Nonlinear Dimensionality Reduction for Music Feature Extraction

Charles Lo
Department of Electrical and Computer Engineering
University of Toronto
Toronto, ON, Canada M5S 3G4
locharl1@eecg.toronto.edu

Abstract

In recent years, a number of interesting nonlinear dimensionality reduction techniques have been developed. In this paper, LLE, multilayer autoencoders and parametric t-SNE are applied to the problem of music feature extraction. Particularly, these methods are used to compress a high-dimensional set of music features into a low-dimensional representation which may be stored with each audio file in a large music collection. The resultant features are evaluated on music genre classification tests with Gaussian Mixture Model clustering as well as supervised K-Nearest Neighbour classification. It is found that of the methods tested, parametric t-SNE is most able to maintain the local structure of the features under heavy compression.

1 Introduction

The advent of digital media has allowed for the creation of large collections of music. Searching through these collections is typically facilitated by the use of artist, album and genre tags. However, since these tags are generated by hand, a great deal of audio content may be unlabelled or poorly labelled. An alternative to tag based methods is the use of content-based information retrieval in which an example song is used to find a set of acoustically similar songs in the database. A major component of such systems is a set of low-dimensional features which characterize the acoustic properties of the music.

The problem of feature extraction for audio has been a topic of recent interest in music information retrieval [1, 2]. Features for timbre, pitch and rhythm are typically gathered for a number of short time frames of 20-50ms. These frame-level features may then be aggregated over longer periods of time by averaging or fitting Gaussian Mixture Models (GMMs) across many frames which represents a form of dimensionality reduction. An alternative to manually selecting a particular subset of features is to use automatic dimensionality reduction to discern the most significant uncorrelated dimensions of acoustic similarity.

In this paper, several nonlinear unsupervised dimensionality techniques are examined to further reduce the dimensionality of the music features. In particular, several recently popular methods of dimensionality reduction will be tested: Locally Linear Embedding (LLE), Multilayer Autoencoders and Parametric t-Distributed Stochastic Neighbour Embedding (Parameteric t-SNE). Similar work has been done using convolutional deep belief networks [3], but that effort focused on generating new features from audio spectra while in this paper the dimensionality of existing audio features is reduced. In addition, a preliminary study using LLE has been performed [4] for visualizing changes in the power spectrum, but again not with the focus of reducing the dimensionality of song features.

2 Feature Selection

Timbral features are popular for music genre classification since they are very good at discriminating the instrumentation of a song. In this paper, the following timbral features were gathered for in 46.44ms frames with 50% time overlap between frames:

- *Mel-Frequency Cepstral Coefficients (MFCCs)*: MFCCs are perceptually motivated representations of the short-term power spectrum which have been used very successfully in speech recognition. They are generated by dividing the log power spectrum into a number of bands spaced using the Mel frequency scale. Coefficients are then created by dimensionality reduction through the discrete cosine transform. In this paper the first 13 MFCCs are used.
- *Zero-Crossing Rate (ZCR)*: The ZCR is another feature popular in speech recognition and is easy to compute. It is simply the rate of sign changes recorded during the time frame and may be used as a rough frequency estimate.
- *Spectral Centroid, Rolloff, Flux*: These three features describe the power spectrum shape. The spectral centroid is the center of mass of the spectrum, spectral rolloff is the frequency below which 85% of the spectrum lies and spectral flux is the variance of the spectrum from one frame to the next.

In addition to the timbral features, a set of chroma features were included in the feature vector which represent the scales most prevalent in the song. To generate the chroma, frequencies were mapped to one of the twelve semitones in the musical scale and the power for each was calculated over the time frame. The feature vector consisted of the ratio of the power of each semitone to the maximum as well as the average and maximum across all the semitones.

The combination of timbral and chroma features produced 31 features per frame. Aggregation of the frame-level features took place in two steps: First, the means and variances of the features were calculated across 40-frame windows to generate a set of 62 dimension feature vectors. Next, the means and variances of those feature vectors were accumulated across the entire song to generate a final 124 dimension feature vector. The open source Marsyas toolkit [5] was used for feature extraction.

