Cover Song Classification

Ken Kao, Ian Tenney

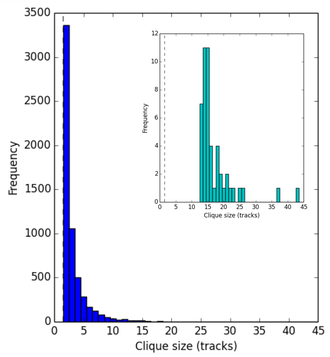
Overview

The overall task of our project is to develop a method to quantify song similarity. Possible applications to this system include:

1. Identify cover songs or concert/live versions of a track
2. Identify songs in the presence of noisy backgrounds and distortion
3. Form the basis for a song-recommendation engine.

Data

We used data from the million Song Dataset (MSD), which consists of various low-level features for a large number of songs. We label this data using the Second Hand Songs Database (SHS), which groups a subset of 18,000 MSD tracks into "cliques". The plot on the right shows us the histogram of the size of each clique. The plot on the top-right is the distribution for the 50 largest cliques.

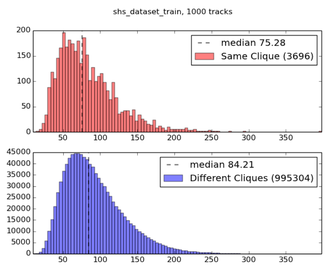


Features

- General song properties: loudness, duration, tempo, key…etc. (combo, combo-plus)
- Pitch and timbre (MFCC-like) features. (combo, combo-plus)
- Loudness features: max, segment, attack at sample points. (combo-plus)
- Segment, beat, and tatum duration statistics. (combo-plus)

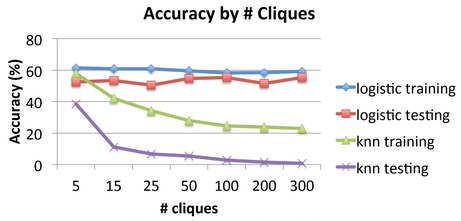
Baseline System

As the baseline, we simply extracted the time-average timbre vector from 309 cliques, which has 1000 songs total. We then find the Euclidean distance between two tracks of the same and different cliques.



We observe a slight difference in the two distributions, it is dwarfed by the variance. The average timbre feature is not sufficient to distinguish cover songs from each other.

Results



As we add more cliques, the data becomes more variable, making it harder to generalize.

Figure 1. Effect of varying the number of cliques. All tests run with k=5 for knn.

For a smaller number of cliques, "combo-plus" seems to generalize better than "combo" features.

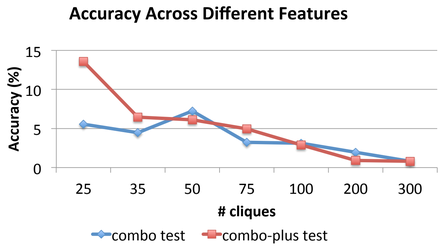


Figure 2. Effect of varying the number of cliques and feature sets. All tests run with k=7 for knn.

k	Training accuracy	Test accuracy
5	27.88	5.48
6	26.12	5.73
7	**25.16**	**7.25**
8	23.72	6.49
9	23.08	6.87
12	22.12	5.34
15	21.31	5.34

Table 1. Effect of changing k. All tests run with "combo" features, logistic regression, and 50 cliques

Increase in k generally increases the test accuracy up to k=7, which seems to be the best balance between precision and noise tolerance.

Increased regularization strength reduces over-fitting, thus generalizes better.
Pro-process whitening also improves accuracy by reducing cross correlations.

	Scale	Whitening
R=1	3.44	5.73
R=100	4.96	6.11

Table 2. Effect of changing r (L1 regularization) and the pre-processing of features. All test runs used combo-plus features.

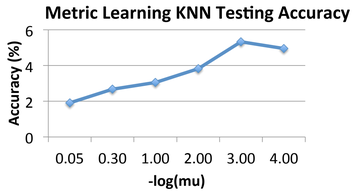


Figure 3. Effect of changing mu in Metric learning

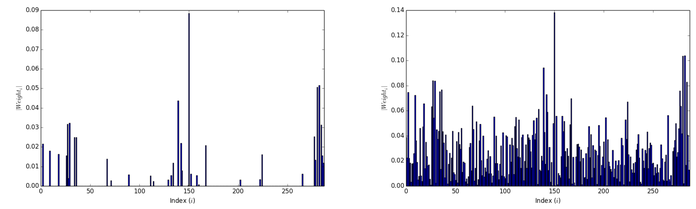
Decreasing mu increases regularization, which in turn improves generalization.

Sources

[1] Blitzer, J., Weinberger, K. Q., and Saul, L. K. Distance metric learning for large margin nearest neighbor classification. In Advances in neural information processing systems (2005), pp. 1473–1480.
[2] Slaney, M., Weinberger, K. Q., and White, W. Learning a metric for music similarity. In ISMIR (2008), pp. 313–318.
[3] Weinberger, K., and Saul, L. Distance metric learning for large margin nearest neighbor classification. The Journal of Machine Learning Research 10 (2009), 207–244.

Supervised Learning

We applied logistic regression to train the weights for our tracks. Below is the weight vector from L1 regularization (on the left) and L2 regularization (on the right). We use this method to automatically learn the importance of each feature within our feature vector. This is equivalent to scaling the feature by a diagonal matrix.



Metric Learning

Objective:

- Minimize distance between a point and its "target" neighbors (same clique)
- Reduce number of nearby "non-target" neighbors (different clique)

Method:

- Learns a transformation that embeds data into Euclidean space.
- Equivalent to learning an nxn matrix such that:

$$d(x_1, x_2) = ||L(x_1 - x_2)|| = \sqrt{||(x_1 - x_2)^T M(x_1 - x_2)||}$$

Advantages:

- Capture cross-relations between different features
- Directly targeting a KNN-like objective function

K Nearest Neighbors

The K Nearest Neighbor plots all the training sample into an n-dimensional graph where n is the size of the feature vector. In our specific case, we multiply the feature vector by the corresponding weight vector (in the logistic regression's case) or weight matrix (for metric learning) and then plot it.

Conclusion/Future Work

Using low-level features on a large dataset with supervised and metric learning seems to be limited in their ability to generalize.

Our best results are from a high-dimensional feature set in a strongly-regularized supervised learning approach, but we predict that the more sophisticated metric learning techniques may be able to perform better with further tuning and refinement.

We suspect that our global parameter approach may not be well-suited for this problem, and further inquiry may extend our framework to a hierarchical model that learns several metrics, each for a restricted subset of the data (e.g. genre).