learning rate can be determined effectively and adaptively by estimation based on the first and second moments of the gradients. Learning rate is initially chosen as $1*10^{-5}$ for both sub-models. Since having a binary classification model for "*Real*" or "*Fake*" classes, the Binary Cross-Entropy loss function is used. It helps to determine how close the predicted value is to the actual value for penalizing. In other words, it means how successful the model is to classify the given audio sample as given as a Mel-scale spectrogram.

In Figure 2, a DCGAN architectural model [43] is used in our study as given as a block diagram schematic in a layered manner. LeakyReLU is shown as 1Relu in Figure 2.

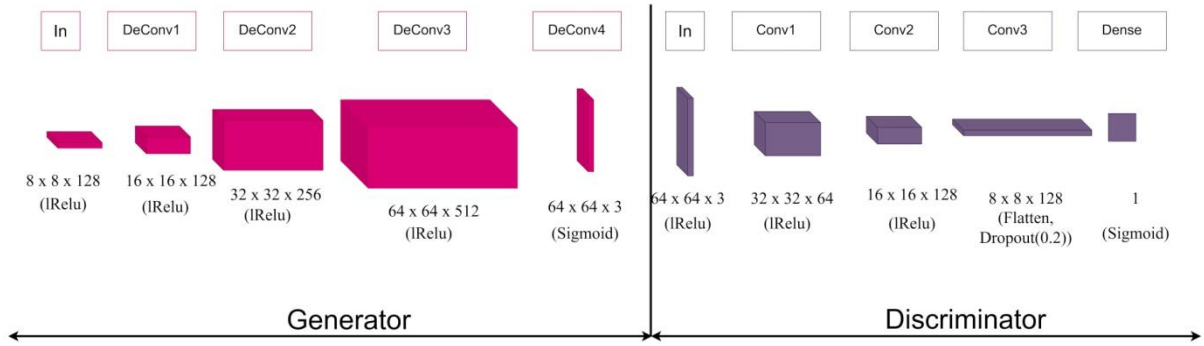


Figure 2. *The DCGAN architectural model used in our study as its mentioned form in [43]*

The generator takes the image, learns from its features by using a transposed convolution layer. At the end, with the last convolutional layer, a new tensor is generated to be given to the discriminator for classification. This tensor is denoted to the image that is generated by the generator and in the training process, the generator learns to make the discriminator classify this output correctly. Then, the discriminator takes that generated tensor as an input to process and classify it. In the dense layer which is at the end of the discriminator, the classification process is performed with the sigmoid activation function. This whole process is performed inside a loop and with this way, the feature learning process is being performed. These generated images are used as "*Fake*" images for the pipeline which is created by machine learning algorithms to ensemble.

Considering the weighted spatial map, the Grad-CAM [44] method is weighted with gradient (slope) values according to the class of the relevant color channel. The intensity map formed here shows in which parts of the image the activity is concentrated. In the literature, this density map is usually given in the form of a heatmap. Afterwards, the relevant heatmap is superimposed over the input image. In this way, the main idea behind the Grad-CAM process is to detect the important points that are used by the model to predict. In other words, there is a high activation in these points. These points are denoted by yellow-red color and the less-activation points are denoted with blue color in Mel-scale spectrogram images given in our experiments. This method does not directly affect the classification process, so these heat maps used in a mining process of interpretable logical rules for classification scope were investigated to extract appropriate rules by its statistical information created in the Red, Green, and Blue (RGB) colorspace.

In our study, the discriminator sub-model is used for this Grad-CAM process. The last Convolutional layer before the classification is selected. Because it basically does a prediction operation using the last layer which as the discriminator model does. But the difference is while the discriminator model is doing the prediction for classification, the Grad-CAM process is used for predicting the activation intensity in pixels to colorize them.

With the help of Tensorflow library [33], gradients per every pixel for an input image is calculated. Then, the average of the gradient values are calculated and used as weights. The ponderation of the filters with respect to the weights is calculated. Values of obtained heatmap are normalized between [0,1] and then scaled to the range as [0,255]. After these processes, the found array is converted into an image. These images and found arrays are used for extracting rules for classification by using extracted features from

found high and less activation points. Minimum and maximum values from the found array for detecting the colors of the image by proximity to white and black, standard deviation of found array for seeing the spread of data, white pixel rate of an image, shannon entropy and Gray Level Co-Occurrence Matrix (GLCM) [45] are those features. Shannon Entropy [46] is an average rate of the information. This kind of information is generated from a stochastic data source. GLCM is a matrix defined on an image as a distribution of co-occurring pixel values at a given offset. Several statistics can be derived from GLCM. The Homogeneity Statistic measures a value that indicates how closely the GLCM's element distribution resembles its diagonal. The Energy Statistic gives the sum of squared elements in GLCM. The Correlation Statistic helps to measure the joint probability occurrence of the specified pixel pairs. The Dissimilarity Statistic measures the local variation. All of these 4 statistics are used to derive the information from GLCM.

3.3. Classification Boosting with the Classifier Pipeline

For classification boosting, a pipeline can be useful to construct one compact model to chain multiple estimators into this one. When a fixed sequence of steps in processing the data, e.g., feature learning before a more complicated classification process, it can be more preferential. In our study, this kind of classifier pipeline is built with the scikit-learn [47] library. In order to apply more accurate classification, our classifier pipeline is assembled with the *k*-Nearest Neighbor Classifier (*k*-NN) [15, 48], support vector classifier (SVC) [16], Gaussian process classifier [17] and decision tree (DT) classifier [18] components. The details of these components and their usage in our study are declared in the following parts as well.

In classical machine learning, *k*-Nearest Neighbor Classifier (*k*-NN) [48] is a non-parametric, supervised learning-based classification algorithm which is used for classifications or predictions of individual data points' grouping. Basically, it uses the majority vote term to classify the points with their similarities. Each point is a member of a class which has *k* members the closest. This distance between the points is calculated by a distance algorithm. Generally, preferred distance metric for this algorithm with the continuous variables is Euclidean distance. Since there are binary classification as usually as in a manner of True or False decision, the *k* value is chosen as two. Two phases are usually used in machine learning, which are the training phase and the testing phase. At the end of the training phase of our study with related datasets (i.e., their details are given in related following parts), the accuracy [49] is equal to approximately 0.92 and 0.92 for both English and Turkish languages' audio samples, respectively. In addition, the accuracy for the testing phase is equal to approximately 0.86 and 0.81 for both English and Turkish languages' audio samples, respectively.

A support vector classifier (SVC) [16] is based on the support vector machine (SVM) [50], which is another classical machine learning algorithm which can be used for both regression and classification problems. The main idea behind using this algorithm as a classification model is to separate all the classes with hyperplanes where these hyperplanes are drawn between the closest data points to find maximum range of margin. Related data points may have classified inside this margin which can be said they are outliers. In the SVC model, the radial basis function (RBF) [51] is chosen as a kernel function for having a non-linear distribution. All of the parameters are chosen as default in which "*C*" value, regularization parameter, equals to 1.0, as well as, gamma value, kernel coefficient for RBF, equals "*scale*" value.

As mentioned above, two phases are usually used in machine learning as called the training phase and the testing phase. At the end of the training phase of our study with related datasets (i.e., their details are given in related following parts), the accuracy is approximately equal to 0.88 and 0.87 for both English and Turkish languages' audio samples, respectively. In addition, the accuracy for the testing phase is approximately equal to approximately 0.87 and 0.86 for both English and Turkish languages' audio samples, respectively.

