DANCE TYPE CLASSIFICATION IN IRISH AND SCANDINAVIAN FOLK MUSIC

Elliot Kermit-Canfield and Iran Roman

Center for Computer Research in Music and Acoustics,
Stanford University, Stanford, CA 94305 USA
[kermit|iran]@ccrma.stanford.edu

ABSTRACT

For centuries, western cultures have written folk songs down. In the 21st century, this has resulted in large databases of music from all around the world. We have built, trained, and tested classifiers on Irish and Scandinavian dance music using songs encoded in symbolic representation (ABC format), downloaded from John Chamber's online folk-song database. These tunes were sorted by dance type: Reel, Jig, Hambo, Pols, and Hornpipe. Raw data was preprocessed to be in the same key, have no ornaments, and no abbreviations. Features extracted were standardized to have zero mean and unit variance. Extra trees classification and variance thresholding allowed us to reduce feature dimensionality from 65 to 23. We classified our data using Support Vector Machines (SVM), Logistic Regression, Naïve Bayes, Gradient Descent, and K-means Clustering. We evaluated these classification algorithms using leave-one-out cross-validation, trained on 80 percent of the data, and tested on 20 percent. Our SVM classification methods turned out to be the most accurate with less than 0.10 training and testing error.

1. INTRODUCTION

As early as 1956, Alan Lomax envisioned using algorithmic methods for the classification of folk music [1]. This automatic method would allow musicologists to associate a newly discovered musical score with other folk songs similar in style and genre. Since then, teams of musicologists have employed a variety of machine learning techniques to classify musical genre and style [2, 3, 4]. Within the past decade, McKay and Fujinaga have worked toward developing feature extraction algorithms that classify song genres using high-level musical features [5]. In this paper, we were interested in classifying folk music of Irish and Scandinavian origin. Using a subset of John Chamber's collection of more than 20,000 codified and organized tunes from Europe, we have built, trained, and tested a classifier on Irish and Scandinavian dance music. In order to extract our features, we used a subset of McKay's and Fujinaga's feature extraction algorithms [6]. More specifically, we wanted to see how several standard machine learning algorithms—including Support Vector Machines using Linear, RBF, and Polynomial Kernels, K-Means Clustering, Logistic Regression, Naïve Bayes, and Gradient Descent—perform at classifying dance type in a one-vs-one classification task for each pair of dance types in our corpus.

2. METHODS

2.1. Song Corpus

We used 1167 folk songs stored in ABC[1] format primarily downloaded from John Chambers's collection [7]. These tunes were divided into several categories, sorted by dance type—Reel [598 songs], Jig [352 songs], Hambo [51 songs], Pols [76 songs] and Hornpipe [91 songs]. After data standardization, for each dance type we randomized the song order and truncated our data sets. This was done so we had the same number of training and testing examples across dance types and thus, decreased the possibility of biasing larger data sets.

2.2. Data Standardization and Normalization

We found obvious notational variability, as the original corpus contains songs that were input by different authors. In order to compare songs both within and across dance type, we standardized our data by removing duplicate entries, transposing all songs into the same key, removing grace notes, removing polyphony, expanding repeated sections that were notated in an abbreviated manner, and normalizing rhythmic subdivisions. Figure 1 outlines the steps we followed to clean the data.

After feature extraction, we performed additional processing to normalize our data so that the various machine learning algorithms would not be biased towards extreme data values. We encoded categorical data as individual features. Then, we tested normalizing the data to be between zero and one, as well as having zero mean and unit variance. The second normalization proved to be more successful across algorithms.

2.3. Feature Selection

While metric features might play a larger role classifying dance types correctly, melodic features can also help identify a song's genre. Melodic feature extraction consists of building histograms of melodic interval and pitch-classes throughout a song. Additionally, we calculated metrics that summarize melodic direction/arc, cadences, and repeated notes. Rhythmic features included time signature, features describing note duration, time between consecutive notes, and variability of time between consecutive notes. We identified 65 total features (55 melodic and 10 rhythmic) that could potentially be useful for classifying song type. We used tools from Music21 to facilitate feature extraction [8].

After extracting features, we employed variance thresholding and extra trees classification to evaluate the usefulness of our fea-

[1] ABC notation is a plain text format designed to be both human and machine readable. The ABC format includes all melodic and rhythmic features of these folk songs. It is primarily used for notating folk songs and uses a standard character set to represent music in symbolic notation.

