Attribution of Musical Works to Josquin des Prez

Philip Lee
Department of Electrical Engineering
Stanford University
Stanford, CA 94305

Kate Stuckman
Department of Electrical Engineering
Stanford University
Stanford, CA 94305

Zachary Sunberg
Department of Aeronautics and Astronautics
Stanford University
Stanford, CA 94305

***Abstract*—Josquin des Prez was one of the most important composers of the middle renaissance era. However, there is disagreement in the academic community about what musical pieces should be attributed to him. The project described here uses a machine learning approach to classify dubiously attributed works to Josquin or his near contemporaries. Using features known as counterpoint modules, we were able to build supervised learning models from a training set of known music. We attempted both multi-class (classifying all composers) and binary (Josquin or not Josquin) classification. The best multi-class model had a cross-validation accuracy of 81%. The best binary classifier had an Area Under the ROC Curve of .8707 and an accuracy of 84%. Using this model, we predict that between 90 and 130 of our test set of 285 questionable works were actually written by Josquin.**

I. Introduction

Josquin des Prez was a renaissance composer of the 15th and early 16th centuries. He is considered one of the greatest composers of the era. However, his fame caused his name to become attached to many musical pieces from that period, and there is disagreement in the music community regarding which historical pieces should actually be attributed to him. Working in conjunction with Professors Jesse Rodin and Craig Sapp of the Josquin Research Project, we analyzed songs from the renaissance for which there is scholarly consensus regarding whether or not they were actually composed by Josquin [1]. This comprised our training set. Using this data, we built a machine learning model to classify borderline cases.

II. Data

We used data collected by the Josquin Research Project at the Stanford Center for Computer Assisted Research in the Humanities (CCARH). The research group has been collecting and manually digitizing music from Josquin and several of his contemporaries from the renaissance. Typically, these songs are choral music with four voices in a song. The digital files include information from historical scores from these composers (including each voice, notes, rhythms, and pitch) and are available in many formats, including a Matlab-friendly matrix. The research team has also developed tools to easily query for features in the music that may be useful in attributing the works, such as note patterns, specific rhythms, and chord progressions.

Of specific interest is a group of music traditionally attributed to Josquin, but determined by field experts to be of questionable origin (due to conflicting sources, a dearth of sources, and/or differing musical styles). We built a machine learning model that performs well on the music with confident labels (the training set) to make predictions on this questionable group of music (the test set).

III. Method

We first began this supervised learning problem as a multi-class classification problem. We extracted several features deemed interesting by researchers Rodin and Sapp (described below) and applied the features to several machine learning models (such as Naive Bayes, Support Vector Machines (SVM), and classification trees) separately to try to classify which renaissance composer was responsible for each piece of music. We used the results from these multi-class classifiers to select relevant features. We then generalized these models to our primary problem: binary classification (Josquin vs. not Josquin). Finally, we combined the binary classification models using an ensemble method. Our feature extraction, feature selection, and models were implemented in Matlab.

IV. Features

The data available for our use in classification were digital representations of scores, including information such as note pitch and timing. We tried a variety of features with varying levels of complexity to differentiate between authors.

A. Simple Pitch Intervals

The first pass of analysis relied simply on creating histograms of the pitch intervals between adjacent notes in the same part. Each feature is the number of times that a given pitch interval occurs in the song.

B. Counterpoint Modules

The concept of counterpoint is an important tool for the analysis of music from Josquin's period [3], [4]. Counterpoint modules are features that compactly capture information about how a composer uses pitch transitions, and were recommended by our music faculty advisers. A counterpoint module can be described by three pitch intervals, and is made up of data from two of the musical parts. For example, consider part A with temporally adjacent notes $(A, 1)$ and $(A, 2)$ and part B with notes $(B, 1)$ and $(B, 2)$ that begin at the same time as $(A, 1)$ and $(A, 2)$ respectively. The counterpoint module for these notes consists of the intervals between notes $(A, 1)$ and $(B, 1)$, $(B, 1)$ and $(B, 2)$, and $(A, 1)$. Figure 1 shows a single counterpoint module. Each feature is the number of times a counterpoint triplet appears in a musical work.