Gaussian process classifier [17, 52] also can be used as both regression and classification algorithms. It can be said that it is a generalization of the Gaussian probability distribution [17]. It is used as the foundation for advanced non-parametric machine learning algorithms. Gaussian process classifier's kernel hyperparameters are optimized during fitting in training so, all of the parameters were given as default in

our study. In default parameters, kernel value, covariance function of the Gaussian Process, equals "*None*" value which equals the Radial Basis Function (RBF). As an optimizer, "*fmin_l_bfgs_b*" is chosen as default which is used to minimize the function by using the Limited-Memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS-B) algorithm which is developed for using a limited amount of memory [53]. At the end of the training phase of our study with related datasets (i.e., their details are given in related following parts), the accuracy is approximately equal to 0.89 and 0.87 for both English and Turkish languages' audio samples, respectively. In addition, the accuracy for the testing phase is approximately equal to 0.88 and 0.86 for both English and Turkish languages' audio samples, respectively.

The decision tree (DT) [54] classifier is another algorithm that can be used for both regression and classification. It is a tree model in which internal nodes specify attribute tests, branches specify result of the test and last, leaf nodes specify the class label. The main idea of this algorithm is to split data with many records into small clusters by applying decision rules. Decision trees criterions decide to split a node into two or more child nodes. In our study, Gini algorithm [55] is used as the criterion. For the Decision Tree (DT) algorithm, "*random_state*" value is chosen as "*zero*" for having the same random tree structure in every run in the experiment. Other parameters are chosen as their default values. "*Criterion*" term for measuring the quality of the split, equals the "*gini*" function. In addition, "*max_depth*" term's value for specifying the maximum depth of the tree is chosen as "*None*", where this "*none*" means nodes will be expanded until every one of the leaves are pure.

The voting classifier (VC) [56] is another machine learning model. It is used to train a model on a given ensemble of numerous models for ensemble classification. With giving these models, it predicts an output based on the highest probability chosen. The hard voting method is used for prediction. It means predicted class will be a category with the highest majority of votes. The *k*-NN [15], SVC [16] and Gaussian process classifier [17] are given as the ensemble of numerous models. Using these models, the VC model is trained. As mentioned above, two phases are usually used in machine learning as called the training phase and the testing phase. At the end of the training phase of our study with related datasets (i.e., their details are given in related following parts), the accuracy is approximately equal to 0.89 and 0.88 for both English and Turkish languages' audio samples, respectively. In addition, the accuracy for the testing phase is approximately equal to 0.88 and 0.87 for both English and Turkish languages' audio samples, respectively.

The pipeline tool that was provided by the scikit-learn [47] library is used to avoid preprocessing and modeling steps into a single step to use all the models together. In other words, different machine learning models are assembled as steps which can be cross validated together. In our study, the VC model as a data preprocessor and the DT [18] model as a transformer are used as these steps. Transformer reacts as a predictor in this case. In other words, for having a strong rule based infrastructure, the DT is chosen as a predictor of given data which come from the VC. In addition, for choosing DT as a transformer and for it lasting long, the train process is not done for a only defined DT model. These rules from data are extracted to be used for the skope-rules [57] module for scoping target class.

At the end of the training phase of our study with related datasets (i.e., their details are given in related following parts), the accuracy is approximately equal to 0.92 and 0.92 for both English and Turkish languages' audio samples, respectively. In addition, the accuracy for the testing phase is approximately equal to 0.86 and 0.81 for both English and Turkish languages' audio samples, respectively. Classifier pipeline's performance results as accuracy values are shown in Table 2.

Table 2. *Performance results for classifier pipeline*

Classifier's Name	Performance for Training and Testing Stages based on the Turkish Language (Accuracy values)	Performance for Training and Testing Stages based on the English Language (Accuracy values)
k-Nearest Neighbor Classifier	Train: 0.92, Test: 0.81	Train: 0.92, Test: 0.86

Support Vector Classifier	Train: 0.87, Test: 0.86	Train: 0.88, Test: 0.87
Gaussian Process Classifier	Train: 0.87, Test: 0.86	Train: 0.89, Test: 0.88
Voting Classifier	Train: 0.88, Test: 0.87	Train: 0.89, Test: 0.88
Pipeline Classifier	Train: 0.92, Test: 0.81	Train: 0.92, Test: 0.86

3.4. Mining Interpretable Logical Rules for Classification Scoping using Skope-Rules

For mining interpretable logical rules for classification scoping in our study, we used a machine learning module of Python language called as "*skope-rules*" [57] which aims at learning logical, interpretable rules for scoping a target class (i.e. detecting with high precision instances of the class). It is built on top of scikit-learn [47] library. With the help of the skope-rules module, usable rules are extracted from the tree ensemble. Its advantage is based on the existing fast algorithms (such as bagged decision trees, or gradient boosting), which is used to produce such tree ensembles [58]. The duplicated or similar rules are then removed by a related process of the skope-rules module. This process is based on a similarity threshold of their rule supports. The main goal of the skope-rules module is to provide rules verifying precision and recall conditions to select best rules. It still implements a score (i.e., a decision function) method, but which does not solve the $L1$-regularized optimization problem as in [59]. Instead, weights are simply proportional to the out of the bag associated precision of the rule.

In our study, a total of 7 different rule-based models are obtained for Turkish, English and Bilingual which have 2 to 8 rules for each language. Two and more best rules are extracted by the skope-rules module with the use of English and Turkish languages' data. The reason for obtaining 2 to 8 rules is for benchmarking. In other words, performance evaluation is being done for selecting the best performed rule. In Tables 3 and 4, there are shown some examples of selected skope-rules module parameters and its extracted rules, respectively.

Table 3. *Selected Skope-Rules Parameters*

Parameter Name	Value	Function
random_state	42	the value used to have an experimental design with randomized initial condition as a shuffled trial for every run in the experiment
n_estimators	10	the number of rules for prediction
recall_min	0.01	the minimal recall for a rule
precision_min	0.01	the minimal precision for a rule
max_depth_duplication	3	the decision tree's maximum depth for rule deduplication
max_depth	3	the maximum depth
max_features	0.5	the ratio of feature number for decision tree to consider
max_samples_features	0.5	the ratio of number of feature for training used to draw each decision tree

As a result, according to the table of extracted rules (i.e.,Table 4), for example by looking to English results, it can be said that, if entropy value of the spectrogram which is obtained from an audio is less than or equal to 6.370113372802 or mean value is higher than 163.1597900390625 or standard deviation (i.e., std. dev.) value less than or equal to 84.57405090332031, it can be classified as "*Fake*" class. Otherwise, it can be classified as the "*Real*" class. In addition, some descriptive statistics for the features of Test Case 1 (i.e., left hand side of the table) and Test Case 2 (i.e., right hand side of the table) are shown in the Table 5. In the table, "*white_pix*" feature indicates the ratio of white colored pixels in given spectrogram image which are considered as bias, and also, obtained values of this feature are relatively very low.

