Comparison of Dimensionality Reduction Techniques on Audio Signals

Tamás Pál, Dániel T. Várkonyi

Eötvös Loránd University, Faculty of Informatics, Department of Data Science and Engineering,
Telekom Innovation Laboratories, Budapest, Hungary {evwolcheim, varkonyid}@inf.elte.hu
WWW home page: http://t-labs.elte.hu

Abstract: Analysis of audio signals is widely used and very effective technique in several domains like healthcare, transportation, and agriculture. In a general process the output of the feature extraction method results in huge number of relevant features which may be difficult to process. The number of features heavily correlates with the complexity of the following machine learning method. Dimensionality reduction methods have been used successfully in recent times in machine learning to reduce complexity and memory usage and improve speed of following ML algorithms. This paper attempts to compare the state of the art dimensionality reduction techniques as a building block of the general process and analyze the usability of these methods in visualizing large audio datasets.

1 Introduction

With recent advances in machine learning, audio, speech, and music processing has evolved greatly, with many applications like searching, clustering, classification, and tagging becoming more efficient and robust through machine learning techniques. Well performing dimensionality reduction methods have been successfully applied in many areas of computer science and interdisciplinary fields.

A very important phenomenon that justifies the use of such methods is the curse of dimensionality. The basic idea is that the increase of dimensionality increases the volume of space, causing the data to become sparse. Therefore, the amount of data required for a model to achieve high accuracy and efficiency increases greatly.

Considering audio signals, meaningful features need to be extracted before applying any dimensionality reduction method. Using methods for feature extraction like Short Time Fourier Transformation (STFT), Mel-frequency cepstral coefficient (MFCC), or high-level descriptors like zero-crossing-rate, representative data can be extracted from the audio data. After extraction these features can be projected into two-dimensional space for cluster analysis and examination of class separation.

The dataset used is the UrbanSound 8k dataset, containing 8732 labeled sound excerpts (≤ 4 seconds) of urban sounds from 10 classes, from which 5 have been used in this work: car horn, dog bark, engine idling, gun shot, and street music [5].

Related work is presented in Section 2, basic mathematical notation used is described in Section 3, while the different methods of the pipeline are briefly presented in Section 4. Section 5 contains data about the evaluation methodology, Section 6 presents the results and conclusions are formulated in Section 7.

The goal of this paper is to find a combination of feature extraction and dimensionality reduction methods which can be most efficiently applied to audio data visualization in 2D and preserve inter-class relations the most.

2 Related Work

A large number of research addresses the problem of mapping a collection of sounds into a 2-dimensional map and examining the results both visually and analytically.

Roma et al. [2] used MFCC and autoencoders as feature extraction methods and compared PCA, tSNE, Isomap, MDS, SOM and the Fruchterman-Reingold dimensionality reduction to map the output of the feature extraction process into 2D.

Fedden [7] used a very similar approach, with only different methods tried. As far as feature extraction is concerned, MFCCs and an autoencoder-like architecture called NSynth were used. PCA, t-SNE, UMAP methods were used for dimensionality reduction.

Hantrakul et al. [8] implements the usual pipeline of applying feature extraction, reshaping, then dimensionality reduction into 2D. STFT, MFCC, high-level features and the Wavenet encoder were used for feature extraction. PCA, UMAP and t-SNE were the candidates for the next step. The dataset consisted of different drum sample sounds. The results were analyzed through external clustering metrics like homogeneity, completeness score and V-measure, using k-means as clustering method.

Dupont et al. [11] used MFCCs combined with spectral flatness to represent the extracted information from the audio. PCA, Isomap and t-SNE were chosen for dimensionality reduction. In addition, supervised dimensionality reduction methods were included as well, like LDA (Linear Discriminant Analysis) and HDA (Heteroscedastic Discriminant Analysis).

Charles Lo's work [1] is focused on dimensionality reduction for music feature extraction. The author used feature vectors composed of: 13 MFCCs and high-level fea-

tures. Locally Linear Embedding (LLE), Autoencoder, t-SNE and PCA were used to map into lower dimensions. The dataset contained 1000 song clips equally distributed from 10 musical genres. In order to test the results, Gaussian mixture models were implemented to test cluster purity and supervised classification was also performed with kNN classifier. tSNE achieved the best classification performance.