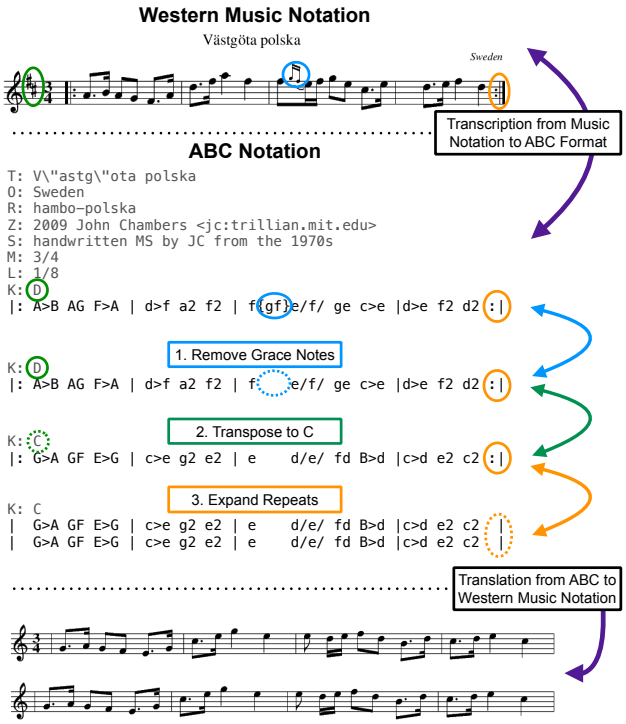


Figure 1: Steps followed to standardize each song in the corpus.

tures. Given the authors' music theory backgrounds, the final feature selection was assessed through a combination of musical intuition, and the results from these feature selection algorithms. In the end, we reduced the number of features from 65 to 23—8 describing rhythmic landmarks in our data, and 15 describing melodic qualities.

2.4. Classification Algorithms

We used a one-vs-one classification approach where we compared every dance type to each of the other song types. We randomly selected 40 songs of each dance type to serve as the training set and 11 to be the final evaluation set. For each classification task, we randomized the order of songs in the training sets and evaluation sets, although the distinction between training and testing was never changed.

For our classification, we used five supervised learning algorithms and one unsupervised learning algorithm. Our supervised learning algorithms included a Support Vector Machine (SVM) with a linear kernel. We decided to use a linear kernel, as the total number of features (23) is small in comparison to the amount of training data (40 songs in each case). We also implemented an SVM with a radial basis function (RBF) kernel (gaussian Kernel), which is shown in equation 1.

$$K(x, z) = \exp(-\frac{||x - z||^2}{2\sigma^2}) \tag{1}$$

The RBF kernel maps to an infinite feature space where x and z are two vectors containing the input feature space for two different classes and σ is a free parameter of the gaussian distribution.

Another supervised learning algorithm that we tried out was Logistic Regression, which makes a binary decision about our data based on the sigmoid function seen in equation 2.

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

where:

$$z = \theta^T X \tag{3}$$

Logistic Regression basis its output on a finite features space describing the data. Given our data set and our one-vs-one classification methodology, this algorithm is a simpler model compared to the SVM algorithm.

The next supervised learning algorithms we implemented was Gradient Descent, for which the formula can be found in equation 4, where α is called the learning rate and $J(\theta)$ is the cost function as seen in equation 5.

$$\theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta) \tag{4}$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2 \tag{5}$$

This algorithm repeatedly takes steps toward the direction that lowers $J(\theta)$ with the steepest slope.

The final supervised learning algorithm was Naïve Bayes. We expect the distribution in several of the melodic and rhythmic features that we extracted to be gaussian. With our data clean and normalized across features, we implemented a gaussian Naïve Bayes algorithm for classification, as shown in equation 6, where the likelihood of the features describing x is assumed to be gaussian.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{6}$$

K-means Clustering was the unsupervised-learning classification method that we also used to classify the data, as seen in equation 7.

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 \tag{7}$$

In our specific case, the number of clusters, S, only had two dimensions, thus $k = 2$, and our task is to find the mean, μ_i, of each S.

We evaluated the performance of the supervised learning algorithms using leave-one-out cross-validation on the training data. In this process, the model is training on all the data except a single training set, which is left aside for validation. This procedure is repeated exhaustively until all data points have been set aside. The results of this algorithm evaluation method are summarized in Table 1.

The metric to test performance of K-means on the training data was the f1 score, which relates the number of correct positive results to the actual number of examples in each of the classes on which the algorithm was trained. These results are also summarized in Table 1.