C. Dissonance

The use of dissonance was emerging in Josquin's time, so the use of dissonant chords was changing in this time period. Dissonance in the renaissance era is defined as 2nds,

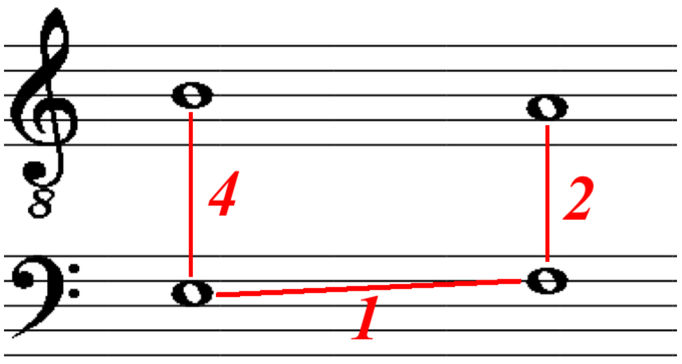

Fig. 1. A counterpoint module

6th, and in rare cases 4ths from the lowest note. Analysis here involved creating a histogram of these chords specifically across different voice parts.

D. *Rhythms*

Rhythmic patterns in a single measure can be analysed for each voice part. This can be quantified as a binary value, with each bit representing whether this beat contained a note attack. A histogram of the distribution of the combinations of rhythms can be extracted. This method required time signature information in the data files, so the data format for each score provided by the Josquin Research Project was altered before analysis of this feature was completed.

V. Machine Learning Approach

We applied several machine learning models to fit our problem. We began with a Naive Bayes approach (with Laplace Smoothing). We then used the libsvm package to utilize Support Vector Machines [2] and built in Matlab functions to train models using classification trees. We then took the probability outputs of these models and used a RobustBoost ensemble learner to combine the models appropriately. Additionally, we performed feature selection and Principal Component Analysis (PCA) to reduce the dimensionality of our features.

A. *Model Choices*

We chose these models based on how well we thought they would apply to the problem, resources available to us, ease of implementation, and how well they would fit into our ensemble framework. Naive Bayes was a model that seemed suited for our primary features (counting occurrences of counterpoints). The classification trees were utilized mostly due to ease of implementation. The RobustBoost algorithm was chosen for our ensemble learner because our data was fairly skewed. Of our training samples, less than 30% were written by Josquin.

B. *Feature Reduction*

The cardinality of the set of all features present in the training set is much larger than the number of training examples. Table I below illustrates the dimensions of each of our features. We believe that this caused nonlinear classifiers such as Gaussian kernel SVMs to over-fit. In order to mitigate this problem, we attempted to reduce the dimensionality of the feature space by using greedy forward search feature selection and PCA. The forward search feature selection proved to be very inconvenient in terms of computational time, and did not lead to improvements in initial tests, so it was abandoned. After experimenting with different numbers of principal components used, we used the components that accounted for 99% of the variance in the original data. This resulted in a feature space with a dimension that was slightly lower than the number of training examples. Use of PCA significantly improved our computational time, however it did not significantly reduce the generalization error estimated using cross-validation, and we remain suspicious of over-fitting.

Dimensionality of Features	
Pitch Intervals	80
Counterpoints	$O(15^3)$
Rhythms	2^12
Dissonance	3

TABLE I. Dimensions of each feature

VI. Evaluation Metrics

Because we have a limited data size (443 total songs), we used cross- validation to measure the accuracy of our models. We relied on mostly 10-fold cross validation to judge our progress. For multi-class classification, we used cross-validation accuracy (sum of the diagonals in the confusion matrix divided by the total number of songs) as an estimate of one minus the generalization error. For binary classification models, we looked to maximize cross-validated Receiver Operating Characteristic (ROC) and the corresponding Area Under Curve (AUC) before attempting to classify the disputed cases.

VII. Results

A. *Multi-Class Classification*

The results of various tests are shown below to illustrate the performance of our multi-class classification models. Figures 2-4 show the results of ten-fold cross-validation testing with various sets of features and machine learning approaches. Each graph shows all of the information from the confusion matrix. The labels below the bars show the actual known composer of the pieces in the bars above, and the size of the bars represent the number of pieces by that composer that were classified to belong to each composer. Perfect classification would leave only the single bar corresponding to that composer above each composer's name.

Through experimentation, we found that the counterpoint features were the most predictive of composers. Adding other features did not significantly improve our results. We give a sample of the results from different sets of features below.