3 Basic Notation

Given a dataset of discrete audio signals, we denote the i-th recording as $\boldsymbol{x^{(i)}}$ and the dataset as $X = \{\boldsymbol{x^{(1)}}, \boldsymbol{x^{(2)}}, ..., \boldsymbol{x^{(N)}}\}$. The corresponding ground-truth labels are stored in $Y = \{y^{(1)}, y^{(2)}, ..., y^{(N)}\}$. A single audio element of the dataset is comprised of a number of samples: $\boldsymbol{x^{(i)}} = (x_1^{(i)}, x_2^{(i)}, ...x_M^{(i)})$.

4 Methods

The building blocks of the process used in this work are presented in the following sections and are also shown on figure 1. As seen on the figure, a set of discrete audio signals are transformed trough consecutive methods to result in a 2D map of datapoints.

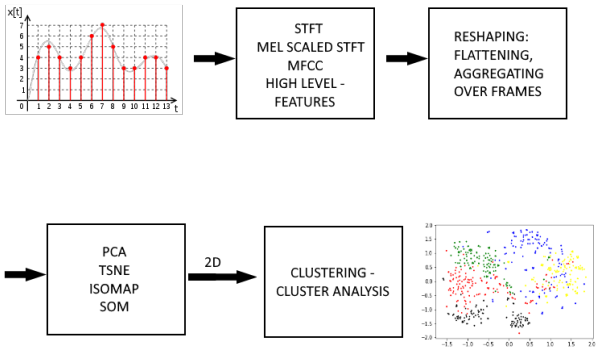

Figure 1: Steps of the process

4.1 Band-pass filtering

Applying filtering of some kind before feature extraction could be justifiable to remove frequency intervals that usually do not have discriminative power. The simplest such solution is applying a band-pass filter. A band-pass filter passes frequencies between a lower limit f_L and a higher limit f_H, while rejecting frequencies outside this interval. Band-pass filters are usually implemented using Butterworth filters, as these have maximally flat frequency response in the passband.

The following low-cutoff and high-cutoff frequency pairs have been used as band-pass filter specifications:

1. Low: 1 Hz, High: 10000 Hz
2. Low: 25 Hz, High: 9000 Hz
3. Low: 50 Hz, High: 8500 Hz
4. Low: 100 Hz, High: 8000 Hz
5. Low: 150 Hz, High: 7000 Hz

4.2 Feature Extraction

In order to extract meaningful data, that can be interpreted by machine learning models, the step of feature extraction is required. This step also acts as a dimensionality reduction process, because it transforms a sound segment represented by hundreds of thousands of samples to a data structure containing only a few thousand values or less. This is a crucial step as it enables to apply dimensionality reduction methods later that will ultimately project the sample into the lowest dimensionality of just 2 components.

STFT The Short-time Fourier transform (STFT) captures both temporal and spectral information from a signal. As a result, the STFT transforms the signal from time-amplitude representation to time-frequency representation. Formally, the STFT is calculated by applying the discrete Fourier transform with a sliding window to overlapping segments of the signal. It can be formulated in the following way:

$$STFT\{\boldsymbol{x^{(i)}}\} := S^{(i)}[k,n] = \sum_{m=0}^{L-1} x^{(i)}[n+m]w[m]e^{-jk\frac{2\pi}{L}m} \tag{1}$$

where $w[m]$ is the window function, L is the window length, n denotes the frame number and k denotes the frequency in question.

The STFT returns a complex-valued matrix, where $|S[n,k]|$ is the magnitude of the frequency bin k at time-frame n.

Mel-scaled STFT As human perception of pitch (frequency) is logarithmic, the Mel-scale is often applied to the frequency axis of the STFT output. Specifically, the Mel-scale is linearly spaced below 1000 Hz and turns logarithmic afterward.

The Mel-scaled frequency value can be computed from a frequency(f) in Hertz using the following equation:

$$Mel(f) = 2595 \cdot \log_{10}(1 + \frac{f}{700}) \tag{2}$$

MFCC Mel Frequency Cepstral coefficients are widely used in audio and speech recognition, and were introduced by Paul Mermelstein in 1976 [10]. Overview of the process:

1. Take the Short-time Fourier transform of the signal.

2. Apply the Mel filterbank to the power spectra of each frame, summing the energies in each frequency band. The Mel filterbank is a set of overlapping triangular filters.