Table 4. *Some Examples for Extracted Rules*

Rule	Rule Count	Language
Rule#1: entropy <= 6.37 *Rule#2:* mean > 163.15 *Rule#3:* std. dev. <= 84.57	3	English
Rule#1: std. dev. <= 69.52 *Rule#2:* glcm_dissimilarity <= 26.63 and std. dev. <= 78.53 *Rule#3:* glcm_energy > 0.03 and glcm_correlation > -0.32 and mean > 173.06 *Rule#4:* glcm_dissimilarity <= 34.07 and mean > 173.06	4	Turkish
Rule#1: glcm_dissimilarity <= 47.51 and mean > 169.17 *Rule#2:* entropy <= 7.02 and mean > 169.17 and std. dev. <= 83.04	2	Bilingual

Table 5. *Some descriptive statistics for the feature columns of Test Case 1 (left hand side) and Test Case 2 (right hand side)*

Test Case 1			Test Case 2		
Column	Mean	Standard deviation	Column	Mean	Standard deviation
min	0	0	*min*	0	0
max	255	0	*max*	255	0
mean	202.04	11.58	*mean*	202.56	11.33
standard deviation	56.37	7.44	*standard deviation*	55.88	7.55
entropy	5.90	0.67	*entropy*	5.90	0.67
white_pix	0.20	0.14	*white_pix*	0.20	0.14
glcm_homogeneity	0.08	0.05	*glcm_homogeneity*	0.08	0.05
glcm_dissimilarity	18.34	11.60	*glcm_dissimilarity*	18.13	11.27
glcm_energy	0.047	0.008	*glcm_energy*	0.047	0.008
glcm_correlation	0.53	0.19	*glcm_correlation*	0.53	0.19

4. EXPERIMENTAL RESULTS

In our experiments, audio sample files for English, Turkish languages and both of them as Bilingual, obtained by mixing Turkish and English, are taken from datasets. Some methods such as audio trimming, audio stretching etc. are used for data augmentation. Spectrograms are obtained from these audios. Data augmentation methods for augmenting image data such as MFCC, Chroma etc. are applied. Fake data is obtained from real audio files. Then, using all of this fake and real data, a DCGAN model is trained and fake audio samples that are generated by this type of GAN model are obtained. With the results of this DCGAN model and real data, a pipeline is generated using some machine learning algorithms and a prediction process is applied. Then, obtaining the heatmaps process is applied by using Grad-CAM. With these heatmaps, features such as white pixel rate, GLCM features etc. are obtained. Finally, with all of these prediction results and heatmap features, the skope-rules module's process is applied by different rule counts to find optimum rule count for the best performance.

In the experiments, Google Colab Pro [60] and a computer which is equipped with Intel Core i7-11800H CPU with 2.30 GHz and has 16 GB RAM memory, nVidia RTX 3050 Ti graphic card (GPU) with 8 GB RAM are used. These experiments are run several times with different experimental setups to get the best result. Python [61] v3.9 programming language is used for performing these experiments with its well-known libraries and frameworks. Keras [37] is an example of above mentioned libraries and one of the most-known and most-used libraries for deep learning research projects. It uses Tensorflow [36], a machine learning platform which is used as a backend. In our study, Keras version 2.6.0 is used to construct the main architectural model of the DCGAN variant in our system which is used as a frontend.

In the following subsections, datasets that are used for these training and performance evaluations with well-known metrics in detail are given for both Turkish and English languages, and also in a manner of Bilingual.

4.1. Datasets

In this study, two different language dataset is used for both languages, English and Turkish, respectively. Both datasets are obtained from a website called Open Speech And Language Resources (OpenSLR) [62] which is devoted to hosting speech and language resources. Some details about these datasets are given in the following parts. These datasets are publicly available at their respective owner's website as well.

The Turkish language is a member of Altaic Language Group's [63] Turkic [64] family. Vowel harmony and widespread agglutination are the Turkish language defining traits. Noun clauses and grammatical gender are nonexistent in Turkish. There are 21 consonants and 8 vowels in a total of 29 letters in the Turkish alphabet. In the scope of spoken coverage area, there are 90 million speakers of Turkish language known and 84 million people speak it as their native language [65]. Turkish is the official spoken language in Turkey. The MediaSpeech [66] dataset is used for the Turkish language dataset. The dataset consists of 4 different language data which are Spanish, Turkish, Arabic and French. There are 10 hours of speech for each language. In this dataset, the Turkish dataset consists of speeches that are taken from the news videos that are found on YouTube [67] and transcriptions of these speeches. Each speech is 14 seconds long and uncategorized. Transcript files of these speeches are all in text file format. The number of text files is as many as speech files.

The English language is a member of the Indo-European Language Family [68]. There are 21 consonants and 5 vowels in a total of 26 letters in the English alphabet. There are over one billion speakers of the English language known and over 379 million people speak as their native language [69]. English is an official language in 59 countries including the United States of America, England. Modern English is described as the first "*global lingua franca*" spoken by non-native speakers. The LibriSpeech [35] dataset is used for English language dataset. The dataset contains English speeches from read audiobooks from the LibriVox [70] project. There are approximately 1000 hours of speeches with 16 KHz frequency and transcriptions of these speeches. These files also have a transcript which is only one file and in Comma

Separated Value (CSV) format for portability, or, with the purpose of information interchange between end-users.

4.2. Objective Evaluation Using Performance Metrics

This section contains our objective performance evaluation methodology and related metrics that are used for evaluating our experiments in the study. For classifying the results by their classes as "*Fake*" or "*Real*", the confusion matrix and the performance metrics are used such as accuracy, recall and precision that are obtained from measurements from the information retrieval process. Confusion Matrix is given in the form of a matrix that can be considered as a statistical table that provides information about classification results. True Positive (*TP*), True Negative (*TN*), False Positive (*FP*) and False Negative (*FN*) are the information that is provided and used in this matrix. In this study, *TP* is the number of data that is labeled as "*Fake*" and predicted by the neural network model as the same whereas, *TN* is the number of data that is predicted by the neural network model as "*Real*". *FP* is the number of data that is labeled as "*Real*" and predicted by the neural network as the same whereas, *FN* is the number of data that is predicted by the neural network as "*Fake*".

The Accuracy metric is one of the metrics that can be found by getting the rate of positives in all data. It helps to detect how much data is predicted correctly. Not only the value can be shown as decimal but also, as a percentage value. In the Equation (1), the formula is shown [30] as given as below

$$Accuracy = (TP + TN)/(TP + FP + TN + FN)\,. \tag{1}$$

The Recall metric can be found by getting the rate of correctly predicted positives in all positive values. It helps to detect how much positive data is predicted correctly. In the Equation (2), the formula can be seen [30] as given as below

$$Recall = TP/(TP + FN)\,. \tag{2}$$

The Precision metric can be found by getting the rate of correctly predicted positives in all correctly predicted values. It helps to detect the rate of correctly predicted positive data among all data correctly predicted. In the Equation (3), the formula can be seen [30] as given as below

$$Precision = TP/(TP + FP)\,. \tag{3}$$

The *F*1 Score can be found by getting the harmonic mean of precision and recall. In order to take into account the severe circumstances in the experiment when performance is compromised, it provides a useful evaluation result. As shown in the Equation (4), the formula can be seen [30] as given as below

$$F1 = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}\,. \tag{4}$$

The Receiver Operating Characteristics (ROC) Curve [71] is a plot graph that is computed using True Positive Rate (*TPR*) versus False Positive Rate (*FPR*) metrics to show the alteration of Area Under Curve (AUC) [72] score for any class in the experiment. The higher value all the ROC plots reach towards the upper left corner, the better the model distinguishes between the classes. In other words, the *x* axis beckons the *TPR* which gives the rate of *TP*'s in whole data and the *y* axis beckons the *FPR* which gives the rate of *FP* in whole data. In this plot, reaching the top left corner means getting a high *TPR* and a high *TPR* means getting a quite good classification result.

