Reveal Hidden Information in the Music Scores: Composer Attribution

CS 229 Milestone Report

Fang-Chieh Chou,* Yi-Hong Kuo,† and Hsiang-Yu Yang‡

(Dated: 11/15/13)

1. INTRODUCTION

Just like books are usually written in characteristic styles that can be used to attribute their authors, music scores contain rich information about the corresponding composers. With modern machine learning (ML) methods, it is possible to extract hidden information from the music scores to attribute the responsible composers. An intriguing composer attribution challenge is related to Josquin des Prez, one of the most famous composers of the Renaissance period. Due to his immense prestige, many anonymous works during the age were misattributed to Josquin. Only 31% of works attributed to Josquin has been verified by independent sources (secure works). For the rest of the works, there is not enough evidence to support whether or not they were composed by Josquin (unsecure works).

In this project, two-class classifications are implemented to solve the attribution problem. The classifiers were first trained and tested using the secure works by Josquin and by several contemporary composers to Josquin (Table I) as samples for negative class. We then applied the trained classifiers to the unsecure works.

2. METHODS

2.1. Machine learning classifiers and cross validations

In this project, we tested three ML methods: multinomial naive Bayes (NB), logistic regression, and linear Support Vector Machine (SVM). A dummy classifier that randomly gives positive class with 50% probability was set to the baseline performance. Cross-Validation is adopted to evaluate and compare learning methods. Because our input dataset contains only 28% positive samples, direct k-fold cross validation is not effective: it is possible that in some runs there are no positive samples. Instead, stratified random sampling validation is adopted. This approach allows an arbitrary number of iterations to reduce the fluctuation in validation statistics.

Table I: The music score dataset

	Number of works
Secure Josquin	130
Others	329
Unsecure Josquin	288

Due to the unbalanced nature of the dataset, and because our ultimate goal is to confidently identify Josquin's works in the unsecure set, it is not proper to optimize test error or accuracy. Instead, we would like to train a classifier with both high precision and high recall. F_1 score, which is the harmonic mean of precision and recall, gives a powerful single statistics to evaluate our models. For SVM and logistic regression, we also optimized the C-penalty parameter and the regularization strength by performing grid search on each parameter to maximize the F_1 score. The results in this report are all associated with optimized parameters.

We used Scikit-learn[1], a Python ML package, to implement all the classifications, cross validations and evaluations mentioned above.

2.2. Feature extraction

Currently, we have been testing two different sets of features which enable the recognizing of Josquin's works. In the first one, each note in music scores is treated as a "letter", characterized by its pitch and duration. Another feature set, related to the counterpoint technique, was used to quantify the relationship between two voices in the music scores.

To facilitate feature extractions, music21[2], a Python-based music processing package, is utilized to analyze digitized music scores from the Josquin Research Project[3].

2.2.1. Pitch-duration tuple

In this feature extraction method, each note is labeled by a tuple (p, d), where p is the pitch of the note in the MIDI representation, and d the note duration. The frequency of appearance of each pitch-duration tuple is used to construct the feature vector of a particular score.

2.2.2. Counterpoint feature

Although the pitch-duration tuples can be considered as "letters" in a composition, they give no information

*Electronic address: fcchou@stanford.edu
†Electronic address: stevekuo@stanford.edu
‡Electronic address: yanghy@stanford.edu

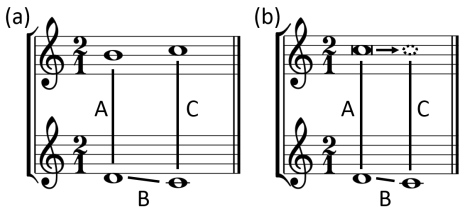

Figure 1: The counterpoint feature (a) The notes in the minimum set are of equal duration. (b)The notes have different time durations.

about the relationship between notes. The second set of features, called counterpoint feature, is used to overcome this limitation. Fig.1 shows two minimum sets of notes carrying both the melodic relationship, which is within one voice, and the counterpoint relationship, which is between voices. Each set of notes can be represented by a vector (A, B, C) indicating the intervals between these notes. To avoid complexity caused by the different modes (types of scale) of works, only the diatonic number of each interval was calculated and the quality, such as perfect, major or minor, of the interval was ignored. This feature can be considered as "words" in the composition and is the building block of polyphonic music. The frequency of appearance of each "word" was counted.

2.3. Feature preprocessing

Since the length and therefore the counts of each "word" are different for each works, using the raw "word counts" as features is not ideal for logistic regression and SVM. To resolve this issue, we use the term frequency–inverse document frequency (tf-idf) method, a common weighting scheme in text mining[4, 5], to reweight our feature vectors. Each feature vector is further normalized to have a unit L^2 norm.

For multinomial naive Bayes classifier, only raw counts were used since this classifier is designed to handle such multinomial "word counts" feature. Laplace smoothing is applied to address the sparsity of the features.

3. PRELIMINARY RESULTS

The accuracy, precision, recall, and F_1 score of each ML algorithm-feature combination are summarized in Table II. Based on the results, we found that the counterpoint feature consistently outperforms the pitch-duration feature. This result illustrates that it is the relationship between the notes, rather than the characteristics of individual notes, that differentiates the works of different composers. Moreover, by comparing different ML methods, we observed that logistic regression and linear SVM give better results than the multinomial naive Bayes algorithm for both feature sets. Between logistic regression and linear SVM, logistic regression tends to give higher precision in the cost of lower recall, whereas SVM gives a better precision-recall balance, especially for the counterpoint feature. In terms of F_1 score, linear SVM gives the highest scores for both feature sets. Therefore, the best combination is to use linear SVM with the counterpoint feature, which gives an F_1 score of 87.2%.

