Interactive Exploration of Musical Space with Parametric t-SNE

Conference Paper · July 2018

CITATIONS
0

READS
18

4 authors, including:

Matteo Lionello
Aalborg University
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Hendrik Purwins
Aalborg University
59 PUBLICATIONS **368** CITATIONS
SEE PROFILE

Mohamed Abou-Zleikha
University College Dublin
23 PUBLICATIONS **32** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Improve View project

Interactive Exploration of Musical Space with Parametric t-SNE

Matteo Lionello[1], **Luca Pietrogrande**[1,2], **Hendrik Purwins**[1], **Mohamed Abou-Zleikha**[3,4]
[1] Audio Analysis Lab, Aalborg University Copenhagen,
mlione16@student.aau.dk, hpu@create.aau.dk
[2] University of Padova, luca.pietrogrande.1@studenti.unipd.it
[3] Moodagent, maz@syntonetic.com, [4] Entrepreneur First

ABSTRACT

This paper presents a user interface for the exploration of music libraries based on parametric t-SNE. Each song in the music library is represented as a stack of 34-dimensional vectors containing features related to genres, emotions and other musical characteristics. Parametric t-SNE is used to construct a model that extracts a pair of coordinates from these features for each song, preserving similarity relations between songs in the high dimensional-feature space and their projection in a two-dimensional space. The two-dimensional output of the model will be used for projecting and rendering a song catalogue onto a plane. We have investigated different models, which have been obtained by using different structures of hidden layers, pre-training technique, features selection, and data pre-processing.
These results are an extension of a previous project published by Moodagent Company, which show that the clustering adaptation of genres and emotions, that is obtained by using parametric t-SNE, is by far more accurate than the previous methods based on PCA. Finally, our study proposes a visual representation of the resulting model. The model has been used to build a music-space of 20000 songs, visually rendered for browser interaction. This provides the user with a certain degree of freedom to explore the music-space by changing the highlighted features and it offers an immersive experience for music exploration and playlist generation.

1. INTRODUCTION

When dealing with large and complex datasets, dimensionality reduction techniques are used to extract information that is most relevant to describe each record, in order to facilitate its interpretation by machines or humans.

If, starting from a large song dataset, we want to create a catalogue of songs sharing perceptual properties, we need to decide which are the features characterizing the playlist.

While facing this topic we rely on heterogenous criteria. Most of these criteria are inherent in our mind, depending on musical preferences, culture, mood, feeling and experience. It is not straight-forward to detect exactly which features lead us to different decisions.

Thanks to worldwide streaming services (such as Youtube, Spotify, Deezer, Last.fm, Pandora Radio, Spotify, and so on) playlist creation is an activity that is largely spread among internet users. How to visualize songs on a two-dimensional plane, when each song is characterized by a high-dimensional feature vector? Machine learning approaches offer different solutions to this problem: feature selection approaches and feature extraction approaches.
Both methods have some advantages. Feature selection individuates a subset of the original features, selecting the features that bring most information [1]. Features extraction methods create new features by combining the original features. By creating new features starting from the cross-information stored in the original set of features, it is possible to optimize the description of a problem in a better way despite of the loss of information initially provided.

Dimensional reduction techniques provide a large set of tools aiming at redistributing a certain amount of data according to specific rules. T-Distributed Stochastic Neighbor Embedding (abbreviated as t-SNE [2]) shows how to preserve the distribution of the elements in a high dimensional space, into a subspace with 2 or 3 dimensions (the latent space), trying to conserve the pairwise distances between elements in the two different spaces. Because of the limitations given by the time and memory resources (see section 3) required by the original algorithm, the initial version of the t-SNE algorithm was limited only to datasets with a constrained number of samples. It did not aim at real time predictions and it was used for a one time only set prediction. The original non-parametric version of t-SNE doesn't produce a parametric mapping between the feature and latent space, generating the latent distribution but not providing a rule to project new points.

Starting from a previous work [3], the current project aims to investigate to which extent a parametric t-SNE [4] based method 1) could be efficiently used for the projection of songs onto a bi-dimensional space, and 2) it improves the clustering of songs sharing similar shades of genres and emotions with respect to a projection using Principal Component Analysis (PCA). The choice of using PCA as term of comparison is motivated by its usage in the previous work and the positive feedback received from the users, who used the corresponding application for playlist generation.