The AUC score [72] is obtained by the area under the curve. It scales from [0,1] and the closer to 1, the higher the model is performed. The threshold for AUC to comment on the model's performance equals 0.5 value. AUC score below 0.5 means that the model is performing not so well. With the help of these metrics and measurements, the objective performance evaluation is done for our experimental results as well.

4.3. Case Study for Turkish Language Based on the Fake Audio Samples

Performance evaluation for fake audio samples for Turkish Language is performed by using the performance metrics that are mentioned in the previous section. Two different results are obtained for Turkish Language based data. The first result is obtained for the Pipeline results and the other one is obtained for skope-rules results. For Pipeline results, the accuracy that is obtained is equal to 0.80. The average precision, recall and *F*1 score is equal to 0.81, 0.80 and 0.80 respectively. For skope-rules [57] results, the accuracy that is obtained is equal to 0.98 for Test Case 1 that is created for "*Fake*" data predicted as correctly or in a wrong way and 0.99 for Test Case 2 that is created for "*Real*" data predicted as correctly or in a wrong way. The number of the rules is chosen as 3, which performs with the highest accuracy. In Figures 3 and 4, the classification report for both Pipeline and skope-rules results can be seen.

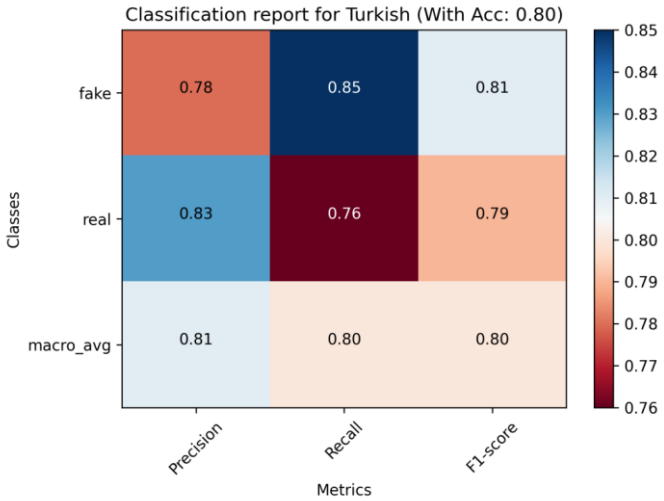
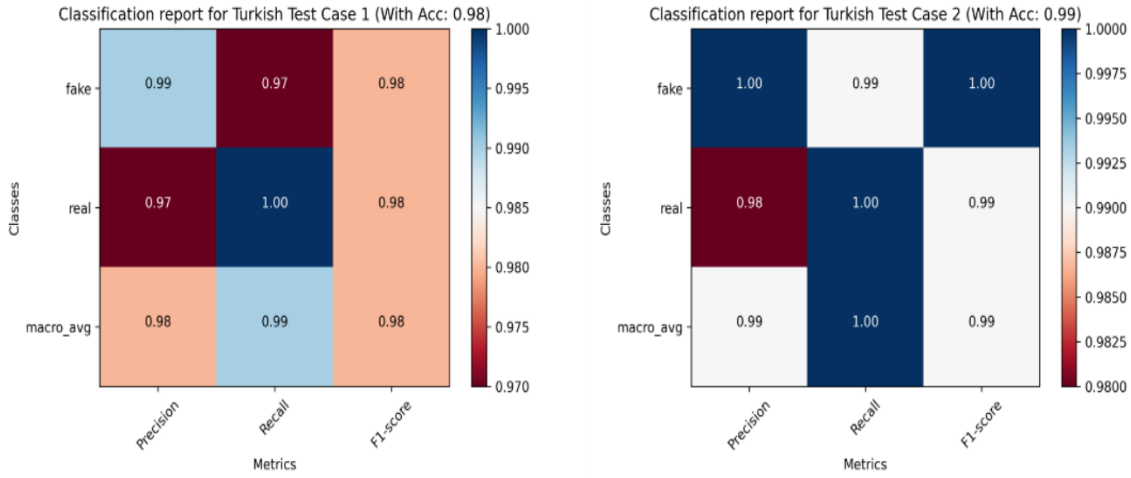


Figure 3. *Classification Report for Pipeline Results for Turkish Language*



(a) (b)

Figure 4. *Classification Report for Skope-Rules Results for Turkish Language; a) Test Case 1, b) Test Case 2*

4.4. Case Study for English Language Based on the Fake Audio Samples

Performance evaluation for fake audio samples for English Language is performed by using the performance metrics that are mentioned in the previous section. Two different results are obtained for English Language based data. The first result is obtained for Pipeline results and the other one is obtained

for skope-rules results. For Pipeline results, the accuracy that is obtained is equal to 0.86. The average precision, recall and *F*1 score is equal to 0.87, 0.86 and 0.86 respectively. For skope-rules results, the accuracy that is obtained is equal to 0.50 for Test Case 1 that is created for "*Fake*" data predicted as correctly or in a wrong way and 0.50 for Test Case 2 that is created for "*Real*" data predicted as correctly or in a wrong way. The number of the rules is chosen as 6, which performs with the highest accuracy. In Figures 5 and 6, the classification report for both Pipeline and skope-rules results can be seen.

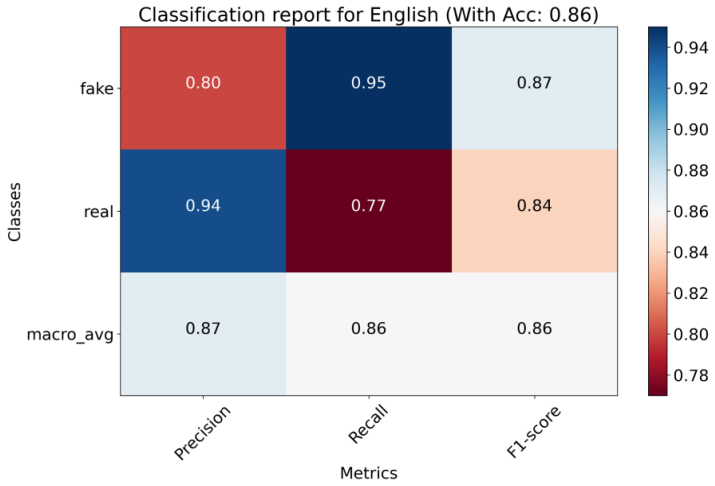


Figure 5. *Classification Report for Pipeline Results for English Language*

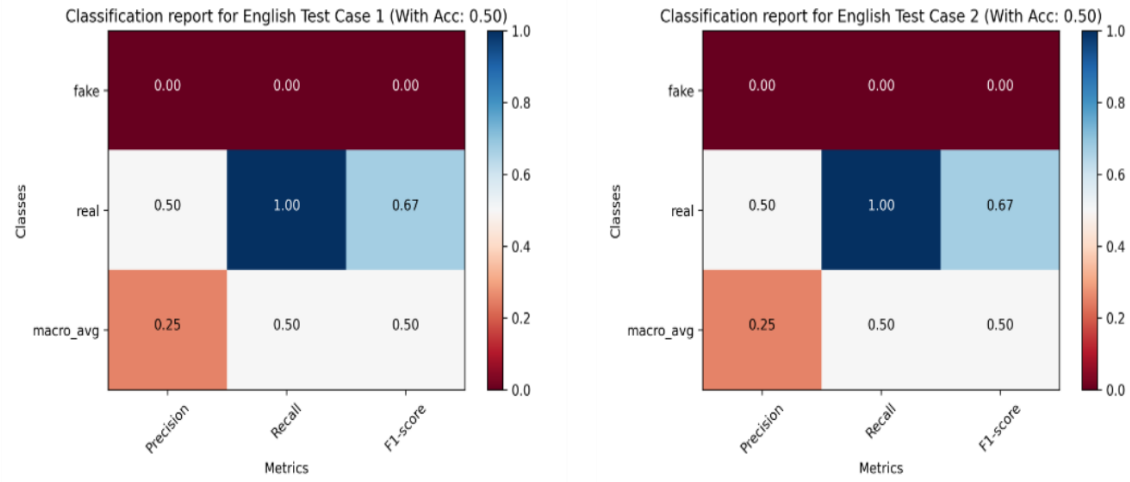


Figure 6. *Classification Report for Skope-Rules Results for English Language; a) Test Case 1, b) Test Case 2*