2.5. Visual Representation of Data

In order to plot the output of our classification algorithms using a 2D scatter plot, we carried out principal component analysis (PCA) to find the first two principal components of the feature space. The equation for PCA that finds the first principal component is shown to maximize

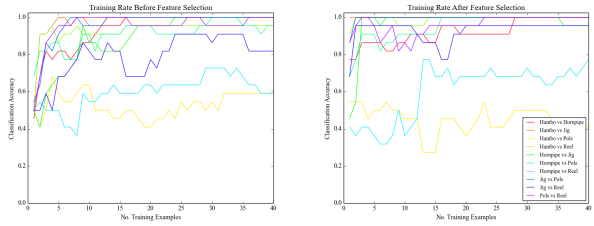
$$\frac{1}{m}\sum_{i=1}^{m}(x^{(i)^T}u)^2 = u^T\left(\frac{1}{m}\sum_{i=1}^{m}x^{(i)}x^{(i)^T}\right)u\,, \qquad (8)$$

where u is the unit vector representing the direction on which the data points $x^{(i)}$ can be projected. The first principal component found by PCA depicts the axis on which the variance of the data is retained and maximized. We can find k principal components by maximizing the first k u's to project the data into k-dimensional subspace.

3. RESULTS AND DISCUSSION

3.1. Algorithm Accuracy After Feature Selection

The training rate of our linear SVM algorithm before feature selection is shown in Figure 2a. Each line represents classification between a pair of song-types. Before reducing our feature dimensionality, the number of training examples needed for classification accuracy of most song-type pairs to converge oscillated around 15. We used variance thresholding and extra trees classification to reduce the number of features to 23. The training rate after feature selection for the same classification algorithm is shown in Figure 2b. Reducing feature dimensionality lowered the number of training examples needed for classification accuracy to converge across song-type pairs.



(a) Before Feature Selection (b) After Feature Selection

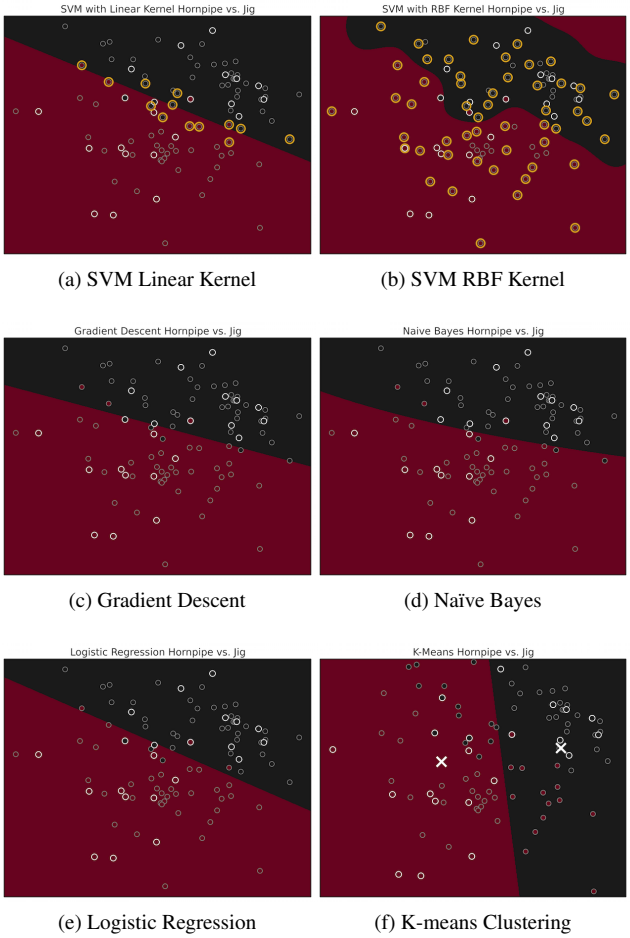
Figure 2: Learning rate for SVM with linear kernel before and after feature selection. Each line represents classification between a pair of song-types.

3.2. Evaluation of Machine Learning Algorithms

The training accuracies of our classification algorithms, as quantified by leave-one-out cross-validation (f1 score was the validation metric of K-means clustering), is summarized in Table 1. Classification algorithms were able to separate most song-type pairs, except Hambos vs Pols and Reels vs Hornpipes. The algorithm with the lowest average training accuracy was K-means clustering.