3 Dimensonality Reduction Techniques

Dimensionality reduction involves the transformation of high-dimensional data into a low dimensional representation. One of the most popular methods of dimensionality reduction is Principal Components Analysis (PCA) in which the dimensions of maximum variance are taken to be most representative of the data. PCA is a linear method in that the mapping from the input to the low dimensional space is performed by a linear transformation. Thus, an underlying assumption in the application of PCA is that the data is best represented by a hyperplane in low dimensional space. This may not be true for complex sets of data and thus nonlinear methods will be the focus of this paper.

3.1 Locally Linear Embedding

Locally Linear Embedding (LLE) [6] is a method of nonlinear dimensionality reduction in which the assumption is made that the data lies on a locally linear, continuous manifold in low dimensional space. In other words, a data point may be linearly reconstructed by its K nearest neighbours. Thus, the first step in LLE is finding the K-nearest neighbours for every point in the data set. Next, a reconstruction matrix W is found from which every data point can be reconstructed given of its neighbours. This is equivalent to minimizing the cost function:

$$E = \sum_{i}^{N} (X_i - \sum_{j}^{N} W_{i,j} X_j)^2 \tag{1}$$

Where $W_{i,j} = 0$ for $i = j$ and any point not in the neighbour set. An additional constraint during this optimization is that $\sum_j W_{i,j} = 1$ which makes the solution invariant to translations of the point

X_i and its neighbours. In the dimensionality reduction step, a set of low dimensional vectors Y_i are found by again minimizing a linear reconstruction error:

$$\Phi = \sum_{i}^{N}(Y_i - \sum_{j}^{N} W_{i,j}Y_j)^2 \tag{2}$$

However, this optimization holds the weight values constant while changing the vectors Y. The resulting embeddings Y maintain the local structure of the data given by the weightings $W_{i,j}$. To solve this second optimization, the inner product $M = (I - W)^T(I - W)$ is calculated where I is the $N \times N$ identity matrix.The coordinates of the embeddings Y are the eigenvectors of M with the lowest non-zero eigenvalues.

The steps required in LLE are relatively fast, but the embedding is non-parametric since there is no direct mapping from the high dimensional space to the low dimensional embedding. Thus to calculate embeddings for new data, LLE must be run again on the entire data set, which for very large sets can be extremely time consuming. In [7], a method of estimating new embeddings was formulated using the Nyström approximation for eigenfunctions:

$$\phi(y) = \frac{1}{\lambda}\sum_{i=1}^{N} K(y, x_i)\phi(x_i) \tag{3}$$

where $K(x, y)$ is a kernel function and $\phi(x_i)$ are the eigenvectors of the matrix $A_{i,j} = K(x_i, x_j)$. For LLE, a kernel is first defined for the weight matrix $W_{i,j} = w(x_i, x_j)$:

$$C(y)_{i,j} = (y - x_i)(y - x_j)^T, C(y)_{i,j} = 0 \text{ if } x_i, x_j \notin N(y) \tag{4}$$

$$w(y, x_i) = \frac{\sum_q C^{-1}(y)_{i,q}}{\sum_{pq} C^{-1}(y)_{p,q}}, w(y, x_i) = 0 \text{ if } x_i \notin N(y) \tag{5}$$

$N(y)$ is the set of K points in the neighbourhood of y. Next, a matrix is defined $\tilde{M} = I - M$ which changes the problem to finding the eigenvectors with largest associated eigenvalues. This construction gives $K(y, x_i) = w(y, x_i)$ for a new out-of-sample data point y. This allows us to quickly estimate new data points provided LLE has been trained on a large enough sample of the dataset and have the rest of the data points available.