The neural network through which the parametric t-SNE is built, aims at creating a model which extracts 2 features

from the 34, maintaining the same distribution in the two spaces, one dictated by the original feature space and the second one given by the obtained bi-dimensional space. Moreover, this work aims at investigating how to give the capability to the user to explore a large music track dataset, to reach distinct, hidden areas and, in this way, present its content from different perspectives. Particularly, the project is focused on the two main classes of features that are considered to be the most important for the users. The two classes are: one class refers to the emotions aroused in the listener during listening, another relates to the shades of different genres perceivable in each song. In order to build an interface that is both reachable and easily usable by many users, the rendering of the obtained music-space has been concretely developed through a web browser interface (described in section 6).

2. RELATED WORKS

Playlist generation is an open research area. Features extrapolated from audio, are processed usually with feature extraction algorithms such as PCA for creating playlist from acoustical characteristics [5] and playlist ordering tools [6]. The importance of moods have also been investigated in playlist generation, showing that user's decision are strongly influenced by positive moods [7]. Music recommendation has been based on the predicted emotion of the user [8].

The visualisation of music collections has been previously investigated using a variety of dimension-reduction algorithms and graphical rendering methods. Self - Organizing Feature Maps (SOMs) [9] and Correspondence Analysis [10] have been employed to display key relations between the fugues in Bach's Well-Tempered Clavier II [11–14]. The visualisation of inter-relations between key modes, composers and pieces has been performed on collections of preludes for piano by Bach, Chopin, Alkan, Scriabin, Shostakovich, and Hindemith [15], using Isomap [16].

PlaySOM and **PocketSOMPlayer** [17] are 2 music interfaces displayed as grids. A cluster of songs is collected in each grid-cell, and the number of songs in each cell is displayed. A SOM maps rhythmical patterns on a plane. A gradient field of the features is superimposed on the grid allowing the user an easy exploration of the dataset and enabeling them to draw a path across the space for playlist generation.

Island of Music [18] arranges a set of songs in "islands" according to a psychoacoustic model that extracts features related to the genre from raw audio data. The psychoacoustic model consists of a SOM trained on rhythms patterns and specific loudness sensations both extracted from 359 popular songs.

Finally, this project has been motivated also by the project [3] conducted by the **Moodagent** company. Using a subset of a 50M collection, a 2D music map rendering on a plane a dataset composed by 20000 songs was developed. Using PCA, SOM and t-SNE based dimensional reduction approaches, the map shows highlighted clouds of points, whose density depends on the k parameter of a k-nearest neighbor, k-NN, algorithm where k depends on the level of zoom applied by the user. The interface offers also two playlist generation tools (based on k-NN algorithms) and paths to be drawn by the user.

3. DATA AND METHODS

3.1 OVERVIEW ON PARAMETRIC t-SNE

Dimension reduction algorithms allow to describe datasets composed by complex objects including a large amount of features [19]. This can be useful for decreasing the amount of memory needed to store data and the computational cost needed to process them and, in general, for visualization purposes.

Given a dataset X of N samples, where each sample belongs to a space $X^* \subset R^d$ (R^d being the real coordinate space of dimension d), X can be represented by a matrix $d \times N$. Dimension reduction aims at projecting the dataset from its original dimension to a subspace with smaller dimensions $d^* < d$, preserving some general properties and/or description of the data.

T-SNE: The t-SNE algorithm has been introduced by Laurens van der Maaten and Geoffrey Hinton in 2008, aiming at offering an optimized version of the Stochastic Neighbor Embedding [20] (SNE) algorithm. Similarly to SNE, T-SNE dimension reduction algorithm aims at preserving the original distribution of a set of data-points into a target space. It projects the data from a multi- dimensional space, to a tri or bi-dimensional space (called the latent space) by means of Gaussian distributions of neighbours. It minimizes the Kullback-Leibler divergence between the distances of all data-points in the original space and of their projection into the latent space.