The trained classifiers were applied to analyze the unsecure dataset and the predictions are summarized in Table III. The logistic regression and the linear SVM with the counterpoint feature give more positive predictions than other classifiers, and our best classifier (SVM/counterpoint) gives 153 positive predictions. To further check the validity of our methods, we will discuss with our collaborators (Prof. Jesse Rodin and Prof. Craig Sapp in the Department of Music) to see if the predictions are consistent with the intuition of music experts.

4. DISCUSSION AND FUTURE WORKS

While our classifiers perform reasonably well, we believe there is still room for improvement. To evaluate the bottleneck of our classification methods, we plotted the test error and training error (evaluated by the F_1 scores) as functions of the training sample size (Fig.2). The linear SVM and the logistic regression, which are better classifiers, are slightly overfitted with high variance. Notably for the counterpoint feature, both linear SVM and logistic regression give zero training error even at the largest sample size tested. This also explains why the performance only got worse when we naively combined the two feature sets (results not shown), as we further overfitted the data. On the other hand, the naive Bayes classifier is under the high-bias regime, where even the training error itself is not satisfactory.

Since the size of our dataset is limited and cannot be easily enlarged, we need to reduce feature dimensions to resolve the high-variance problem. We will test feature selection and decomposition methods, such as recursive feature elimination, principal components analysis, or latent semantic analysis, as well as other methods inspired by basic music theory. For example, we will count compound intervals, or intervals larger than an octave, as corresponding simple intervals and therefore reduce the number of interval types. With reduced feature dimensions, there is a good chance that we can combine the counterpoint and pitch-duration features with reduced size to define a better classifier. Moreover, our earlier experiment (not shown here) shows that kernelized SVM significantly overfits the data, leading to worse performance than linear SVM. With reduced feature dimensions, we will also test if an SVM with different kernels leads to better performance.

In addition to feature reduction, we will also try to improve our models by combining more comprehensive features. First, based on our previous discussion, we know it is the relationship between notes, captured in the

Table II: Accuracy, precision, recall and F_1 score of each ML algorithm-feature set combination, compared with random guessing.

		Accuracy (%)	Precision (%)	Recall (%)	F_1 score (%)
Random guessing		50.0	39.6	50.0	44.2
Pitch-duration	NB	68.5	46.4	55.1	49.9
	Logistic	75.8	59.1	52.2	54.6
	SVM	74.9	55.4	64.6	59.2
Counterpoint	NB	87.4	79.2	76.8	77.4
	Logistic	92.3	90.9	81.3	85.5
	SVM	92.8	87.4	87.9	87.2

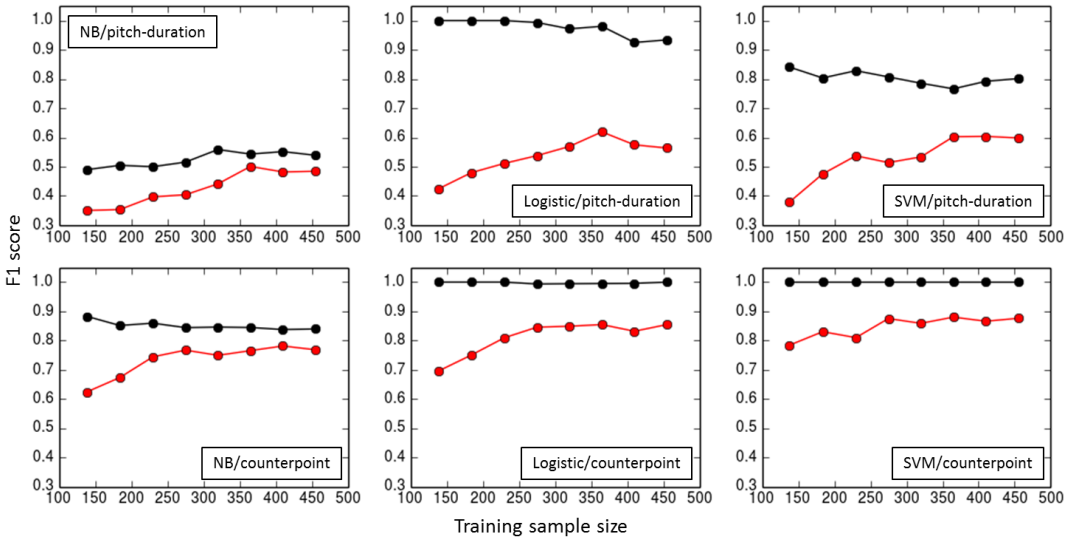

Figure 2: Learning curve of each ML algorithm-feature combination

Table III: Prediction on the unsecure Josquin dataset

		Positive	Negative
Pitch-duration	NB	102	185
	Logistic	104	183
	SVM	124	163
Counterpoint	NB	103	184
	Logistic	131	156
	SVM	153	134

counterpoint feature, that discriminates works of different composers. However, duration information is missing in the counterpoint feature set. We would like to expand our models to take the durations of neighboring notes into account. Second, it is well-known that Josquin liked to use repeated elements in his compositions. We would like to construct a new feature set which quantifies the use of motifs.

[1] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
[2] music21: A Toolkit for Computer-Aided Musicology http://web.mit.edu/music21/
[3] Rodin, J., The Josquin Research Project http://josquin.stanford.edu/
[4] Joachims, T. "Text Categorization with Support Vector Machines: Learning with Many Relevant Features" Machine Learning: ECML-98
[5] Salton, G., Buckley, C. (1988). "Term-weighting approaches in automatic text retrieval". Information Processing and Management 24 (5): 513–523.