3. Take the logarithm of the calculated energies.

4. Take the Discrete Cosine Transform (DCT) of the log filterbank energies, resulting in a so-called cepstrum. Intuitively, the cepstrum captures information of the rate of change in frequency bands. The DCT is related to the DFT (Discrete Fourier Transform). While DFT uses both sine and cosine functions to express a function as a sum of these sinusoids, the DCT uses only cosine functions. Formally (using the DCT-II):

$$C[k] = \sum_{n=0}^{N-1} x[n] \cdot \cos\left[\frac{\pi}{N}(n+\frac{1}{2})k\right], \qquad 1 \leq k \leq N \tag{3}$$

5. The MFCC coefficients are the resulting DCT coefficients.

High-level features High level descriptors that capture temporal or spectral information about the signal are often used in similar experiments. The following six features have been used for frame number n:

- Root mean square:

$$v_{RMS}(n) = \sqrt{\frac{\sum_{m=0}^{L-1} x[n+m]^2}{L}} \tag{4}$$

  where L is the window length.

- Zero crossing rate:

$$v_{ZC}(n) = \frac{1}{2L}\sum_{m=0}^{L-1} |sgn(x[n+m]) - sgn(x[n+m-1])| \tag{5}$$

  where L is the window length. The zero crossing rate is the number of changes of sign in consecutive audio samples.

- Spectral centroid:

$$v_{SC}(n) = \frac{\sum_{k=0}^{K-1} k \cdot |S[k,n]|^2}{\sum_{k=0}^{K-1} |S[k,n]|^2} \tag{6}$$

  The spectral centroid indicates where the center of mass of the spectrum is located.

- Spectral rolloff:

$$v_{SR}(n) = i \Big|_{\sum_{k=0}^{i} |S[k,n]| = \kappa \sum_{k=0}^{K-1} |S[k,n]|} \tag{7}$$

  Spectral rolloff is the frequency below which a specified percentage of the total spectral energy lies, with κ denoting the percentage score.

- Spectral bandwidth:

$$v_{SS}(n) = \sqrt{\frac{\sum_{k=0}^{K-1} (k - v_{SC}(n))^2 \cdot |S[k,n]|^2}{\sum_{k=0}^{K-1} |S[k,n]|^2}} \tag{8}$$

  The spectral bandwidth is a measure of the average spread of the spectrum in relation to its centroid.

- Spectral flatness:

$$v_{SF}(n) = \frac{\sqrt[K]{\prod_{k=0}^{K-1} |S[k,n]|}}{\frac{1}{K}\sum_{k=0}^{K-1} |S[k,n]|} \tag{9}$$

  The spectral flatness is the ratio of geometric mean and arithmetic mean of the magnitude spectrum.

4.3 Reshaping methods

After the feature extraction step, the feature set takes the (N,k,n) form, where N is the size of the dataset, k denotes the number of frequency bins (or features) and n is the number of time frames. To be able to perform dimensionality reduction on the whole dataset, the feature set needs to be transformed to a (N,d) form, where d is the size of a feature vector after the reshaping transformation. Three such methods have been found in associated papers:

Flattening Stacking consecutive rows horizontally to form one, long 1-dimensional vector. Using this method, a feature set of shape (k,n) will take the $(1,k \times n)$ form after the reshaping operation. Used by [8].

$$\begin{bmatrix} S_{11} & S_{12} & S_{13} & \dots \\ S_{21} & S_{22} & S_{23} & \dots \\ \vdots & & \ddots & \end{bmatrix} \Longrightarrow$$

$$\begin{bmatrix} S_{11} & S_{12} & S_{13} & \dots & S_{1N} & S_{21} & S_{22} & S_{23} & \dots \end{bmatrix}$$

Aggregation of features over frames: mean, std By calculating some statistical features across all frames, the spectrogram can be reduced to one feature vector, where statistical values (μ_S - mean of random variable, σ_S - standard deviation of random variable) calculated for each frequency band are stacked vertically in a continuous fashion. Using this method, a feature set of shape (k,n) will take the $(k \times A, 1)$ form after the reshaping operation, where A is the number of statistics calculated. Used by [2], [7], [11].

$$\begin{bmatrix} S_{11} & S_{12} & S_{13} & \dots \\ S_{21} & S_{22} & S_{23} & \dots \\ \vdots & & \ddots & \end{bmatrix} \Longrightarrow \begin{bmatrix} \mu_{S1} \\ \sigma_{S1} \\ \mu_{S2} \\ \sigma_{S2} \\ \vdots \end{bmatrix} \tag{10}$$

Aggregation of features over frames: mean, std, min, max Similar to the solution above, but with additional minimum and maximum values calculated for each frequency bin. Used by [1].

4.4 Dimensionality Reduction Methods

Once the data is reshaped, dimensionality reduction is applied in order to project the data into a lower dimensional space.

Principal Component Analysis (PCA) Principal Component Analyis is a basis transformation that results in a set of orthogonal basis vectors along which the variance of the data is maximal.