The one free parameter in LLE is the number of nearest neighbours K to use for linear reconstruction. To find a suitable value for K, values between 4 and 50 were tested and $K = 5$ was found to perform best on the K-Nearest Neighbour genre classification tasks.

3.2 Deep Autoencoder

An autoencoder is a neural network structure which uses a set of recognition weights to generate a low dimensional code layer and a mirrored set of generative weights that reconstruct the input from the codes. Thus, similar to PCA, a low-dimensional representation of the data is captured in the code layer which minimizes reconstruction error. In this paper a multilayer autoencoder was used in which the input nodes, output nodes and the code layer nodes were linear units with Gaussian noise while the internal hidden nodes were binary stochastic units. The connectivity was restricted such that each pair of node layers formed a fully connected bipartite graph and the input to a hidden node was the weighted sum of all the node states in the previous layer. Binary nodes states were activated using the sigmoid function: $\sigma(x) = \frac{1}{1+e^{-x}}$. To train the autoencoder, the squared reconstruction error was calculated by taking the difference between the input and output layers:

$$\phi = \frac{1}{N}\sum_{i=1}^{N}(d_i - x_i)^2 \tag{6}$$

The gradient of this error may then be backpropagated to all of the hidden layers to train the network. However, there are a number of problems using simple backpropagation to train the network. First, when starting with random weights, there may be many steps required to converge on an optimal solution. In addition, the solution that is found through gradient descent may be a poor local optima. Recently, Restricted Boltzmann Machines (RBMs) been used to efficiently pre-train autoencoders

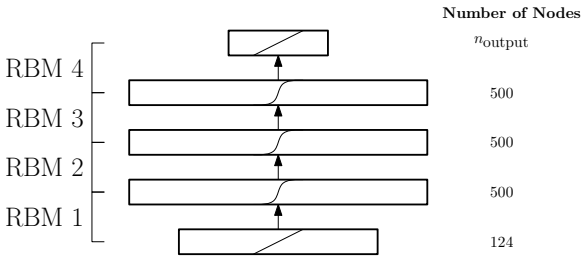

Figure 1: Structure of the first half (recognition section) of the deep autoencoder. This is also the structure of the deep belief network used for t-SNE

.

before backpropagation [8]. This is done by greedily training pairs of layers in the deep network. It has been found that this technique avoids local optima and provides a good starting point on which backpropagation may be used to fine-tune the parameters.

Motivated by the experiments in [8], an autoencoder with three hidden layers between the input and code layers was used. A diagram of the first half of the autoencoder is shown in Fig. 1. To determine the number of hidden nodes required, the code layer was set to 20 dimensions and a search was performed using different numbers of nodes in each layer. No significant improvement in reconstruction error was observed when using over 500 nodes per layer. During training, 50 epochs of pretraining and 50 epochs of fine-tuning were used. To estimate the number of epochs of fine-tuning, the training data was broken into a training and validation set and reconstruction error on the training and validation data was observed. The change in reconstruction error was not significant over after 50 epochs. In addition, no overfitting was observed even after 200 epochs.

3.3 Parametric t-Distributed Stochastic Neighbour Embedding

Parametric t-Distributed Stochastic Neighbour Embedding [9] (parametric t-SNE) is very similar to a multilayer autoencoder. However, a different cost function and only half of the network, up to the code layer, are used. The idea behind the t-SNE cost function is to maintain local structure in a probabilistic manner. First, a set of probabilities are built for both the high-dimensional input $\mathbf{x}$ and the low-dimensional output $\mathbf{y}$:

$$p_{j|i} = \frac{\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma_i^2)}, q_{ij} = \frac{\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2/\alpha)^{(\frac{-\alpha+1}{2})}}{\sum_{k\neq i}\exp(-||\mathbf{y}_i - \mathbf{y}_j||^2/\alpha)^{(\frac{-\alpha+1}{2})}} \tag{7}$$