4.5. Case Study for Bilingual Language Based on the Fake Audio Samples

Bilingual Language is created by mixing Turkish and English Language data. Evaluation for fake audio samples for Bilingual Language is performed by using the performance metrics that are mentioned in the previous section. Two different results are obtained for Bilingual Language based data. The first result is obtained for Pipeline results and the other one is obtained for skope-rules results. For Pipeline results, the accuracy that is obtained is equal to 0.74. The average precision, recall and *F*1 score is equal to 0.76, 0.74 and 0.73 respectively. For skope-rules results, the accuracy that is obtained is equal to 0.47 for Test Case 1 that is created for "*Fake*" data predicted as correctly or in a wrong way and 0.94 for Test Case 2 that is created for "*Real*" data predicted as correctly or in a wrong way. The number of the rules is chosen as 6,

which performs with the highest accuracy. In Figures 7 and 8, the classification report for both Pipeline and skope-rules [57] results can be seen.

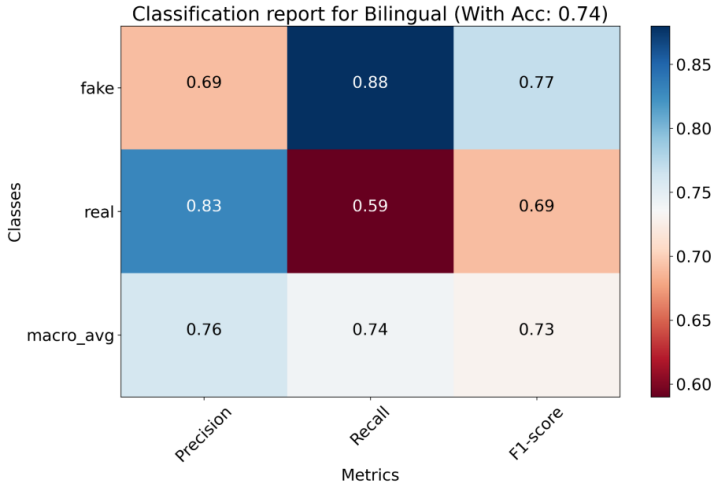


Figure 7. *Classification Report for Pipeline Results for Bilingual Language*

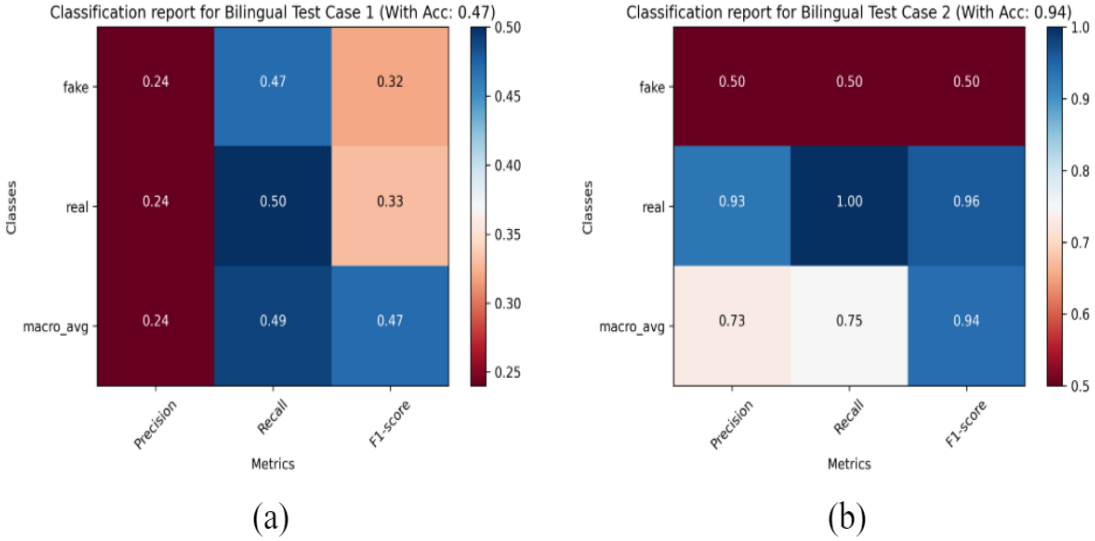


Figure 8. *Classification Report for Skope-Rules Results for Bilingual Language; a) Test Case 1, b) Test Case 2*

4.6. Evaluation of Case Study for All Languages Based on the Fake Audio Samples

This section involves comparison based on all three experimental set results (e.g., English, Turkish and Bilingual) obtained. These results are obtained by using the evaluation metrics that are mentioned in the previous sections. At first, to show the best performing rule number for skope-rules, the plot graph below in Figure 9 can be useful. Mean accuracy of rule numbers for all three languages and both Test Cases are shown in the plot graph with markers. In these figures (i.e., Figures 9, 10 and 11), the abbreviations are defined as *EN* for English, *TR* for Turkish language, and also as *BIL* for Bilingual languages, as well.

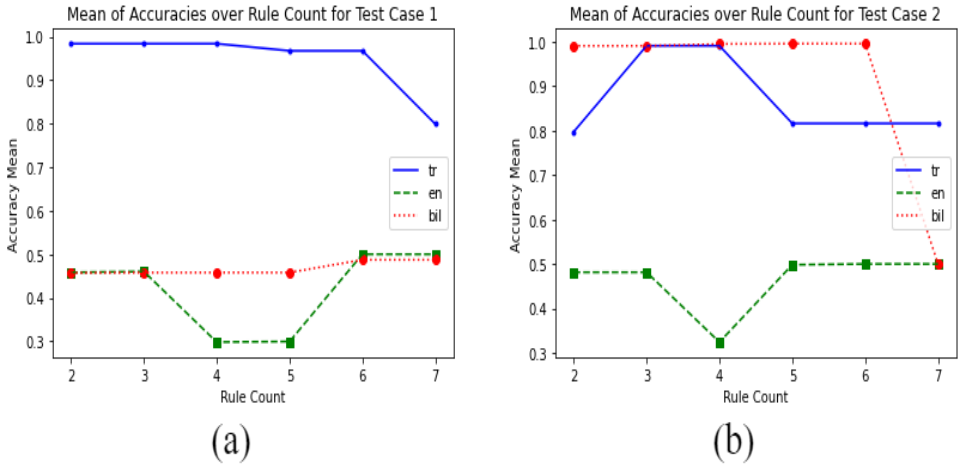


Figure 9. *Plot graph of Mean Accuracies to choose the best performing rule count; a) Test Case 1, b) Test Case 2*

To evaluate the performance of skope-rules and Pipeline, the ROC Curves are obtained. These curves also contain AUC scores for all languages in the experiments and all Test Cases for skope-rules. These AUC scores are obtained by getting the mean of the clusters for all Test Cases in Figures 10 and 11.