Figure 2 shows the poor results produced by using only pitch intervals with a multinomial naive Bayes classifier; figure 3 shows the greatly improved results using counterpoint features; and figure 4 shows the results using a Gaussian kernel SVM. Even after reducing the dimensionality of the feature space using PCA, naive Bayes produces a slightly lower estimated generalization error than the SVM. We believe that this could be due to over-fitting by the SVM.

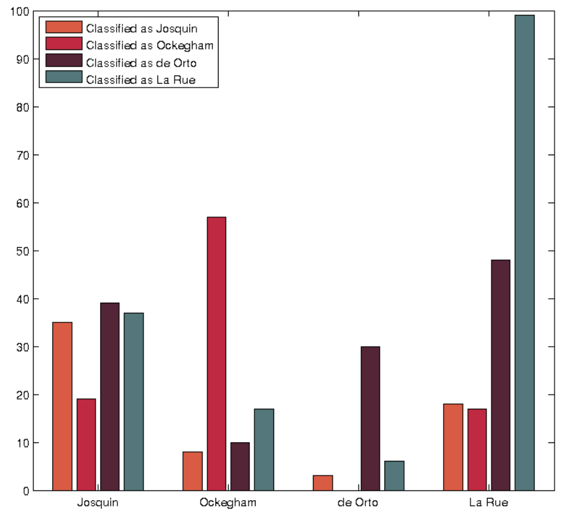

Fig. 2. Cross-validation classification using a naive Bayes classifier with pitch interval features. Accuracy: 50%

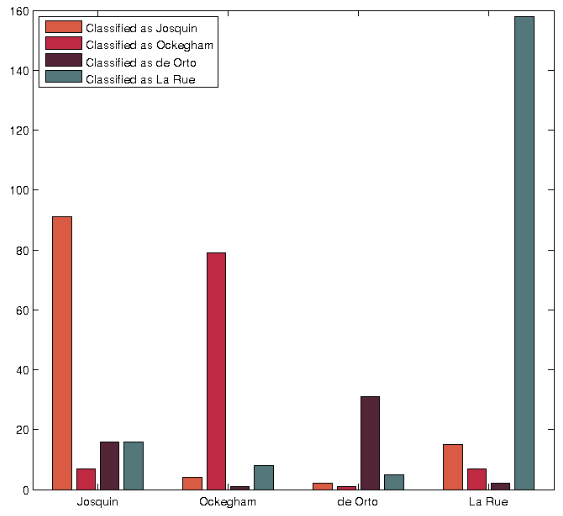

Fig. 3. Cross-validation classification using a naive Bayes classifier with counterpoint features. Accuracy: 81%

B. Binary Classification

Our binary classification model results are displayed and discussed in this section. Using 10-fold accuracy predictions, we plotted ROC curves for each of our binary classification models in Figures 5-8 below. The features of these models were counterpoint modules (using PCA where appropriate).

The classification trees in figure 5 performed the worst for our task by a large margin. The AUC of the trees (.6) was only slightly above random and significantly lower than the AUC of the other models. The Naive Bayes model and Gaussian-kernel SVM performed very similarly (AUC of .867 and .861 respectively). The results are close enough that the difference

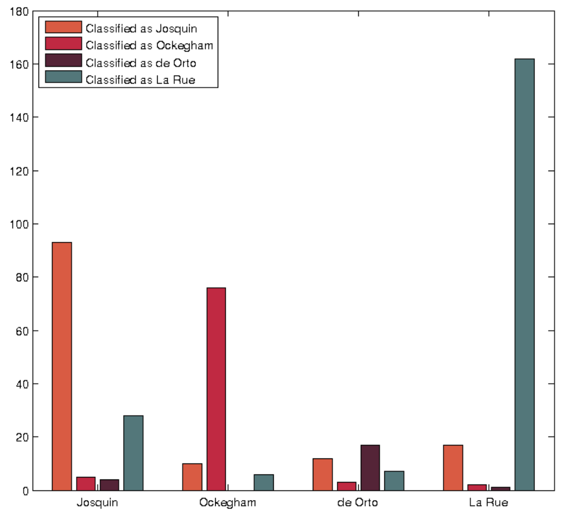

Fig. 4. Cross-validation classification using a Gaussian kernel SVM classifier with a PCA-reduced set of counterpoint features. Accuracy: 79%