Calculating PCA involves the following steps:

1. Denote the reshaped dataset with $X \in \mathbb{R}^{N \times d}$.
2. Standardization: subtract the mean, and divide by the standard deviation for each feature column. Each column will a have a mean of 0 and a standard deviation of 1.
3. Form the covariance matrix of X.

   The variance-covariance matrix consists of the variances of the variables along the main diagonal and the covariances between each pair of variables in the other matrix positions. The covariance matrix is symmetric, thus it's diagonalizable.

   The covariance matrix is calculated from X using the following formula:

$$C = \frac{1}{n-1} X^T X, \qquad C \in \mathbb{R}^{d \times d} \tag{11}$$

4. Compute the eigenvectors and eigenvalues of the covariance matrix. Formally, the eigendecomposition of a matrix is described as:

$$C = VDV^{-1} \tag{12}$$

   where V is a square matrix whose ith columns represents the ith eigenvector of C. D is a diagonal matrix whose diagonal elements represent the eigenvalues of C.

   These eigenvectors are called the principal components of the dataset, ie. the axis on which the data is more dispersed.

5. Sort the eigenvalues and the corresponding eigenvectors in decreasing order.
6. Leave the first d' number of eigenvalues and their corresponding eigenvectors. $V^{'} = V_{ij}, \quad 1 \leq i \leq d, 1 \leq j \leq d'$
7. The transformed dataset will be:

$$X_{low\_dim} = XV^{'}, \quad \in \mathbb{R}^{N \times d'} \tag{13}$$

t-Stochastic Neighbor Embedding (TSNE) t-SNE is a nonlinear dimensionality reduction method developed by Laurens van der Maaten and Geoffrey Hinton in 2008 [9]. t-SNE has the ability to preserve local structure by finding a distribution that models the neighboring relations of the data in the original space and mapping this into lower dimensional embedding (ie.: finding a lower dimensional distribution) while preserving local spatial relations.

Using this probabilistic approach, the first main idea is that to convert Euclidean distances into conditional probabilities that reflect these spatial similarities. Let P be the joint probability distribution of the data in high dimensional space, represented by a normal distribution:

$$p_{j|i} = \frac{1}{\sum_{k \neq i} exp(-\frac{\|\boldsymbol{x^{(i)}} - \boldsymbol{x^{(k)}}\|^2}{2\sigma_i^2})} exp(-\frac{\|\boldsymbol{x^{(i)}} - \boldsymbol{x^{(j)}}\|^2}{2\sigma_i^2}) \tag{14}$$

Intuitively, this expresses the conditional probability of x_i picking x_j as its neighbor if neighbors were chosen in proportion to their density around x_i.

Student t-distribution (with one degree of freedom) Q is used to model the distribution of the low dimensional data:

$$q_{i|j} = \frac{(1 + \|y^{(i)} - y^{(j)}\|^2)^{-1}}{\sum_{k \neq i} (1 + \|y^{(i)} - y^{(k)}\|^2)^{-1}} \tag{15}$$

Due to difficult optimization, t-SNE introduces a modified version of the P distribution, a symmetric probability density function, defined as:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2n} \tag{16}$$

t-SNE uses gradient descent to minimise the error between the two distributions and adds a decaying momentum term as an optimization. The authors introduce the Kullback-Leibler (KL) divergence as the cost (loss) function:

$$C = KL(P||Q) = \sum_{i \neq j} p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{17}$$

The gradient of the cost function is the following:

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j) \tag{18}$$

In essence, the t-SNE algorithm is based on the following steps:

1. Compute pairwise similarities $p_{j|i}$.
2. Let $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2n}$.
3. Randomly sample the low dimensional points from a Gaussian distribution with small variance (ie.: $Y \sim N(0, 10^{-4} I)$).
4. Repeat the following steps for T number of iterations: `for t=1 to T do:`

(a) Compute low-dimensional similarities q_{ij}.

(b) Compute gradient $\frac{\partial C}{\partial y}$.

(c) Update $y^{(t)} = y^{(t-1)} + \eta \frac{\partial C}{\partial y} + \alpha(t)(y^{(t-1)} - y^{(t-2)})$, where η is the learning rate, α the decaying weight coefficient of the momentum term.

An important hyperparameter of the t-SNE method is the perplexity. According to the authors, perplexity is the measure of the assumed number of neighbors, i.e. it balances the attention between the local and the global structure of the dataset. While typical values are situated between 1 and 50, the optimal perplexity value is a function of the dataset size, thus manual tuning is necessary.

Isomap Isomap is a nonlinear dimensionality reduction method, belonging to the group of manifold learning methods [4]. It assumes that the data cannot be well represented in a subspace, but it can by a manifold.