The joint probability in the high dimensional space can be written as a symmetrized conditional probabilities $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2}$. These joint probabilities describe the similarity between data points i and j and $p_{ii} = q_{ii} = 0$. Thus, to maintain the local structure in low dimensional space, the new cost function is given by:

$$C = KL(P||Q) = \sum_{i\neq j} p_{ij}\log\frac{p_{ij}}{q_{ij}} \tag{8}$$

Once again the gradient may be found and backpropagation used to adjust the weight parameters after pretraining with RBMs. It should be noted that the distance in input space is represented by the probability under isotropic Gaussians while the distance in the output is represented by Student's t-distribution. The heavy tailed t-distribution helps emphasize local similarity and avoid crowding since a slightly distant pair of points in the input space can be much farther apart in the low dimensional space. The variance in the Gaussian is found by searching for a value which maintains a user-defined *perplexity* value:

$$\text{Perp}(P_i) = 2^{-\sum_j p_{j|i}\log_2 p_{j|i}} \tag{9}$$

To keep comparisons valid between parametric t-SNE and autoencoders, the same network structure (Fig. 1) and training epochs were used. Perplexity was chosen by running nonparametric t-SNE on a validation set with different values and choosing the one with the best K-NN genre classification performance. The final value for perplexity was 30. Finally, the free parameter α was chosen to be one less than the dimensionality of the output as suggested in [9].

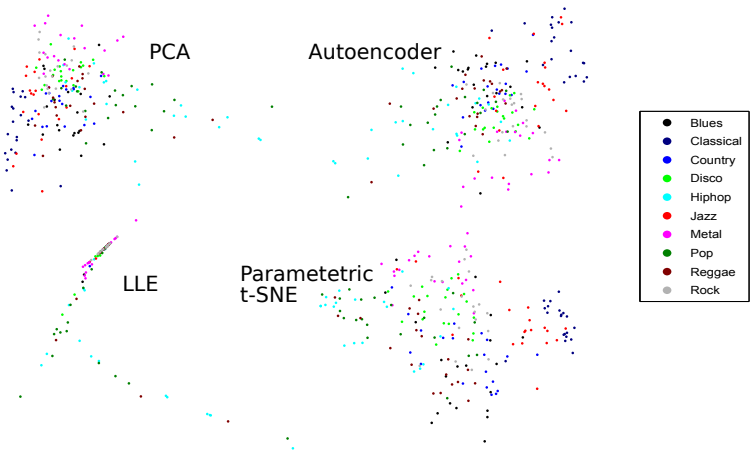

Figure 2: 2-D visualizations of the 200 held out data points using the different dimensionality reduction techniques.

4 Experiment Setup

The dataset used in the experiments was first introduced in [1]. It consists of 1000 song clips evenly distributed across 10 genres. Each clip is 30 seconds long and sampled monophonically at 22050 Hz. The 1000 songs were randomized and divided into a balanced training set of 800 songs and a test set of 200 songs. After feature extraction, the features were normalized to zero mean and unit variance to facilitate learning the Gaussian linear units in the neural network structures.

Acoustic similarity measures generally separate different genres into a number of clusters in feature space. Thus, the performance of the dimensionality reduction techniques was evaluated by testing how well the songs were clustered in the lower dimensional space. Unsupervised clustering via a Gaussian Mixture Model (GMM) was used to test general clustering performance. Ten Gaussians with diagonal covariance matrices were fitted to the training data, each cluster was assigned a genre class based on the majority genre under each Gaussian. The purity of each cluster was then evaluated on the held out data, where purity was the percentage of correctly classified songs. This test was repeated 10 times with different Gaussian fits and the mean purity was taken. In addition supervised genre classification was performed using K-Nearest Neighbours (K-NN). Both the dimensionality reduction techniques and the classifiers were trained on the training set then generalization performance was gathered on the held out data.