With the test data, our classification algorithms separated most song-type pairs with test accuracies around 90 percent. However, classification of Hambos vs Pols and Reels vs Hornpipes remained with close-to-chance performance across algorithms. The linear and RBF SVM algorithms showed the highest average test accuracy, while K-means clustering presented the lowest one.

Figure 3 shows the output of various classification algorithms of the first two principal components of a PCA reduced version of the training data for the Hornpipe vs Jig classification task. It is interesting to compare the performances of the algorithms back to back. Visually, it is clear that the data separates fairly well. The SVM with Linear Kernel, Gradient Descent, Naïve Bayes, and Logistic Regression all draw a similar decision boundary. While the SVM with RBF Kernel has a similar error to the other algorithms, its large number of support vectors is evidence to extreme overfitting. K-Means Clustering does not know the labels of dance types, and its decision boundary is not as accurate as the other algorithms. It finds two gaussian distributions among the data that do not reflect the true categorical labels.



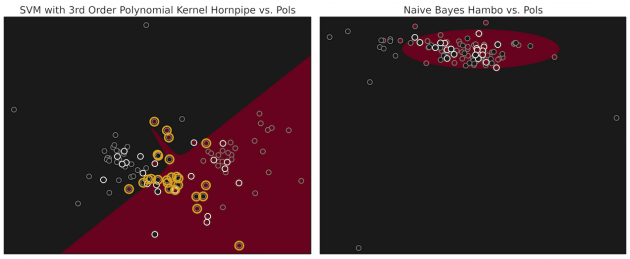
(a) SVM Linear Kernel (b) SVM RBF Kernel

(c) Gradient Descent (d) Naïve Bayes

(e) Logistic Regression (f) K-means Clustering

Figure 3: Comparison of classification algorithms for the Hornpipe vs Jig dance pair. The graphs plot the first two principle components of a PCA reduced version of the data. Training data is displayed with gray circles and test data is printed with white circles. Gold circles ring the support vectors and X's show the centroids of clusters (when applicable). Data points that match the color of the background show correctly classified songs while ones that are the opposite color are misclassified.

Figure 4 shows cases where the classification goes horribly wrong. In one plot, we see the result of an SVM with a 3rd order Polynomial Kernel. There are several mislabeled points in the training set that the algorithm tries to overfit (unsuccessfully). In the second plot, the dance types are completely inseparable. Clearly the algorithm stands no chance to draw a reasonable decision boundary for this pair of dance types.



(a) SVM 3rd Order Polynomial Kernel

(b) Naïve Bayes

Figure 4: Two cases where the algorithms failed to classify the data satisfactorily. In (a) we see a Polynomial Kernel with severe overfitting. In (b), the data is inseparable, and thus the classification is no better than chance.

4. CONCLUSIONS

Clearly, our use of machine learning has a long way to come before it will rival a human ear. Dance types that are more dissimilar, such as Hornpipe vs Jig, are easy to classify. However, classification of dances that are similar in meter and style, such as Reel vs Hornpipe, turned out to be no better than chance. Our results suggest that we can write new feature extraction methods that take advantage of mesoscale qualities in our data. These qualities could include phrase and sub-phrase quantifiers, form decoders, and N-gram identification of rhythmic and melodic motives. In all cases, we suspect a more sophisticated treatment of events at multiple temporal scales would greatly increase the accuracy of our classification algorithms.

Finally, we can expand our classification methods to include unsupervised learning algorithms and neural networks. Most of the algorithms we explored are supervised algorithms that preassume we know the dance type of the training data. We think it would be interesting to see if we could trace musical similarity through dance forms in an unsupervised manner, in order to gain cultural and historical insights about how folk music spread throughout Northern Europe.

5. REFERENCES

[1] Alan Lomax, "Folk song style: Notes on a systematic approach to the study of folk song," *International Folk Music Journal*, 1956.

[2] Wei Chai and Barry Vercoe, "Folk music classification using hidden markov models," *International Conference on Artificial Intelligence*, 2001.

[3] Michael Cuthbert, Christopher Ariza, and Lisa Friedland, "Feature extraction and machine learning on symbolic music using the music21 toolkit," *ISMIR*, 2011.

[4] Shyamala Doraisamy, Shahram Golzari, Noris Mohd. Norowi, Md. Sulaiman, and Nur Izura Udzir, "A study on feature selection and classification techniques for automatic genre classification of traditional malay music," *ISMIR*, 2008.