The steps of the Isomap algorithm are:

1. Construct a weighted neighborhood graph using k-NN or ε-radius and assign the Euclidean distances between the nodes to the edge weights.

2. Compute the shortest path between each node using the Floyd-Warshall or Dijkstra's algorithm. Construct a distance matrix D from these pairwise geodesic distances.

3. Apply the multidimensional scaling (MDS) algorithm on the distance matrix calculated above. The following steps are part of the classical MDS algorithm. Firstly, double centering is applied to the squared distance matrix, using the centering matrix: $H = I - \frac{1}{n} ee^T$, where e is a $N \cdot 1$ column vector containing 1-s.
$$B = -\frac{1}{2} HD^2 H \tag{19}$$

4. Calculate the eigendecomposition of B.
$$B = P\Lambda P^{-1} \tag{20}$$
where P contains the eigenvectors and Λ is a diagonal matrix containing the eigenvalues.

5. The lower dimensional embedding of the datapoints is given by:
$$X_{low\_dim} = P_{d'} \cdot \Lambda_{d'}^{\frac{1}{2}} \tag{21}$$
where $\Lambda_{d'}^{\frac{1}{2}}$ contains the square roots of the d' largest eigenvalues, and $P_{d'}$ is the matrix containing the eigenvectors corresponding to the first d' largest eigenvalues.

Self Organizing Map (SOM) Self Organizing Maps (SOMs) were introduced by Finnish professor Teuvo Kohonen in 1982 [12]. They are a type of Artificial Neural Network, that learns a two-dimensional representation of an input data of arbitrary dimension through an unsupervised learning process. Architecturally, SOMs contain two types of neurons (nodes): input and map neurons. The number of input nodes correspond to the dimensionality of the data, while the number of map nodes are arbitrarily chosen, and are arranged into a rectangular grid (map). Every input node is connected to every map node, with a certain weight w_{ij} associated to it.
The learning process can be summarized in the following way:

1. Randomly initialize all weights.

2. Pick one random sample from the dataset.

3. For each node of the map, calculate the Euclidean distance between the input vector and the node's associated weight vector:
$$d_j(\boldsymbol{x^{(i)}}) = \sum_{k=1}^{d'} (x_k^{(i)} - w_{jk})^2 \tag{22}$$
where d' is the dimensionality of the input data, $x^{(i)}$ is the input vector and w_{jk} represents the weight associated to the connection between the j-th map node and the k-th element of the input vector. The node that produces the smallest distance is called the BMU (Best Matching Unit).

4. The weights of the BMU, as well as the weights of its neighboring nodes, are updated. The radius of neighbors to be updated is initially large, but it is reduced after each iteration.
$$w_{jk}^{t+1} = w_{jk}^t + \eta(t) T_{j,BMU}(t)(x_k^{(i)} - w_{jk}^t) \tag{23}$$
where $\eta(t)$ is the decaying learning rate, BMU is the index of the Best Matching Unit and $T_{j,BMU}(t)$ is the neighborhood function of the form:
$$T_{j,BMU}(t) = e^{-dist(j,BMU)/2\sigma(t)^2} \tag{24}$$
If the distance between the two map nodes with indices j and BMU is larger, then the output of the neighborhood function will be smaller, and the j-th node will not get a significant update, as opposed to a closer node. The $\sigma(t)$ parameter shrinks with time, considering smaller and smaller neighborhoods.

5. If t exceeds the maximal iteration number, the algorithm stops.

5 Evaluation

For the evaluation of the class separability on the resulting 2D map, external cluster analysis metrics are used to analyze the correspondence between the ground-truth labels and the results of the clustering algorithm.

5.1 Clustering

K-means is used for clustering, as it performed more positively during preliminary testing, as opposed to other methods like hierarchical clustering, DBSCAN or OPTICS. The number of centroids was fixed to the number of classes, namely 5. The algorithm ran 10 times with different centroid seeds.

5.2 Evaluation metrics

We concentrated on using external clustering evaluation methods that compare the labels assigned by a clustering algorithm with ground truth labels; similarly to [6], [8]. Notation: TP - number of pairs of points that are in the same cluster and in the same class, FP - number of pairs of points that are in the same cluster but in different classes, TN - number of pairs of points that are in different clusters and in different classes, FN - number of pairs of points that are in different clusters but in the same class.