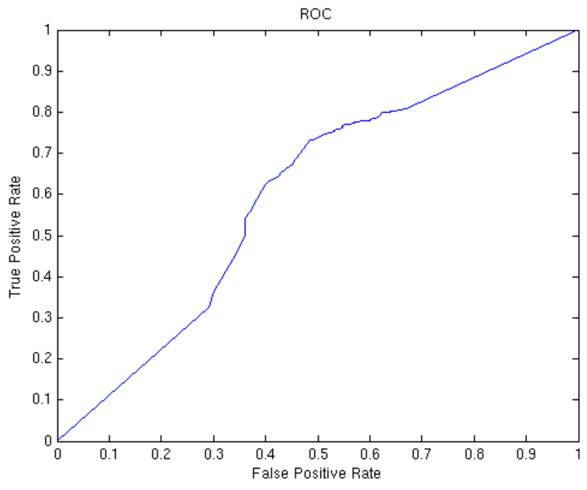

Fig. 5. Cross-validation ROC using classification trees with a PCA-reduced set of features. AUC: .598

in performance may be due to our selection of the cross-validation folds. Our ensemble method, with an AUC of .848, did not perform as well as the SVM or Naive Bayes model. We hypothesize that the difference in performance may be due our training methodology. Training a meta-learner requires data for training, meta-training, and testing. Because we have a limited sample of observations, each step in the meta-learner was able to train on less data than the individual models, providing a less accurate model. Thus, a direct comparison between the ensemble learner and the other models may not be entirely applicable. A summary of AUC values for our various binary classification models is presented below in Table II.

C. Classification on Unknown Data

Finally, we present the predictions of our models on the testing set. Figure 9 shows the number of pieces with

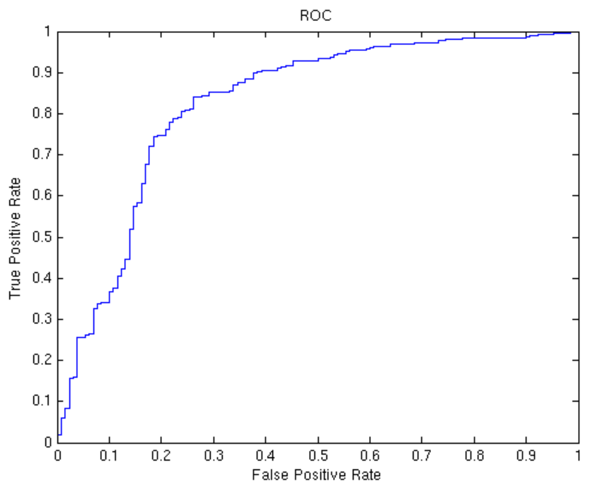

Fig. 6. Cross-validation ROC using Naive Bayes. AUC: .867

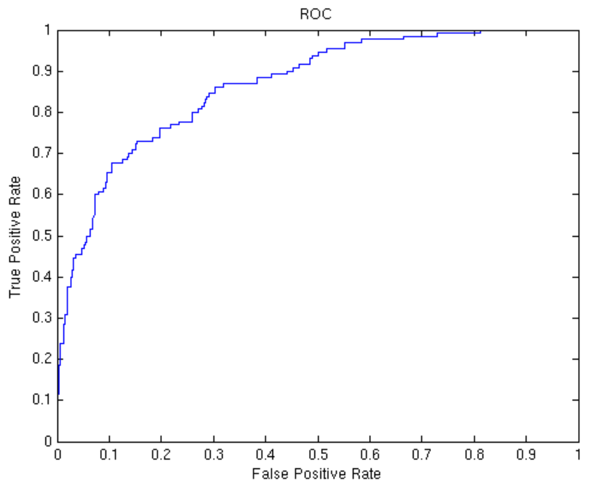

Fig. 7. Cross-validation ROC using a Gaussian Kernel SVM with a PCA-reduced set of features. AUC: .861

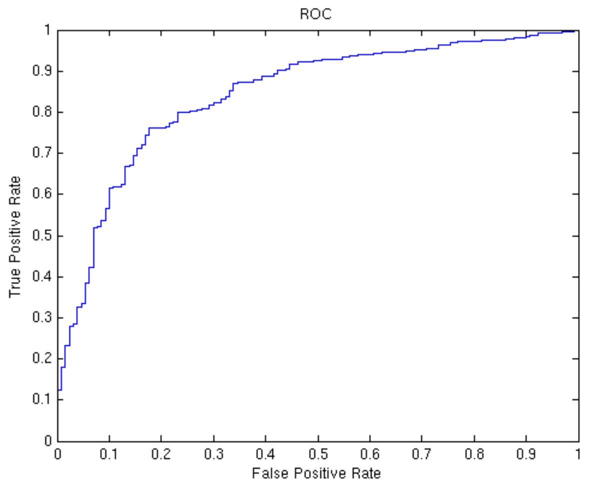

Fig. 8. Cross-validation ROC using a RobustBoost Ensemble Learner. AUC: .848

Learning Approach	Area Under Curve
Trees	.598
Naive Bayes	.867
SVM	.861
Ensemble	.848