[5] Cory McKay and Ichiro Fujinaga, "Automatic genre classification using large high-level musical feature sets," *ISMIR*, 2004.

[6] Cory McKay and Ichiro Fujinaga, "jsymbolic: A feature extractor for midi files," *ICMC*, 2006.

[7] "Jc's abc music collection," `http://trillian.mit.edu/~jc/music/`.

[8] "Music21: a toolkit for computer-aided musicology," `http://web.mit.edu/music21/`.

Table 1: Training accuracy for algorithms on a normalized 40-sample training data set. Data was normalized to have zero mean and standard deviation of one. Cross-validation is the validation method for all algorithms except K-means clustering, for which f1 score is the validation metric.

	SVM (linear)	SVM (RBF)	K-means	Logistic Regression	Naïve Bayes	Gradient Descent
Hambo vs Hornpipe	1.00 (+/- 0.00)	0.99 (+/- 0.16)	0.92 (+/- 0.03)	1.00 (+/- 0.00)	1.00 (+/- 0.00)	0.97 (+/- 0.22)
Hambo vs Jig	1.00 (+/- 0.00)	1.00 (+/- 0.00)	1.00 (+/- 0.00)	1.00 (+/- 0.00)	0.99 (+/- 0.16)	1.00 (+/- 0.00)
Hambo vs Pols	0.72 (+/- 0.55)	0.81 (+/- 0.53)	0.61 (+/- 0.15)	0.69 (+/- 0.62)	0.55 (+/- 0.30)	0.72 (+/- 0.59)
Hambo vs Reel	0.95 (+/- 0.30)	0.97 (+/- 0.22)	1.00 (+/- 0.05)	0.97 (+/- 0.22)	0.97 (+/- 0.22)	0.96 (+/- 0.26)
Hornpipe vs Jig	0.99 (+/- 0.16)	1.00 (+/- 0.00)	0.81 (+/- 0.04)	0.99 (+/- 0.16)	0.99 (+/- 0.16)	0.97 (+/- 0.22)
Hornpipe vs Pols	0.99 (+/- 0.16)	0.97 (+/- 0.22)	0.92 (+/- 0.06)	0.99 (+/- 0.16)	0.99 (+/- 0.16)	0.97 (+/- 0.22)
Hornpipe vs Reel	0.61 (+/- 0.72)	0.64 (+/- 0.77)	0.58 (+/- 0.09)	0.61 (+/- 0.72)	0.57 (+/- 0.36)	0.61 (+/- 0.65)
Jig vs Pols	0.97 (+/- 0.22)	0.99 (+/- 0.16)	1.00 (+/- 0.05)	0.97 (+/- 0.22)	0.96 (+/- 0.26)	0.96 (+/- 0.26)
Jig vs Reel	0.95 (+/- 0.30)	0.95 (+/- 0.30)	0.63 (+/- 0.06)	0.96 (+/- 0.26)	0.95 (+/- 0.30)	0.95 (+/- 0.30)
Pols vs Reel	0.94 (+/- 0.33)	0.93 (+/- 0.36)	1.00 (+/- 0.07)	0.93 (+/- 0.36)	0.95 (+/- 0.30)	0.85 (+/- 0.51)
Average Accuracy	0.91	0.93	0.85	0.91	0.90	0.90

Table 2: Test error of algorithms on a normalized 22-sample test set. Data was normalized to have zero mean and standard deviation of one.

	SVM (linear)	SVM (RBF)	K-means	Logistic Regression	Naïve Bayes	Gradient Descent
Hambo vs Hornpipe	1.00	0.95	0.92	1.00	0.96	1.00
Hambo vs Jig	1.00	1.00	1.00	1.00	1.00	0.96
Hambo vs Pols	0.41	0.68	0.64	0.46	0.50	0.59
Hambo vs Reel	1.00	1.00	1.00	1.00	1.00	1.00
Hornpipe vs Jig	0.96	0.96	0.81	0.96	0.96	0.96
Hornpipe vs Pols	1.00	1.00	0.92	1.00	0.96	1.00
Hornpipe vs Reel	0.77	0.50	0.60	0.64	0.55	0.41
Jig vs Pols	1.00	1.00	1.00	1.00	1.00	0.96
Jig vs Reel	0.96	0.96	0.64	0.96	0.96	0.91
Pols vs Reel	1.00	1.00	1.00	1.00	1.00	1.00
Average Accuracy	0.91	0.91	0.85	0.90	0.90	0.88