Adjusted Rand Index

$$RI = \frac{TP+TN}{TP+FP+TN+FN} \tag{25}$$

Fowlkes-Mallows score

$$FM = \sqrt{\frac{TP}{TP+FP} \cdot \frac{TP}{TP+FN}} \tag{26}$$

V-measure It is calculated as the weighted (here $\beta = 1$) harmonic mean between homogeneity (h) and completeness (c), which are also external cluster analysis metrics. Homogeneity is satisfied when every cluster contains only data points which are members of a single class. Completeness is satisfied when all points that are members of the same class, are also the members of the same cluster.

$$V_\beta = \frac{(1+\beta) \cdot h \cdot c}{(\beta \cdot h) + c} \tag{27}$$

The evaluation of the results with these supervised metrics measures the extent to which audio recordings of the same class get projected to points that are close to each other and form spatially coherent, independent clusters.

5.3 Distribution of data

Five categories of sound samples were used from the UrbanSound8k dataset: car horn (150 samples), dog bark (150 samples), street music (142 samples), gunshot (150 samples), idling engine (150 samples).

The recordings of each category contain recordings of varying noise levels, environmental sounds and overlapping sounds.

5.4 Hyperparameters

A specific hyperparameter-configuration notation is introduced for presenting the results, which is formatted in the following way: *F-FE-NrBin-Reshape-DimRed-Hyperp*. The meaning of the symbols and the possible values of the hyperparameters are described below:

- *F*: value between 0 and 5. 0 means no filtering, 1-5 encode the cutoff frequency pairs identified in section 4.1.
- *FE*: identifier of feature extraction method: STFT, Mel scaled STFT (*mel*), MFCCs or high-level features (*feat*).
- *NrBin*: number of frequency bins: 13, 20, 40 bins. Only valid for the STFT, Mel-STFT, MFCC methods.
- *Reshape*: identifier of reshaping method as described in section 4.3:
  1. Flattening.
  2. Calculating mean and standard deviation across frames.
  3. Calculating mean, standard deviation, minimum and maximum values across frames.
- *DimRed*: Dimensionality reduction method, with 4 possibilities and possible further parameters for some methods (*Hyperp*):
  1. PCA - no further hyperparameter was identified, that would significantly influence the outcome.
  2. t-SNE - identified hyperparameter: *Perplexity*. Values to be tested: 20, 40, 60.
  3. Isomap - identified hyperparameter: *Number of neighbors*. Values to be tested: 20, 40, 60.
  4. SOM - identified hyperparameter: *Sigma*. Values to be tested: 1, 2, 4.

All possible value combinations for the different hyperparameters described above generates 1500 different runs. Gridsearch was executed over the whole set of combinations. Each configuration was run 3 times to ease the effects of random components.

6 Results

Table 1., Table 5., Table 6. show the top 10, the middle 10 and the worst 10 results according to the mean of the clustering indices. Similarly, Table 2., Table 3., Table 4 show the best 3 results for each dimensionality reduction method. As one can see processes containing MFCC as feature extraction method and TSNE as dimensionality reduction method performed the best and are presented in the top results. Not just the top 10 but the best 26 results are all variations of this configuration. Although several

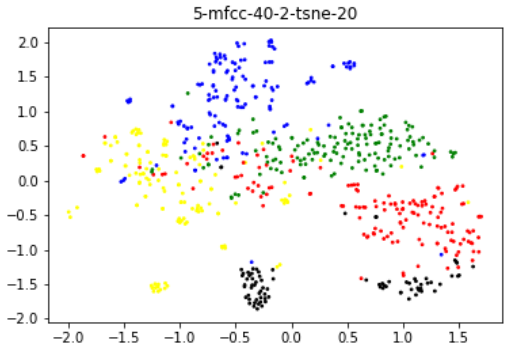

Figure 2: Best result.

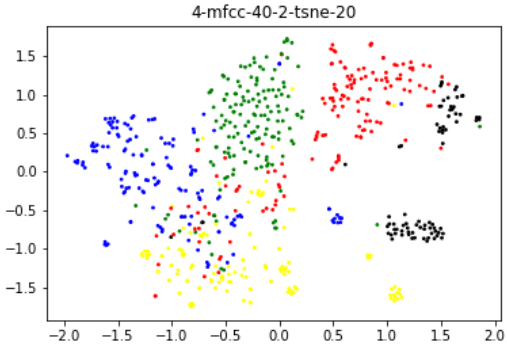

Figure 3: Second best result.

researchers didn't find significant differences between the performance of audio feature extraction methods e.g.: [3], our research showed the superiority of MFCC-TSNE combination compared to other examined combinations. MFCC outperforms the STFT and the high level statistical feature extraction (which are more general methods for not only audio, but other signals as well). TSNE performs well with every feature extraction method: it occupies the top 5 results for each feature extraction method in our experiments.

