Musical Information Retrieval: Genre Classification

Vicky Li, Sama Shrestha

December 5, 2016

Introduction

Given the song, what is the genre?

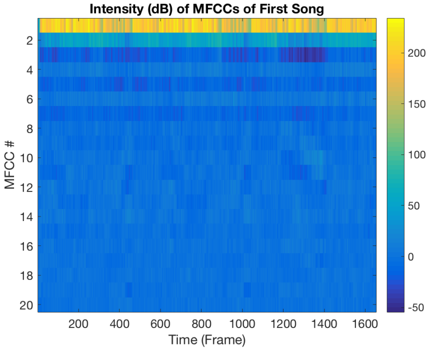
- ISMIR 2004 Audio Description Contest[1]
- 729 songs: classical (320), electronic (114), jazz/blues (26), metal/punk (45), rock/pop (102), world (122)
- mono channel WAV files, $f_s = 11025$ Hz

[1] Cano, P., Gómez, E. et al. (2006) ISMIR 2004 audio description contest (MTG-TR-2006-02). Music Technology Group, Universitat Pompeu Fabra.

Dimension Reduction: MFCC

1. Take first 1653 frames (256 samples each, 50% overlap) of middle 2 minutes of each song
2. Apply window function
3. Obtain intensity (dB) vs. frequency vs. time (frame):
4. Apply FFT $\rightarrow$ 129 frequency bins (Hz)
5. Apply filter bank $\rightarrow$ 40 frequency bands (Mel)
6. Apply DCT $\rightarrow$ 20 MFCCs

(Matlab ma toolbox)
MFCCs mimic the human ear



Further dimension reduction: FJLT

$$\mathbf{x_1}, ..., \mathbf{x_n} \,\epsilon\, \mathbb{R}^d \quad \xrightarrow{\text{Random Projection}} \quad PHD\mathbf{x_1}, ..., PHD\mathbf{x_n} \,\epsilon\, \mathbb{R}^k$$

- P is a $k \times d$ matrix with $P_{ij} \sim N(0, 1/q)$ with probability q and $P_{ij} = 0$ with probability $1 - q$, $q = \min\{\Theta(\frac{\varepsilon^{p-2}(\log n)^p}{d}), 1\}$.
- H is a $d \times d$ normalized Walsh-Hadamard matrix.
- D is a $d \times d$ diagonal matrix where D_{ij} takes the values -1 and 1 with probability $1/2$.

We used Gabriel Krummenacher's code (https://github.com/gabobert/fast-jlt), with some debugging (https://github.com/vkli/fast-jlt).

Further dimension reduction: FJLT

Pros:

- Assumes no prior info (vs. PCA, compressed sensing)
- Preserves distances with low distortion (vs. locality sensitive hashing)
- Fast (vs. JL), worst-case $O(d \log d + qd\varepsilon^{-2} \log n)$ per vector

Cons:

- Tradeoff between distortion ϵ and dimension reduction $k = O(\frac{\ln n}{\epsilon^2})$

$\epsilon = 0.1, n = 729 :\ k \geq \frac{8 \ln(20n)}{\varepsilon^2} = 7670$ (for .995 accuracy, JL)

We computed results for
$k = 7670, 8570, 9470, 10370, 11270, 12170, 13070, 13970, 14870, 15770$

Distance Metrics

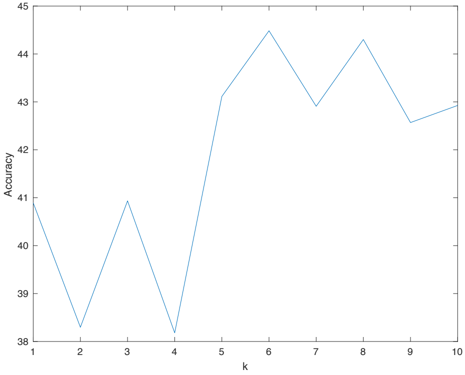
- Frame-based clustering: Mixture of 30 Gaussian models (GMM) & Single Gaussian Model
  - Each song fitted with either a GMM or G1 for each of the 20 MFCCs.
  - Distance between each song measured by an exact (G1) or an approximation (G30)[2] of the Kullback-Leibler divergence for use in K-nearest neighbor classifier.
  - Mean vectors for each song used as feacture vectors for Naive-Bayes classifier.

- Euclidean Norm
  - Used in KNN with the reduced features after performing the FJLT.