5 Results

It can be seen from Fig. 2 that for low dimensional outputs, parametric t-SNE has the best overall clustering with genres most clearly separated. In contrast, the LLE embeddings cluster largely about one point. This can be explained by observing that one solution to the cost function in Eqn. 2 is to cluster most of the points very close to each other and have a few points very far away to balance the linear reconstruction error. PCA and the deep autoencoder produce similar, partially clustered visualizations. During training, the autoencoder is looking to minimize reconstruction error, thus it makes intuitive sense that it finds dimensions of maximum variance akin to PCA.

The characteristics observed in the 2-D visualizations carry over to the K-Nearest Neighbour (K-NN) and Gaussian Mixture Model (GMM) results shown in Table 1. One of the first things to notice is the very poor classification performance of the Gaussian Mixture Model. This model assumes that the data was generated from a Gaussian mixture distribution, but as seen the visualization, the genres do not generally fall into simple Gaussian shapes. K-Nearest Neighbours on the other hand only assumes that the data is somewhat localized and thus performs better especially when genres are separated into many small clusters. Fig. 3 shows the K-NN classification accuracy for different numbers of neighbours after reduction to five dimensions. Again, parametric t-SNE has the best overall performance particularly at low dimensions. PCA and the autoencoder perform similarly

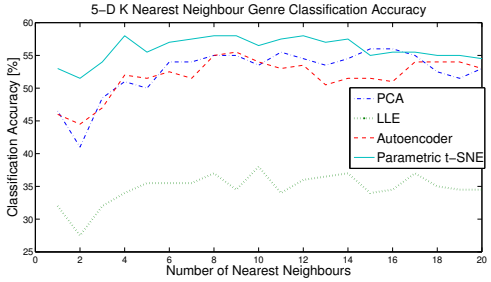

Figure 3: K-Nearest Neighbour Classification Accuracy on 5-Dimension Music Features

	2D 10-NN	5D 10-NN	20D 10-NN	2D GMM	5D GMM	20D GMM
PCA	36.5	53.5	58	7.15	11.7	8
LLE	31.5	38	41.5	11.55	8.85	9.25
Autoencoder	40.5	54	55.5	9.5	9.9	10.65
Parametric t-SNE	47	56.5	60.5	8.95	8.1	10.2

Table 1: Music Genre Classification Accuracy (10-NN) and Cluster Purity (GMM) results on the 200 held-out data points after reduction to 2, 5 and 20 dimensions.

while LLE has the worst performance due to its poor clustering. The performance of the original feature vector for 10-NN classification is 63%. Thus at 20 dimensions, parametric t-SNE is able to retain the vast majority of the local structure.

6 Conclusions

The results show that parametric t-SNE dimensionality reduction may be used to heavily compress music features while maintaining good local neighbourhood structure due to its heavy tailed distance measure in low-dimensional space. LLE recovers heavily compacted clusters so the performance on this feature extraction task is very poor. Finally the deep autoencoder performs roughly on par with the much simpler PCA, thus given its long training time it is also not well suited for this task.

References

[1] George Tzanetakis and Perry Cook. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speed and Audio Processing*, 10(5):293–302, July 2002.

[2] Martin McKinney and Jeroen Breebaart. Features for Audio and Music Classification. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 151–158, 2003.

[3] Honglak Lee, Yan Largman, Peter Pham, and Andrew Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems*, pages 1096–1104, 2010.

[4] Viren Jain and Lawrence Saul. Exploratory Analysis and Visualization of Speech and Music by Locally Linear Embedding. 3:984–987, 2008.

[5] George Tzanetakis and Perry Cook. Marsyas: A framework for audio analysis. *Organized Sound*, 4(30):169–175, 2000.

[6] Sam Roweis and Lawrence Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, 290(5500):2323–2326, 2000.

[7] Yoshua Bengio, Jean-François Paiement, and Pascal Vincent. Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering. In *Advances in Neural Information Processing Systems*, pages 177–184. MIT Press, 2004.

[8] Geoffrey Hinton and Ruslan Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[9] Laurens van der Maaten. Learning a Parametric Embedding by Preserving Local Structure. *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, 5:384–391, 2009.