Configuration	Σ	ARI	FM	V-m
5-mfcc-40-2-tsne-20	0.525	0.466	0.579	0.530
4-mfcc-40-2-tsne-20	0.510	0.455	0.568	0.506
4-mfcc-40-3-tsne-20	0.496	0.439	0.553	0.497
4-mfcc-40-3-tsne-40	0.494	0.435	0.551	0.498
3-mfcc-40-3-tsne-20	0.492	0.435	0.550	0.491
5-mfcc-40-3-tsne-20	0.488	0.429	0.545	0.488
2-mfcc-40-2-tsne-20	0.484	0.426	0.545	0.482
0-mfcc-40-3-tsne-40	0.480	0.419	0.537	0.485
1-mfcc-40-3-tsne-20	0.479	0.422	0.540	0.474
0-mfcc-40-2-tsne-40	0.478	0.419	0.537	0.479

Legend: Σ = Mean of all cluster metrics, ARI = adjusted Rand index, FM = Fowlkes-Mallows index, V-m. = V-measure

Table 1: Cluster evaluation results: best 10.

The top 3 results are illustrated on Figures 2, 3 and 4.

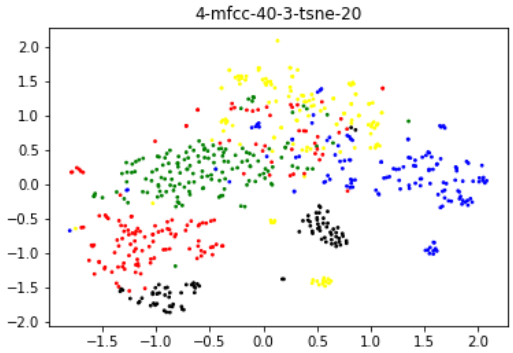

Figure 4: Third best result.

Configuration	Σ	ARI	FM	V-m
3-mfcc-40-2-pca-False	0.404	0.319	0.477	0.417
4-mfcc-40-2-pca-False	0.393	0.302	0.466	0.411
2-mfcc-40-2-pca-False	0.376	0.286	0.450	0.390

Table 2: Best 3 PCA results

Configuration	Σ	ARI	FM	V-m
0-mfcc-40-2-isomap-40	0.423	0.332	0.486	0.452
0-mfcc-40-2-isomap-60	0.422	0.331	0.484	0.451
0-mfcc-40-3-isomap-40	0.412	0.318	0.476	0.441

Table 3: Best 3 Isomap results

Configuration	Σ	ARI	FM	V-m
5-mfcc-40-3-som-1	0.399	0.326	0.470	0.401
4-mfcc-40-2-som-1	0.399	0.326	0.476	0.395
4-mfcc-40-2-som-4	0.398	0.339	0.472	0.382

Table 4: Best 3 results SOM

Configuration	Σ	ARI	FM	V-m
3-mel-13-2-tsne-60	0.213	0.143	0.317	0.180
0-mel-40-2-som-2	0.213	0.126	0.326	0.187
4-mfcc-13-1-pca-False	0.213	0.138	0.320	0.181
5-feat-6-2-tsne-40	0.213	0.134	0.308	0.196
4-feat-6-2-isomap-60	0.212	0.115	0.334	0.189
4-mel-20-3-som-2	0.212	0.132	0.329	0.177
1-feat-6-3-som-1	0.212	0.138	0.312	0.187
2-mfcc-13-1-pca-False	0.212	0.137	0.323	0.177
0-mel-13-2-tsne-20	0.212	0.139	0.315	0.183
3-stft-20-2-pca-False	0.212	0.074	0.369	0.194

Table 5: Middle 10 results

According to the results, the V-measure index is the closest to the mean of all albeit it goes without saying that the mean is heavily dependent on the selected indexes. The mean for the top results is very close to the value of the V-Measure index. Unfortunately the middle and the worst

results are not correlating to the V-Measure index.