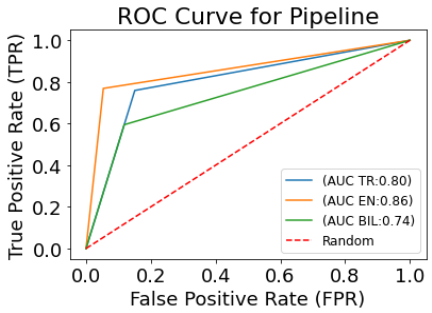


Figure 10. *ROC Curve for Pipeline Results*

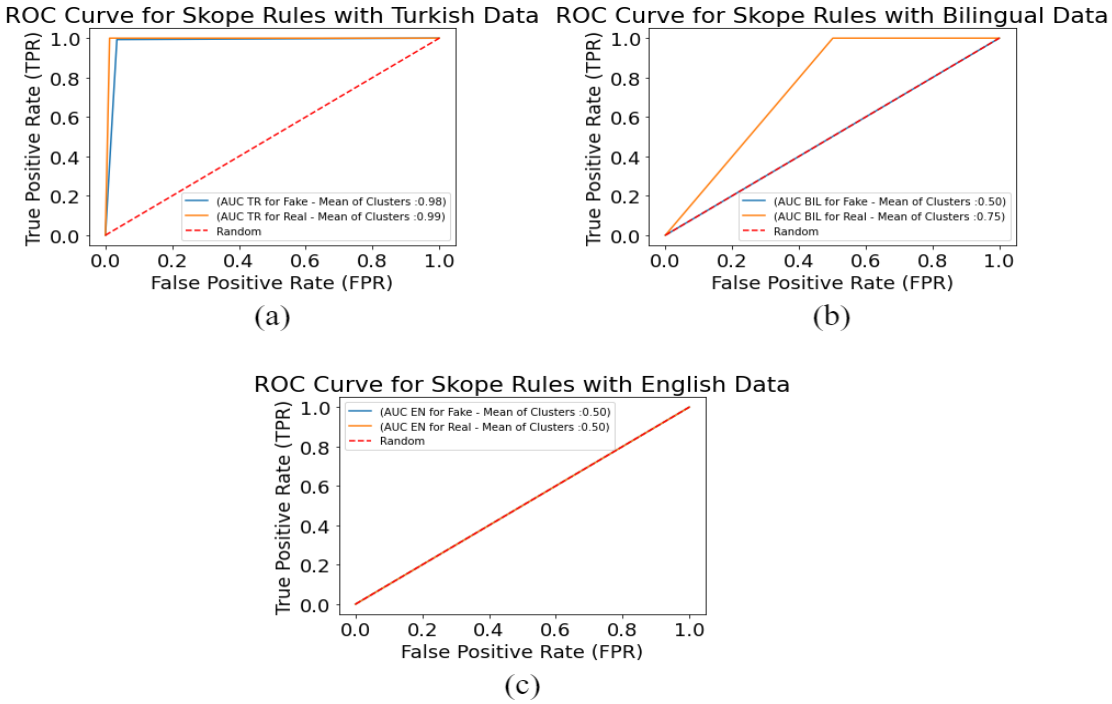


Figure 11. *ROC Curves for skope-rules Results obtained by Mean of Clusters; a) Turkish Language Data, b) Bilingual Language Data, c) English Language Data*

Confusion Matrix is also taken to see *TP*, *TN*, *FP*, *FN* results for both of the results. In Table 6, the confusion matrix can be seen. In this table, abbreviations are defined as *EN* for English, *TR* for Turkish language, and also as *BIL* for Bilingual languages, as well.

Table 6. *Confusion Matrix for both Pipeline and skope-rules Results*

	Pipeline		skope-rules Test Case 1		skope-rules Test Case 2		
Classes	*Fake*	*Real*	*Fake*	*Real*	*Fake*	*Real*	Language
Fake	141	25	104	3	303	5	*TR*
Real	40	126	1	106	0	308	*TR*
Fake	975	55	1	2024	1	549	*EN*
Real	238	792	0	2025	0	550	*EN*
Fake	1055	141	973	1159	745	113	*BIL*
Real	485	711	1100	1032	1	857	*BIL*

In the bilingual aspect of our study, there is an independence for the type of language which is considered to classify as "*Fake*" or "*Real*" class. The reason why this kind of independence highly corresponds to the majority voting and skope-rules are based on the Grad-CAM motivated class activations as well. In the representation learning perspective of our study, deep and machine learning harmonium was empowered with the interpretable logical rules based on some cases in the experiments for classification scoping.

The generalization is a golden aim for the machine learning research area for general purpose problem solving. Although it is very difficult to solve as a real-world problem, with the help of the objective evaluation of the experimental results in our study as shown above, we prove the suitable form of our bilingual approach with its high generalizability to classify the "*Fake*" or "*Real*" speech parts from raw audio samples.

5. CONCLUSION

In the literature, there are a lot of GAN-based models. Regardless of what language it is in such a bilingual form (i.e., Turkish or English), the main contribution of our study is to be able to recognize the fake speech as spoken content with the conversation. For a generalized classification purpose, deep learning-based features were used to feed an ensemble classification pipeline and a mining process of interpretable logical rules. The semantical gap is a known problem in machine learning research area. Deep learning has a solution to fill this gap with the use of the representational learning, and it achieves a more accurate classification and detection result for bilingual fake speech recognition. As a contribution of our study, our experimental results have higher classification and recognition accuracy. These results are obtained for each individual language performance. Mentioned results are approximately as 90% for training and as 80.33% for testing stages with classification pipeline in our study. This pipeline is compiled with the use of a majority voting-based ensemble classifier in which the performance is reached as 73% with the appropriate test cases as well. This kind of accurate classification and detection results are mostly based on the feature learning and interpretation capabilities of representational learning in our deep learning infrastructure.

For a fair generalization of the bilingual fake speech recognition process, deep learning's generative model's class activation as heat maps used in a mining process of interpretable logical rules for classification scope were investigated to extract appropriate rules. In the manner of objective performance metric from information retrieval theory, our approach with generalization capabilities seems to be more preferable than single staged non-bilingual (i.e., monolingual) models as well. In further studies, we plan to add some other deep learning models to our methodology. In addition, we will expand the test sets with different scenarios as well.

CONFLICTS OF INTEREST

No conflict of interest was declared by the authors.

REFERENCES

[1] Imran, M., Ali, Z., Bakhsh, S. T., Akram, S., "Blind Detection of Copy-Move Forgery in Digital Audio Forensics", IEEE Access, 5: 12843-12855, (2017).

[2] Mannepalli, K., SubbaRamaiah, V., Raghu, K., "Speech forgery detection of framed sentences in audio recordings using DTW", European Journal of Molecular & Clinical Medicine, 7(8): 2269-2274, (2020).

[3] Baskoro, A. B., Cahyani, N., Putrada, A. G., "Analysis of voice changes in anti-forensic activities case study: voice changer with telephone effect", International Journal on Information and Communication Technology (IJoICT), 6(2): 64-77, (2020).

[4] Shi, Y., Liu, H., Wang, Y., Cai, M., Xu, W., "Theory and application of audio-based assessment of cough", Journal of Sensors, Article ID: 9845321, 1–7, (2018).