Parametric t-SNE: In order to reduce the amount of memory, storage and time requirements, Laurens van der Maaten introduced a parametric version of the algorithm [4] where the t-SNE projection is computed by a Deep Neural Network $F_{net} : X \rightarrow Y$, projecting the data from the feature-space X into the latent space Y. The data used in the training set is divided in batches shuffling at each iteration their content in order to avoid over fitting. Parametric t-SNE is computationally cheaper than the original. Computing pairwise conditional probabilities presents quadratic complexity for both memory usage and time. In its original version, T-SNE does not create a model that can be used to predict data and so all the data-points are distributed over the space depending on their pairwise distances. Moreover, depending on the random variables that are computed, the distribution changes because of the lack of a non-convex objective functions and of the random initial state of the gradient descent. Beside the low computational cost the parametric version of t-SNE, offers a training set of data-points. The model representation is unique, so it does not depend on the amount and coordinates of the data used for the test set. A third advantage is that only one prediction by the neural network is required to project a new datapoint into the latent space, instead of computing every time all distances among all data-points in the two spaces.

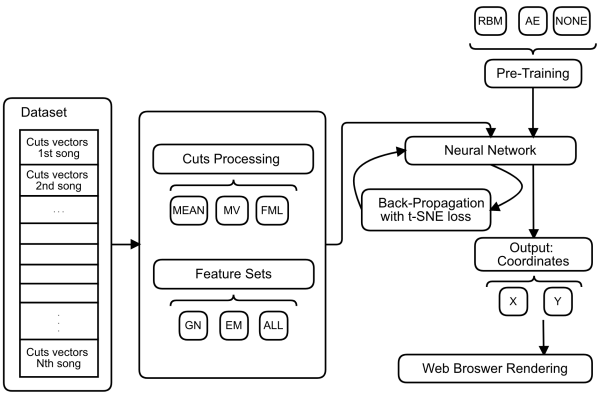

Figure 1: Flow chart of the work starting from the dataset storing the stack of cuts representing each song, till the rendering.

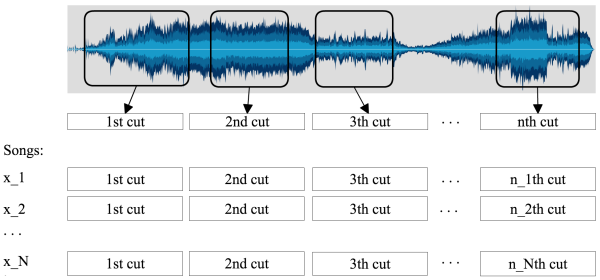

Figure 2: Representation of 'cuts' - 34 dimension vectors - extrapolating sequential information from the songs database composed by N songs; the amount of cuts for each song (n_i with $i = 1 \dots N$) depends on the length of each audio track

3.2 EXPERIMENTS

A flow chart describing the different steps of our work is shown in Fig. 1. The dataset used for the experiment has been provided by the Moodagent Company. This dataset includes N songs, each of them described by $K_{i=1:N}$ vectors called 'cuts' in this study, see Fig. 2. These vectors represent short sections in the song that are sequentially extracted from it, aiming at summarising its characteristics. Each song, depending on its length, has a different number of cuts and they are sorted according to the time within the song at which they were extracted. Each cut is described by $d = 34$ features. Beside the 34-dimensional feature vectors, each song is also provided with the following labels: genre, title, artist/band name, and the Spotify id.

The 34 features describing each cut, are composed of 6 features related to perceived moods, 17 features related to the perceived genres, and 11 features representing a set of other musical characteristics. The 6 mood features are 'Angry', 'Fear', 'Erotic', 'Joy', 'Tender', and 'Sad', where these emotions cover the circumplex model of affect [21] in clock-wise order with the first four emotions ('Angry', 'Fear', 'Erotic', 'Joy') covering the upper half (high arousal) and the last two ('Tender', and 'Sad') covering the lower half (low arousal) of the circumplex model. In an attempt to cover major popular music styles, the 17 genre features are the following: 'Blues', 'Country', 'Easy Listening', 'Electronica', 'Folk', 'Hip Hop Urban', 'Jazz' , 'Latin', 'New Age', 'Pop', 'R 'n' B Soul', 'Rock', 'Gospel', 'Reggae', 'World', 'Composition Style Era' and 'Classical Ensemble'. In our experiments, we will use the entire feature set of 34 features ('ALL'), the subset of the 6 mood features only ('EM'), and the subset of the 17 genre features only ('GN'). The mood and genre features are calculated by a perceptually validated computer model. Those features have discrete values ranging from 1 to 7, a 1 indicating the absence of that genre or emotion and a 7 its full presence.