Configuration	Σ	ARI	FM	V-m
4-stft-13-1-som-1	0.137	0.060	0.260	0.090
5-feat-6-1-isomap-40	0.137	0.050	0.267	0.092
4-mel-20-1-som-1	0.136	0.037	0.293	0.079
4-feat-6-1-som-1	0.133	0.060	0.252	0.087
0-mel-13-1-som-1	0.133	0.047	0.271	0.081
2-feat-6-1-som-1	0.132	0.059	0.254	0.084
0-feat-6-1-som-1	0.132	0.057	0.254	0.086
1-feat-6-1-som-4	0.130	0.057	0.247	0.085
2-feat-6-1-som-2	0.126	0.054	0.247	0.077
4-feat-6-1-som-2	0.120	0.046	0.241	0.075

Table 6: Worst 10 results

7 Conclusions

As it has been pointed out, the MFCC-TSNE combination achieved the best results. MFCC's success is supported by its widespread use in environmental sound classification [13] and it is also recommended and used by the authors of the dataset [5]. Apart from this observation, other conclusions can be drawn. By examining the impact of filtering, it is obvious that none of the five filtering variants were able to improve the results. Mel-scaled STFT and high-level features produced inefficient data representations and received weak evaluation scores. The poor performance of the Mel-scale could be attributed to it being optimized for speech. Using just 6 high-level descriptors is probably insufficient to represent the audio data well. The performance of these feature extraction methods is dependant on the nature of the dataset, some methods may perform better for some specific type of sounds. Considering the optimal number of frequency bins, higher values achieved better results, i.e. using 40 frequency bins proved to be the best choice. This is understandable, as more dimensions can hold more data, but it is likely that based on the curse of dimensionality, there is an upper limit at which performance starts to degrade. As far as the optimal reshaping method is concerned, both the second and third methods attained top 5 positions, while the first method (flattening) had a significantly weaker performance, as it is included in every list of worst performances. Yet again, the poor performance of the flattening method is a consequence of the curse of dimensionality phenomenon.

Considering dimensionality reduction methods, t-SNE clearly emerged as a winner, both in a visual and metrical sense. TSNE is famous for generally producing well separated, good visualizations, as pointed out by its authors [9]. While t-SNE achieved the best results according to the evaluation metrics, SOM also succeeded in producing well separable, good quality clusterings. For Isomap, many resulting plots were elongated and inhomogeneous. This may be explained by the fact that Isomap expects the data to lie on a manifold, but in actuality the data is not structured in such way. PCA also had poor results in the majority of the cases, probably in consequence of assuming that the data lies on a linear subspace, thus being unable to discover nonlinear relations.

7.1 Future work

Our future plan is to refine the developed process, elaborate new ones and apply the accumulated knowledge in medical and agricultural research, through classification of breathing sounds (COVID-19 detection) and bee sounds.

References

[1] C. Lo, "Nonlinear Dimensionality Reduction for Music Feature Extraction", Tech. Rep. CSC2515, Toronto University, 2012

[2] G. Roma, O. Green, P.A. Tremblay, "Adaptive Mapping of Sound Collections for Data-driven Musical Interfaces", The International Conference on New Interfaces for Musical Expression, 2019

[3] J. Jensen, M. Christensen, M. Murthi, S. Jensen, "Evaluation of MFCC estimation techniques for music similarity", Proceedings of the EUSIPC, 2006

[4] J. B. Tenenbaum, V. de Silva, J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction", Science 290 (5500), pp 2319-2323, 2000

[5] J. Salamon, C. Jacoby, J. P. Bello, "A Dataset and Taxonomy for Urban Sound Research", 22nd ACM International Conference on Multimedia, Orlando USA, 2014.

[6] K. Kim et al., "Clustering Music by Genres Using Supervised and Unsupervised Algorithms", 2015

[7] L. Fedden, "Comparative Audio Analysis With Wavenet, MFCCs, UMAP, t-SNE and PCA", `https://medium.com/@LeonFedden/comparative-audio-analysis-with-wavenet-mfccs-umap-t-sne-and-pca-cb8237bfce2f`, 2017

[8] L. Hantrakul, A. Sarwate, "klustr: a Tool for Dimensionality Reduction and Visualization of Large Audio Datasets", `https://github.com/lamtharnhantrakul/klustr/blob/master/paper/klustr_final.pdf`, 2017

[9] L. van der Maaten, G. Hinton, "Visualizing Data using t-SNE", Journal of Machine Learning Research 9, pp 2579-2605, 2008

[10] P. Mermelstein, "Distance Measures for Speech Recognition, Psychological and Instrumental", Pattern Recognition and Artificial Intelligence, pp. 374–388., 1976

[11] S. Dupont et al., "Nonlinear Dimensionality Reduction Approaches Applied to Music and Textural Sounds", IEEE International Conference on Multimedia and Expo (ICME), 2013

[12] T. Kohonen, "Self-Organized Formation of Topologically Correct Feature Maps", Biological Cybernetics 43, p. 59-69, 1982

[13] Z. Shuyang et al., "Active Learning for Sound Event Classification by Clustering Unlabeled Data", International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017