TABLE II. BINARY CLASSIFIER PERFORMANCE COMPARISON

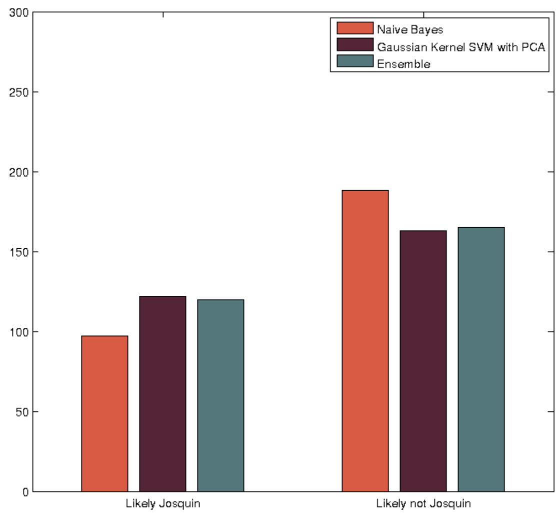

Fig. 9. Final classification of works with questionable attribution

questionable attribution that should be attributed to Josquin according to our classifiers. All three classifiers indicate that between one third and one half of the pieces have features similar to those in Josquin's body of known works.

VIII. CONCLUSIONS

The most important conclusion that can be drawn from this work is that the counterpoint module is a useful feature for classifying musical works by composer. This feature alone offered considerably more power for classification than any of the other features we considered, and in our final analysis, counterpoint modules were the only features used.

A Naive Bayes learning model performed at least as well as any of the other models that we attempted to use. This is somewhat surprising given that more complex devices such as SVMs are better able to model features that are interrelated with one another as we would expect the case to be with counterpoints. We believe that with the relatively small training set, the SVM tended to over-fit the data. When we used cross validation to examine performance, the SVM usually had very low training error, however, the estimate of the generalization error could not be brought below about 20%. The fact that the naive Bayes model performed so well may suggest that the occurrence of different counterpoint features is relatively independent of others. The ensemble learner also performed slightly worse than the Naive Bayes model. Further analysis is required to discern the underlying cause. We suspect it is due to the small amount of training data available and/or specific ensemble learner used.

Our results may also be used in the future as evidence when classifying pieces within the set of works with questionable attribution. This work can provide researchers with an estimate of the probability of each piece having been composed by Josquin.

IX. Future Work

Some extensions of this work include different selections of features and model approaches. Some features that we had considered implementing, but were unable to include were analyzing notes, rhythms, and dissonance on strong beats. In addition, increasing the range of intervals allowed may yield more information about songs.

Additionally, discussion with other groups that attempted to solve similar problems provided ideas about other feature approaches. First, at least one of the other groups did not use counterpoints, but did have success in accurately classifying some of the work. They used less complex features, but the key to their success was to use combinations of simple features determined by PCA. We believe that our prediction capability could be improved by integrating simple timing and rhythm features with our pitch-based counterpoint features.

Finally, an interesting continuation of this work may be a clustering based approach to the renaissance music. The music and features could be used to identify which composers were most similar, if they changed over time, and/or if any of the unknown music exhibit such trends.

Acknowledgment

The authors would like to thank Craig Sapp and Jesse Rodin of the Stanford Josquin Research Project for their guidance in the project, the project idea, and help with the data. Additionally, we would like to thank Dr. Andrew Ng and the T.A.'s for the opportunity to learn about machine learning.

References

[1] J. Rodin and C. Sapp, *Josquin Research Project* [Online]. Available http://http://josquin.ccarh.org/.

[2] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm

[3] John Milsom, "Analysing Josquin" *The Josquin Companion*, Oxford University Press, 2000.

[4] Peter Schubert,"Hidden Forms in Palestrina's *First Book of Four-Voice Motets*", *Journal of the American Musicological Society*, Vol 60, No.3 (Fall 2007) pp. 483-556.