Because of the different amount of vectors that describe each song, one main point of this work consisted in understanding how to manage this dissimilar description of datasets.

To process the data, we compared three methods. The first method consists in representing each song with the mean features of its cuts. The second method concatenates the mean vector with a vector of variances of the cut features. The third method was motivated by the large differences of the first and the last part of the songs with respect to their central part. In order to maintain this distinction, for each song, we concatenated the first cut, the last cut, the mean of the features in the cuts in the central part, and the respective variances. Two stacks of models were pre-trained by using Autoencoders [22] and Restricted Boltzmann Machine [23] while a third set of models was processed with no pre-training. Thus, the developed procedure allows for 3 degrees of freedom regarding:

- The pre-processing method applied to the samples (using the means of the features over all the cuts 'MEAN', using the concatenation of their means and their variances 'MV', and concatenating the first cut extracted from the first minutes of the songs, the last cut, representing the last minutes of the songs, and the mean and the variances of the cuts in the central part 'FML'),

- The type of pre-training used (no pre-training 'NONE', Auto-Encoders 'AE', and Restricted Boltzmann Machine 'RBM'),

- Feature sets (using all the features available 'ALL', using only emotion-related features 'EM', using only genre-related features 'GN').

In total we trained 27 different models whose results will be discussed in the next section. The feature selection was made in order to obtain two main different models, one based on emotion, and a second based on genre. Each of them were constructed to optimize the clustering of the respective features. The dataset has been standardised and normalised for each of its feature.

The parametric t-SNE has been implemented with a neural network provided with the loss function described in the original paper. The neural network is built as an autoencoder [24] which is fine-tuned by means of the t-SNE

loss. The implementation of the neural network was obtained by using Keras [25] a library for Python [26] running Tensorflow [27]. The neural network used was composed of 5 layers. The input layers's shape depends on the dataset structure and the data pre-processing method chosen. The output layer dimension is fixed to 2, as we need a bi-dimensional representation of the data. The dimension of the 3 internal layers have been respectively fixed to 80, 80 and 500 neurons. This architecture choice is motivated by the similar choice made in the original paper [4], scaled according to the input dimension. Different other architectures (500-500-2000, 120-120-2000, 80-80-2000, 40-40-2000, 500-500-1000, 120-120-1000, 80-80-1000, 40-40-1000, 120-120-800, 80-80-800, 40-40-800, 120-120-600, 80-80-600, 40-40-500, 80-80-300, 120-120-500, 120-120-80, 120-80-30) were tried but without achieving good results during the first 50 epochs. To prevent over-fitting problems, a dropout layer has been placed after the largest layer, using a drop rate of 0.1. The choice of optimiser to be used fell on Adam [28]. During the training, the probabilistic representation of the data is used to calculate the pairwise distances between data-points inside the same batch and the probability distribution is then calculated. Because of that, in order to not keep the model working on the same partition of the dataset, we introduced a shuffling into the training set. Doing so the data-points within a batch, change epoch by epoch, maintaining a certain degree of richness of the model during the training. Each layer of the pre-trained models, have been pre-trained by using for the first stack of models a Restricted Boltzmann Machine and a stack of Auto Encoders for the second.

4. RESULTS

To assess to what degree t-SNE preserves the proximity relations between genre and emotion attributes of our data across the dimension reduction, we will perform a 1-Nearest Neighbor (1-NN) classification on the original input space and on the projected two-dimensional target space. Thus, we will determine the classification accuracy degradation between the two representations, comparing different feature sets ('ALL', 'EM', 'GN'), the three cuts preprocessing methods ('MEAN', 'MV', 'FML'), the three initialisation methods ('AE', 'RBM', 'NONE'), and the two methods of dimension reduction (PCA and t-SNE).

Results:

Evaluation	Mean	Variance	Best Result	Best Model
Genres accuracy[1]	0.3087	0.0397	0.3496	{MV, AE, GN}
Emotion recall[2]	0.6494	0.0730	0.7649	{FML, AE, EM}
Genre recall[2]	0.3779	0.0669	0.4716	{MEAN, NONE, GN}

Table 1: The second and third columns show the means and the variances across all the different models for each kind of evaluation. The fourth column shows the best results obtained for each of the different evaluations and the corresponding model is explicit in the last column. [1] calculated for the genre ground-truth, [2] calculated for the perceived features

The models have been trained using 70000 samples for the training set collected in batches of 5000 samples each and iterated for 70 epochs. The testing set was composed of 30000 samples. Before the pre-training, from the 34 features describing each cut, we selected feature subsets, i.e. the subset consisting of the 6 emotion features ('EM') consisting of 'Angry', 'Fear', 'Erotic', 'Joy', 'Tender', and 'Sad' and the subset of the 17 genre features.