[5] Maher, R. C., "Audio forensic examination", IEEE Signal Processing Magazine, 26(2): 84-94, (2009).

[6] Ally, M., Alotaibi, M. S., "A novel deep learning model to detect COVID-19 based on wavelet features extracted from Mel-scale spectrogram of patients' cough and breathing sounds", Informatics in Medicine Unlocked, 32(101049): 1-11, (2022).

[7] Lia, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M., "Deep Learning for Generic Object Detection: A Survey", International Journal of Computer Vision, 128: 261-218, (2020).

[8] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., "Generative Adversarial Networks", Communications of the ACM, 63(11): 139-144, (2020).

[9] Radford, A., Metz, L., Chintala, S., "Unsupervised representation learning with deep convolutional generative adversarial networks", arXiv preprint, Machine Learning (cs.LG), Computer Vision and Pattern Recognition (cs.CV), arXiv:1511.06434, 1-16, (2015).

[10] Beguš, G., "CiwGAN and fiwGAN: Encoding information in acoustic data to model lexical learning with Generative Adversarial Networks", Neural Networks, 139: 305-325, (2021).

[11] Donahue, C., McAuley, J., Puckette, M. S., "Adversarial audio synthesis", 7th International conference on learning representations (ICLR2019), New Orleans LA, USA, May 6-9, OpenReview.net, 1–16, (2019). Online: https://openreview.net/forum?id=ByMVTsR5KQ.

[12] Rodionov, S., "Info-wgan-gp", (2018). Online: https://github.com/singnet/semantic-vision/tree/master/experiments/concept_learning/gans/info-wgan-gp. Access date: 25.05.2023

[13] Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., Teoh, W.Z., Sotelo, J., de Brébisson, A., Bengio, Y., Courville, A.C., "MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis", ArXiv Preprint, Audio and Speech Processing (eess.AS), Computation and Language (cs.CL), Machine Learning (cs.LG), Sound (cs.SD), 1-14, arXiv:1910.06711, (2019).

[14] Kong, J., Kim, J., Bae, J., "HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis", ArXiv Preprint, Sound (cs.SD); Machine Learning (cs.LG); Audio and Speech Processing (eess.AS), 1-14, arXiv:2010.05646, (2020).

[15] Kocaoğlu, D., Turgut, K., Konyar, M. Z., "Sector-Based stock price prediction with machine learning models", Sakarya University Journal of Computer and Information Sciences, 5(3): 415-426, (2022).

[16] Bhateja, V., Taquee, A., Sharma, D. K. "Pre-Processing and classification of cough sounds in noisy environment using SVM", 4th International Conference on Information Systems and Computer Networks (ISCON), Mathruda, India, November 21-22, 822-826, (2019).

[17] Rasmussen, C.E., Williams, C.K.I., "Gaussian Processes for Machine Learning", the MIT Press, Massachusetts Institute of Technology, (2006). ISBN 026218253X.

[18] Gao, W., Bao, W., Zhou, X., "Analysis of cough detection index based on decision tree and support vector machine", Journal of Combinatorial Optimization, 37: 375–384, (2019).

[19] Karasulu, B., "Sound scene and events detection using deep learning in the scope of cyber security for multimedia systems", Acta Infologica, 3(2): 60-82, (2019).

[20] Virtanen, T., Plumbley, M.D., Ellis, D. (Eds.)., "Computational analysis of sound scenes and events", Book Cham, Switzerland: Springer International Publishing AG. (2018).

[21] Bäckström, T., Räsänen, O., Zewoudie, A., Zarazaga, P.P., Koivusalo, L., Das, S., Mellado, E.G., Mansali, M.B., Ramos, D., Kadiri, S., Alku, P., "Introduction to Speech Processing", Second Edition, (2022). Online: https://speechprocessingbook.aalto.fi. Access date: 25.05.2023

[22] Çakır, E., "Deep neural networks for sound event detection", (Doctoral Dissertation, Tampere University, Finland), (2019). Online: https://tutcris.tut.fi/portal/files/17626487/cakir_12.pdf. Access date: 25.05.2023

[23] Juillerat N., Hirsbrunner, B., "Low latency audio pitch shifting in the frequency domain", International Conference on Audio Language and Image Processing (ICALIP), Shanghai, China, November 23-25, 16-24, (2010).

[24] Damskägg, E.-P., Välimäki, V., "Audio time stretching using fuzzy classification of spectral bins", Applied Sciences, 7(12): 1293, (2017).

[25] Govender, D., "Investigating Audio Classification to Automate the Trimming of Recorded Lectures", University of Cape Town, February, (2018). Online: https://pubs.cs.uct.ac.za/id/eprint/1260/1/Thesis-final.pdf. Access date: 25.05.2023

[26] McFee, B., Raffel, C., Liang, D., Ellis, D., Mcvicar, M., Battenberg, E., Nieto, O., "Librosa: Audio and Music Signal Analysis in Python", Proceedings of the Python in Science Conference, Austin, Texas, USA, 6 - 12 July, 18-24, (2015).

[27] Griffin, D., Lim, J., "Signal estimation from modified short-time Fourier transform", IEEE Transactions on Acoustics, Speech, and Signal Processing, 32(2): 236-243, (1984).

[28] Laroche, J., Dolson, M., "Improved phase vocoder time-scale modification of audio", IEEE Transactions on Speech and Audio Processing, 7(3): 323-332, (1999).

[29] Zhang, Z., Xu, S., Zhang, S., Qiao, T., Cao, S., "Learning attentive representations for environmental sound classification", IEEE Access, 7: 130327-130339, (2019).

[30] Boztepe, E.B., Karakaya, B., Karasulu, B., Ünlü, I., "An approach for audio-visual content understanding of video using multimodal deep learning methodology", Sakarya University Journal of Computer and Information Sciences (SAUCIS), 5(2): 181-207, (2022).

[31] Kıvrak, E.A., Karasulu, B., Sözbir, C., Türkay, A., "A deep learning based software tool for audio mood classification using audio attributes", Veri Bilimi Dergisi, 4(3): 14-27, (2021).

[32] Korzeniowski, F., Widmer, G., "Feature learning for chord recognition: The deep chroma extractor", Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR), New York, USA, arXiv preprint, arXiv:1612.05065, August 7-11, 1-7, (2016).

[33] Ganchev, T.D., "Speaker recognition", University of Patras, Wire Communications Laboratory, Dept. of Computer and Electrical Engineering, Gree e, Dissetation for Doctor of Philosophy, (2005). https://thesis.ekt.gr/thesisBookReader/id/13812#page/1/mode/2up. Access date: 25.05.2023

[34] Müller, M., Ewert S., "Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features", Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR2011), Miami, Florida, USA, October 24-28, 215-220, (2011). http://ismir2011.ismir.net/papers/PS2-8.pdf. Access date: 25.05.2023

[35] Panayotov, V., Chen, G., Povey, D., Khudanpur, S., "Librispeech: An ASR corpus based on public domain audio books", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, April 19-24, 5206-5210, (2015).

[36] Tensorflow Library Documentation, (2023). Online: https://www.tensorflow.org/api_docs. Access date: 25.05.2023

[37] Keras Library Documentation, (2023). Online: https://keras.io/api/. Accessed on May 25, 2023.

[38] Xu, B., Wang, N., Chen, T., Li, M., "empirical evaluation of rectified activations in convolutional network", arXiv preprint, Machine Learning (cs.LG), Computer Vision and Pattern Recognition (cs.CV), Machine Learning (stat.ML), arXiv:1505.00853v2, 1-5, (2015).