[2] Hershey, John R., and Peder A. Olsen. "Approximating the Kullback Leibler divergence between Gaussian mixture models." 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07. Vol. 4. IEEE, 2007.

Statistical Learning: K-nearest neighbors (KNN)

- A non-parametric algorithm, thus makes no assumption about the underlying data distribution,
- Stores all available information and classifies new data based on some similarity measure,
- Optimal *k* chosen based on exploration.



Statistical Learning: Naive Bayes

- A classification algorithm based on Bayes Theorem $\left[P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}\right]$ with a 'naive' assumption of independence among features.
- Essentially, this translates to posterior $=$ prior $\times$ likelihood.
- Uses the notion of 'probability' rather than a notion of some distance metric to classify nodes.
- Can do well with a small number of training data to estimate parameters needed for classification.
- Gaussian Naive Bayes posterior:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{1}$$

Statistical Learning: Naive Bayes

- Outline of the algorithm:
  - Use training data to compute the posterior probability $P(x_i|y)$ for each class or genre.
  - Find the vector of parameters $\hat{\theta}$ that maximizes the posterior probability.
  - For a new testing data, the output label $\hat{y}$ of the classifier is given by,

$$\hat{y} = \text{arg max}_{y \in 1,..,k} P(x_i|y, \hat{\theta})$$

- Implemented through Matlab function: *fitcnb*

Results of KNN: G1

	Classical	Electronic	Jazz/Blue	Metal/Punk	Rock/Pop	World
Classical	**0.9844**	0.0094	0.0000	0.0000	0.0031	0.0031
Electronic	0.0881	**0.5700**	0.0000	0.0000	0.3328	0.0091
Jazz/Blue	0.5067	0.1133	**0.1933**	0.0000	0.1867	0.0000
Metal/Punk	0.0000	0.1111	0.0000	**0.4444**	0.4444	0.0000
Rock/Pop	0.0791	0.0600	0.0000	0.0500	**0.8018**	0.0091
World	0.5237	0.1135	0.0000	0.0000	0.2391	**0.1237**

Table : Confusion matrix averaged over a 5-fold 10 times cross-validation.

Actual Accuracy: 51.96%

Results of KNN: GMM

	Classical	Electronic	Jazz/Blue	Metal/Punk	Rock/Pop	World
Classical	**0.9688**	0.0031	0.0000	0.0000	0.0000	0.0281
Electronic	0.0443	**0.6850**	0.0000	0.0000	0.2360	0.0348
Jazz/Blue	0.3333	0.2000	**0.2667**	0.0000	0.1600	0.0400
Metal/Punk	0.0000	0.1778	0.0000	**0.2444**	0.5778	0.0000
Rock/Pop	0.0200	0.1545	0.0100	0.0500	**0.7555**	0.0100
World	0.3763	0.1968	0.0000	0.0000	0.2308	**0.1962**

Table : Confusion matrix averaged over a 5-fold 10 times cross-validation.

Actual Accuracy: 51.94%

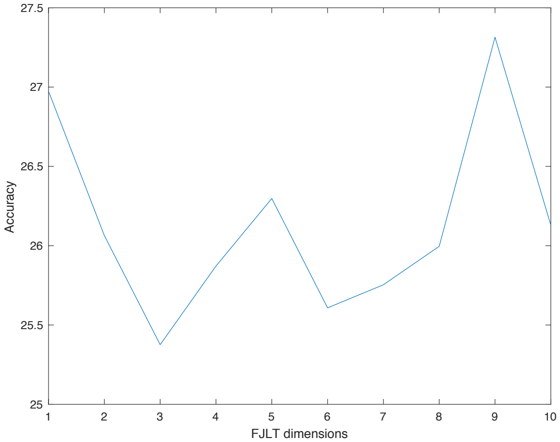


Figure : Comparison of dimensions of FJLT versus overall accuracy over all songs

Results of KNN: FJLT with Euclidean

	Classical	Electronic	Jazz/Blue	Metal/Punk	Rock/Pop	World
Classical	**0.8122**	0.0550	0.0075	0.0000	0.0231	0.1022
Electronic	0.3869	**0.1885**	0.0026	0.0939	0.1418	0.1862
Jazz/Blue	0.4933	0.1493	**0.0760**	0.0000	0.1440	0.1373
Metal/Punk	0.1200	0.1622	0.0000	**0.1422**	0.5089	0.0667
Rock/Pop	0.2325	0.1460	0.0200	0.1311	**0.3180**	0.1524
World	0.4778	0.1664	0.0269	0.0250	0.1178	**0.1861**

Table : Confusion matrix averaged over a 5-fold 10 times cross-validation.