4.1 Models Performance

To evaluate how well songs showing similar features were distributed close to each other, we assigned each song to multiple classes. Each class refers to one among the 6 moods and 17 genres. After the processing of the cuts, each song has been assigned to a class if the average among the cuts of the feature referring to that class, was strictly greater than 5. This threshold has been decided according to the observations on the distribution of the data-set. For each of these classes, subsets, the confusion matrix of the 1-NN classifier and the recall were calculated. The usage of the recall is motivated by the palylist generation goal of our application, aiming at gathering as many similar songs together as possible in despite of presence of false positive. The average among the recall coefficients for each model was calculated, weighted by the size of the classes. Another measure of evaluation we used, consists in the 1-NN accuracy computed for the ground-truth genres given for each song. Here follow the results for the best models obtained for each different type of evaluation (genres accuracy, emotion recalls and genre recalls) from the 1-NN classification of the 27 models.

The configuration of each model is denoted by the following notation with curly brackets: {type of pre-processing, type of pre-training, features }. As shown in table 1, the best accuracy obtained for the genre ground-truth is given by the model {MV, AE, GN}, while the best recall for emotion using both the features intensity and number of element as weights, refers to the model {FML, AE, EM} and the best recall for genres using both the features intensity and number of element as weights belongs to the model {FML, AE, EM} as reported in the table 1. Including all the features in the feature test the best model for mood evaluation is provided by {MEAN, RBM, ALL}, whose recall for the mood is equal to 0.6691, and has genres recall equal to 0.3508. Similarly, using all the features the best model respect to the genres is given by {FML, RBM, ALL}, with recall fro the genres equal to 0.3941, and recalls for mood 0.5742. Comparing to the table, each of these two models has the recall for its secondary feature lower than their mean across all the models. This suggest not an good usage of one of these model to obtain a representation sharing good results for both moods and genre feature sets.

4.2 Dimensionality Reduction Performance

Because of the aim of the parametric t-SNE algorithm, the quality of the recall coefficients obtained applying the 1-NN classifier must be compared to how well the data-points are distributed in the original high dimensional space by using the same evaluation strategy we used to evaluate our models. By applying 1-NN to the original dataset processed with {FML, EM} in the high dimensional space, we obtained a recall coefficient equal to **0.7699** while processing the dataset with {MEAN, GN} the recall is **0.5564**. By comparison with the table 1, the model {FML, AE, EM} deviates by the 0.65% from the original data-space, while the model {MEAN, NONE, GN} organizes the data-points even better than how they are in the original set, with a deviation equal to the 15.25% as shown in table 2. The high deviation error recorded for model based on genre features suggests a dissimilarity in the distribution in the high dimensional space respect to emotions and the genres descriptors and a lower figure of accumulation for the genre features. Running the same 1-NN evaluation on the output from a Principal Component Analysis (PCA) dimension reduction technique, by processing the dataset with {FML, EM} the recall for the emotions based class deviates by the 34.85% from the distribution in the high dimensional space, and by processing the cuts with {MEAN, GN} the deviation error rises to 9.69%.

5. THE APPLICATION

To exemplify the potential our dimension reduction approach, we present an application prototype for music exploration. [1] To render the model we obtained, we designed a client interface for music exploration that is shown in Fig. 3.

The ideas behind the interface: We designed our interface in order to offer the following functionalities:

- Moving in a 2D plane, with zoom-in and -out to access songs details.
- Mixing some graphical characteristics (e.g color, size) to simultaneously see more than one feature.
- Streaming a song while navigating.
- Filtering the dataset according to either artist name, genre or a keyword.
- Creating a playlist that evolves in genre and mood according to the user preferences.