[39] Buduma, N., Lacascio, N., "Fundamentals of deep learning: designing next-generation machine intelligence algorithms", O'Reilly Media UK Ltd., (2017). ISBN: 978–1–491–92561–4.

[40] Krizhevsky, A., Sutskever, I., Hinton, G.E., "ImageNet classification with deep convolutional neural networks", Communications of the ACM, Research Highlights, 60(6): 84-90, (2017).

[41] Pratiwi, H., Windarto, A.P., Susliansyah, S., Aria, R.R., Susilowati, S., Rahayu, L.K., Fitriani, Y., Merdekawati, A., Rahadjeng, I.R., "Sigmoid activation function in selecting the best model of artificial neural networks", Journal of Physics Conference Series, 1471, 012010, 1st Bukittinggi International Conference on Education, West Sumatera, Indonesia, October 17-18, 1-8, (2019).

[42] Kingma, D.P., Ba, J., "Adam: A Method for Stochastic Optimization", Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), San Diego, USA, May 7-9, 1-13, (2015).

[43] Suh, S., Lee, H., Jo, J., Lukowicz, P., Lee, Y.O., "Generative oversampling method for imbalanced data on bearing fault detection and diagnosis", Applied Science, 9(4:746): 1-16, (2019).

[44] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D., "Grad-CAM: visual explanations from deep networks via gradient-based localization", International Journal of Computer Vision, 128(2): 336–359, (2020).

[45] Esener, I.I., Ergin, S., Yüksel, T., "A Genuine GLCM-based feature extraction for breast tissue classification on mammograms", International Journal of Intelligent Systems and Applications in Engineering (IJISAE), 4 (Special Issue), 124-129, (2016).

[46] Özkan, K., "Comparing Shannon entropy with Deng entropy and improved Deng entropy for measuring biodiversity when a priori data is not clear", Forestist, 68(2): 136-140, (2018).

[47] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., "Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research", arXiv preprint, Machine Learning (cs.LG), Mathematical Software (cs.MS), 12(85): 2825-2830, (2011).

[48] Cunningham, P., Delany, S.J., "*k*-Nearest neighbour classifiers", ACM Computing Surveys, Article No: 128, 54(6): 1-25, (2007).

[49] Manning, C.D., Raghavan, P., Schütze, H., "Introduction to Information Retrieval", Cambridge University Press. (2008). ISBN:978-0-521-86571-5

[50] Evgeniou, T., Pontil, M., "Support Vector Machines: Theory and Applications", In: Paliouras, G., Karkaletsis, V., Spyropoulos, C.D. (Eds.), "Machine Learning and Its Applications", ACAI 1999, Lecture Notes in Computer Science, 2049, Springer, Berlin, Heidelberg, 249-257, (2001).

[51] Bors, A.G., "Introduction of the Radial Basis Function (RBF) Networks", Online Symposium for Electronics Engineers, DSP Algorithms: Multimedia, 1:1-7, (2001).

[52] Ebden, M., "Gaussian Processes: A Quick Introduction", arXiv preprint, Statistics Theory (math.ST), arXiv:1505.02965, 1-13, (2015).

[53] Fei, Y., Rong, G., Wang, B., Wang, W., "Parallel L-BFGS-B algorithm on GPU", Computers & Graphics, 40: 1-9, (2014).

[54] Rokach, L., Maimon, O., "Decision Trees", In: Maimon, O., Rokach, L. (Eds.), "Data Mining and Knowledge Discovery Handbook", Springer, Boston, MA, 165-192, (2005).

[55] Suryakanthi, T., "Evaluating the Impact of GINI Index and information gain on classification using decision tree classifier algorithm", International Journal of Advanced Computer Science and Applications, 11(2): 612-619, (2020).

[56] Anagnostopoulos, T.T., Skourlas, C., "Ensemble Majority Voting Classifier for Speech Emotion Recognition and Prediction", Journal of Systems and Information Technology, 16(3): 222-232, (2014).

[57] Gardin, F., Gautier, R., Jaffre, R., Goix, N., Ndiaye, B., Schertzer, J.-M., "GitHub - scikit-learn-contrib/skope-rules: machine learning with logical rules in Python", v1.0.1, (2020). Online: https://github.com/scikit-learn-contrib/skope-rules. https://2018.ds3-datascience-polytechnique.fr/ wp-content/uploads/2018/06/DS3-309.pdf. Access date: 25.05.2023

[58] Lal, G.R., Chen, X., Mithal, V., "TE2Rules: Extracting Rule Lists from Tree Ensembles", arXiv preprint, Machine Learning (cs.LG), Artificial Intelligence (cs.AI), arXiv:2206.14359, 1-17, 2022.

[59] Friedman, J.H., Popescu, B.E., "Predictive learning via rule ensembles", The Annals of Applied Statistics, 2(3): 916-954, (2008).

[60] Google Colab Website, (2023). Online: https://colab.research.google.com. Access date: 25.05.2023

[61] Python Doc Website, (2023). Online: https://www.python.org/doc/. Access date: 25.05.2023

[62] OpenSLR Dataset, (2023). Online: https://www.openslr.org. Access date: 25.05.2023

[63] Piispaanen, P. Blažek, V., "Altaic Languages – History of research, survey, classification, and a sketch of comparative grammar in collaboration with M. Schwarz and O. Srba", Journal of Old Turkic Studies, 4(1): 266-274, (2020).

[64] Johanson, L., "Turkic languages - Old Turkic, Uyghur, Qarakhanid, Ottoman", Encyclopædia Britannica website, (2023). Online: https://www.britannica.com/topic/Turkic-languages/Linguistic-structure. Access date: 25.05.2023

[65] Eberhard, D.M., Simons, G.F., C. D. Fennig, C. D. (Eds.), "Ethnologue: Languages of the World. Twenty-sixth edition", Dallas, Texas: SIL International, Turkish Language Ethnologue, (2023). Online: https://www.ethnologue.com/language/tur/. Access date: 25.05.2023

[66] Kolobov, R., Okhapkina, O., Omelchishina, O., Platunov, Bedyakin, A.R., Moshkin, V., Menshikov, D., Mikhaylovskiy, N., "MediaSpeech: Multilanguage ASR Benchmark and Dataset", arXiv preprint, Audio and Speech Processing (eess.AS), Sound (cs.SD), arXiv:2103.16193, 1-4, (2021).

[67] Youtube Website, (2023). Online: https://www.youtube.com. Access date: 25.05.2023

[68] Cowgill, W., Jasanoff, J.H., "Indo-European languages", Encyclopædia Britannica website, (2023). Online: https://www.britannica.com/topic/Indo-European-languages. Access date: 25.05.2023

[69] Eberhard, D.M., Simons, G.F., C. D. Fennig, C. D. (Eds.), "Ethnologue: Languages of the World", Twenty-sixth edition. Dallas, Texas: SIL International, English Language Ethnologue, (2023). Online: https://www.ethnologue.com/language/eng/. Access date: 25.05.2023

[70] Librivox free public domain audiobooks, LibriVox, (2023). Online: https://librivox.org. Access date: 25.05.2023

[71] Fawcett, T., "Introduction to ROC analysis", Pattern Recognition Letters, 27(8): 861- 874, (2006).

[72] Powers, D.M.W., "The problem of area under the curve", Proceedings of the IEEE International Conference on Information Science and Technology (ICIST2012), Wuhan, China, March 23-25, 567-573, (2012).