Actual Accuracy: 28.72%

Results of Naive Bayes: G1

	Classical	Electronic	Jazz/Blue	Metal/Punk	Rock/Pop	World
Classical	**0.8031**	0.0344	0.0063	0.0000	0.0563	0.1000
Electronic	0.0439	**0.4984**	0.0087	0.0530	0.2826	0.1134
Jazz/Blue	0.1533	0.2600	**0.3533**	0.0400	0.1133	0.0800
Metal/Punk	0.0000	0.0222	0.0000	**0.7556**	0.2222	0.0000
Rock/Pop	0.0382	0.1573	0.0100	0.3118	**0.4627**	0.0200
World	0.3929	0.1808	0.0237	0.0000	0.1481	**0.2545**

Table : Confusion matrix averaged over a 5-fold 10 times cross-validation.

Actual Accuracy: 52.13%

Naive Bayes GMM Actual Accuracy : 52.05%

Limitations and Computation Time

- KNN performance was highly dependent on the number of samples for each genre.
- Distance metric used influences the accuracy a lot.
- The independence assumption of Naive Bayes does not reflect real life and affects performance.

- KNN has a larger runtime than Naive Bayes.
- GMM with KL divergence had the longest runtime ($\sim$ 30min)

Statistical Learning: Graph Laplacian

Partitioning the nodes (songs) into k nearly disconnected components (genres) is equivalent to partitioning the indices (by their corresponding entry values) of the first k eigenvectors (corresponding to the k smallest eigenvalues) of $L_{rw} = D^{-1}L$

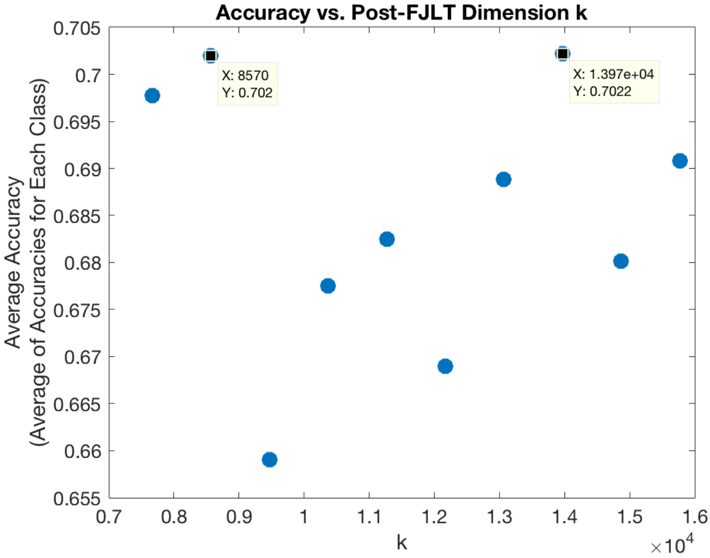
$L = D - W$

$D_{ii} = \sum_{j=1}^{n} W_{ij}$

$$W_{i,j} = \begin{cases} \frac{1}{1+D_{ij}} & i \neq j \\ 0 & i = j \end{cases}$$

1. Compute the first k eigenvectors $v_1, ..., v_k$ corresponding to the k smallest eigenvalues of L.
2. Form the matrix V with v_i as columns.
3. Let y_i be the vector corresponding to the i^{th} row of V.
4. Cluster $y_1, ..., y_n$ into clusters $C_1, ..., C_k$ using the k-means algorithm.
5. Return clusters $A_i = \{j | y_j \epsilon C_k\}$.

Statistical Learning: Graph Laplacian



Statistical Learning: Graph Laplacian

Best Confusion Matrix (k = 13970):

	Classical	Non-classical
Classical	**.5444**	.4556
Non-classical	.1400	**.8600**

Actual Accuracy: 70.22%

Conclusion/Future work

60-70% accuracy

Compare computation times:
FJLT: 100 minutes per song = 10 minutes per fjlt per song
Graph Laplacian: 5783 s per 729 songs

	G30	G30S	G1
FC	25000	700	30.0
CMS	400	2	0.1

CPU times per song for Flame Clustering, Cluster Model Similarity[3]

[3]Pampalk, E. (2006) Computational models of music similarity and their application in music information retrieval (Doctoral dissertation). Technischen Universität Wien, Vienna, Austria.