[1] Video: `https://vimeo.com/273294798`

Error deviations:

Model reference	Deviation error
Param t-SNE {MEAN, NONE, GN}	15.25%
PCA {MEAN, GN}	34.85%
Param t-SNE {FML, AE, EM}	0.65%
PCA {FML, EM}	9.69%

Table 2: Error deviation between the recalls coefficients describing the distribution over the original high dimensional space and the latent bi-dimensional space obtained with t-SNE and PCA

The web browser: In addition to the previously mentioned features, the application should comply with the following requirements:

- Cross-platform software: The implementation should be independent of any particular operating system.
- Portability: the interface should be accessible by as many devices as possible, including mobile ones.

For the design of the website we used CreateJS, Bootstrap and Jquery. Furthermore we utilised HTML5 and CSS3, to build the interface, Javascript, to interact with the user, and PHP, to interact with the dataset.

Mapping features to color mixtures and dot sizes: Our interface should convey as many features as possible, while maintaining simplicity. Each song is represented by a dot. Features of the song are mapped to the three basic colour components and the diametre of the dot. This design decision visualizes the relations between the features of a song as a mixture of the corresponding basic colours. As an example, a user could associate three different genre freatures to the three basic colors and a mood feature to size. A simple approach would have been to associate the three features to red, green and blue, consequently using RGB, the standard color model for CSS3. However, RGB as an additive color model, does not reflect our idea of color mixing. For instance, when summing up all the components of the color model, white is obtained. A more suitable color model is CMYK (i.e. Cyan, Magenta, Yellow, black Key), a subtractive colour model. In such a model, the color components are mixed in the same way as the human eye perceives them. Mixing all colors for example yields black. For this reason, we used the CMYK color model.

Music streaming: It was a needed requirement to let the user listen to the songs of the interface. Therefore, each song in the dataset is stored along with its Spotify ID. This allowed to incorporate a Spotify iframe element, to directly stream the song from the website. Unfortunately, the iframe element by Spotify doesnt provide any API, neither to change song nor to manage events related to it. Hence, forward and backward buttons have been also placed, to randomly shuffle among the displayed songs.

Playlist creation: To let the interface become richer, it was intuitive to let the user draw a line on the screen, to connect songs adding them to a playlist. Doing so, the resulting playlist can start from certain moods and end with others, following the path that the user prefers. Pressing the pencil button, the user enters in playlist mode and can connect several dots in order to create a playlist. Once a playlist is created, a modal window appears, showing the

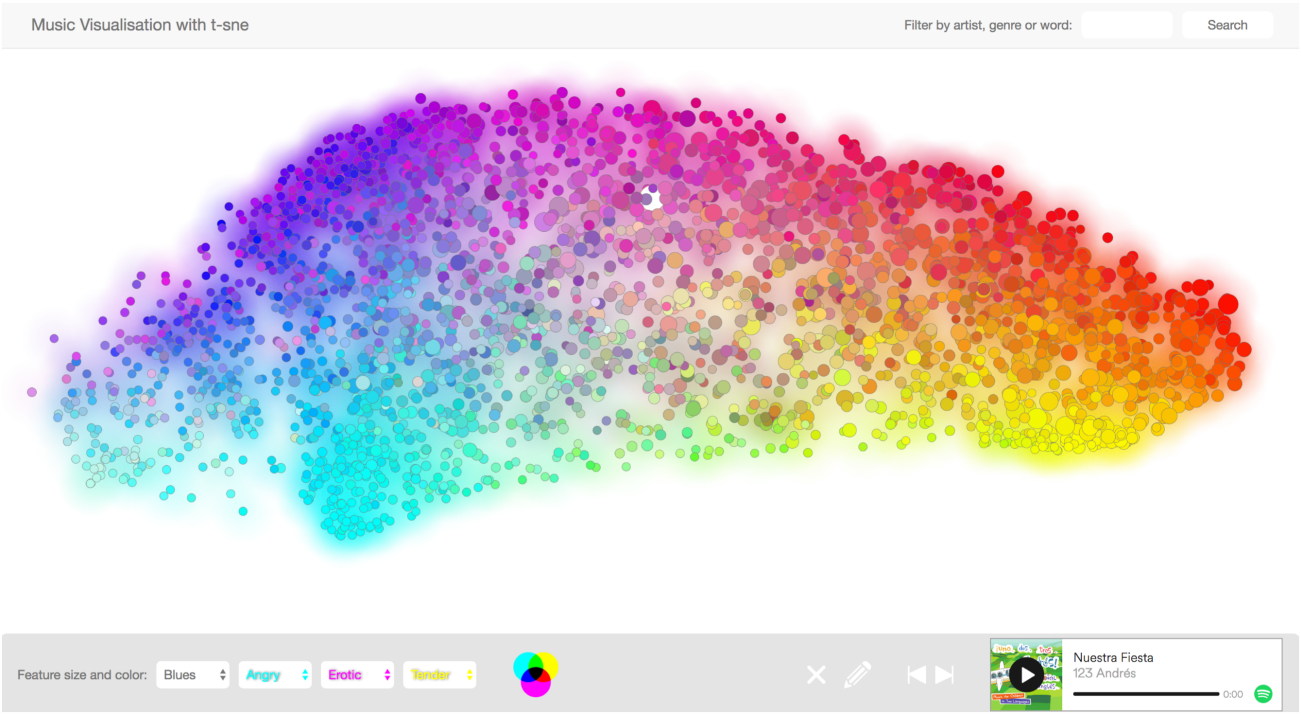

Figure 3: Rendering of 3000 songs-catalogue through the model {FML, AE, EM}. Each coloured point is the representation of a song. The location of each song is determined by the output of the parametric t-SNE model. The colour of each point is obtained by mixing the total information of three features, which set can be selected by the user. The size corresponds to what degree the songs belong to a genre, to be selected by the user.

songs of the playlist and the playlist is also drawn as a dashed line connecting the related dot. When a playlist is drawn upon the canvas, it is possible to navigate between its tracks with the two shuffle buttons. In the end, the user can clear a drawn playlist with the cross button.

Search field: In the end, a search field has also been introduced. It allows to filter the dataset and look for a specific artist, genre or keyword. In order to let the user look for something that is actually present in the dataset, the search field has been provided with an auto-complete function, implemented in jQuery.

5.1 INFORMAL USABILITY TEST OF THE INTERFACE

After having implemented a first version of the interface, an informal interview has been set up, to rate the consistency of the designing choices. To do this, we set up a usability test environment where several users have been asked to try the interface in different conditions and answer some questions about their impressions.

A summary of the usability test: Different experiments have been conducted with 27 participants, that were considered a representative sample of the people who could use the interface as a final product. They were both students and professionals, aged between 20 and 27 years old.They were told to test our interface through these steps:

- The testers were told to use the interface for two minutes and try to understand the various functionalities.
- The testers were shown a picture with explanation notes about the various functionalities. Afterwards, they were asked to try them out again on the interface.
- The testers were asked to explore the dataset in four different configurations. Each one gave the user a different number of controllable features: 1, 2, 3 and 4.

Consideration about the results The overall intuitiveness of the interface has been rated well, according to our expectations, even though, it still needs be improved, especially concerning the playlist drawing and clearing mechanism. Even after an explanation picture had been shown, still 40.7% of the testers claimed not to have completely understood how to draw and clear a playlist. A confirmation, instead, has been received in relation to the choice of the CMYK color model, since most the users appreciated how the colors were combined. Another interesting result referred to the number of controllable features. The configurations with just one or two controllable features met a good response among the users who stated they would make use of them quite often, while it seems clear enough, again from the users answers, that there is no need for a 5th feature, at least with the current interface settings. The design of a rather simple interface, appeared to be a good choice, since most of users found it sufficiently complex.

6. CONCLUSION

In this study, a parametric t-SNE model for music playlist generation is introduced. Beside the computational advantages of using parametric t-SNE, this representation preserves the structure of the high dimensional space in the latent space within a small error range regarding the models based on emotion features. Moreover, this approach provides a function which maps the data from the feature space to the latent space, by preserving a similar distribution of the data in the two spaces. Our project reaches good results compared to the previous methods based on PCA, showing that the parametric t-SNE method is able to represent high-dimensional data, better than PCA, and offers a more suitable strategy in music dataset context.

The small variance and gap between the models and best results obtained for emotion-related features, show that the combination of applied strategies has been successfully confirmed. Further, comparing model and best results on genres does not show a optimum choice among combination of pre-training type and pre-processing strategies, suggesting that in similar experiments all should be implemented and compared to each other.

Furthermore, our work presents an original interactive user interface visually rendering the songs projected from the feature space over a plane. This allows a user interaction with the original dataset, rendering songs in a browser with similar characteristic close to each other. This provides the user with playlist generation tools and a link with Spotify to listen to them directly.

However, different aspects must be further analyzed and developed in future work: the high difference between the emotion evaluation of the model with respect to the genre evaluation, and the implementation of an interface which could easily jump for the same data-points between the model given by emotion and the model given by genre.

7. REFERENCES

[1] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Icml*, vol. 97, 1997, pp. 412–420.

[2] L. van der Maaten and G. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, no. 9, pp. 2579–2605, Nov. 2008.

[3] B. Vad, J. W. Daniel Boland, R. Murray-Smith, and P. B. Steffensen, "Design and evaluation of a probabilistic music projection interface," in *16th International Society for Music Information Retrieval Conference*, 2015.

[4] L. van der Maaten, "Learning a parametric embedding by preserving local structure," in *In Proceedings of the Twelfth International Conference on Artificial Intelligence Statistics*, vol. 5, Nov. 2009, pp. 384–391.

[5] D. Hughes and B. Manaris, "Fractal dimensions of music and automatic playlist generation: Similarity search via mp3 song uploads," in *2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, July 2012, pp. 436–440.

[6] T. Nakano, J. Kato, M. Hamasaki, and M. Goto, "Playlistplayer: An interface using multiple criteria to change the playback order of a music playlist," in *Proceedings of the 21st International Conference on Intelligent User Interfaces*, ser. IUI '16. New York, NY, USA: ACM, 2016, pp. 186–190. [Online]. Available: http://doi.acm.org.zorac.aub.aau.dk/10.1145/2856767.2856809

[7] M. Pichl, E. Zangerle, and G. Specht, "Understanding playlist creation on music streaming platforms," in *2016 IEEE International Symposium on Multimedia (ISM)*, Dec 2016, pp. 475–480.

[8] J. J. Deng and C. Leung, "Emotion-based music recommendation using audio features and user playlist," in *2012 6th International Conference on New Trends in Information Science, Service Science and Data Mining (ISSDM2012)*, Oct 2012, pp. 796–801.

[9] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep 1990.

[10] M. J. Greenacre, *Theory and Applications of Correspondence Analysis*. Academic Press, 1984.

[11] H. Purwins, B. Blankertz, and K. Obermayer, "A new method for tracking modulations in tonal music in audio data format," in *Int. Joint Conf. on Neural Network (IJCNN'00)*, vol. 6, 2000, pp. 270–275.

[12] H. Purwins, T. Graepel, B. Blankertz, and K. Obermayer, "Correspondence analysis for visualizing interplay of pitch class, key, and composer," in *Perspectives in Mathematical and Computational Music Theory*, ser. Osnabrück Series on Music and Computation. Electronic Publishing Osnabrück, 2004, pp. 432–454.

[13] H. Purwins, "Profiles of pitch classes circularity of relative pitch and key- experiments, models, computational music analysis, and perspectives," 2005.

[14] H. Purwins, B. Blankertz, and K. Obermayer, "Toroidal models in tonal theory and pitch-class analysis," in *Computing in Musicology*, W. B. Hewlett and E. Selfridge-Field, Eds. Center for Computer Assisted Research in the Humanities and MIT Press, 2008, vol. 15.

[15] H. Purwins, B. Blankertz, G. Dornhege, and K. Obermayer, "Scale degree profiles from audio investigated with machine learning techniques," in *Audio Engineering Soc. 116th Convention*, 2004.

[16] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[17] M. D. Robert Neumayer and A. Rauber, "Playsom and pocketsomplayer, alternative interfaces to large music collections," in *ISMIR*, 2005, p. 618623.

[18] E. Pampalk, A. Rauber, and D. Merkl, "Content-based organization and visualization of music archives." in *10th ACM Int. Conf. on Multimedia*, 2002, pp. 570–579.

[19] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, pp. 66–71, 2009.

[20] G. Hinton and S. Roweis, "Stochastic neighbor embedding," *Advances in Neural Information Processing Systems*, vol. 15, p. 833840, 2002.

[21] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980.

[22] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[23] G. E. Hinton, "A practical guide to training restricted boltzmann machines," 08 2010.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] F. Chollet. (2015) Keras. [Online]. Available: github.com/fchollet/keras

[26] G. van Rossum, "Python tutorial," 05 1995.

[27] M. A. et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, 2015.