Composer Style Attribution

Jacqueline Speiser, Vishesh Gupta

Introduction

Josquin des Prez (1450 1521) is one of the most famous composers of the Renaissance. Despite his fame, there exists a significant debate over which of his works are actually his and which have been misattributed to him. There are 336 pieces attributed to Josquin between 1488 and the early 17th century, a period extending far beyond his death. Unfortunately, there are very few surviving manuscripts from late 15th-century France, so most pieces do not have an exact historical record of when and where they were composed. This difficulty is compounded by the fact that Josquin reached the peak of his fame later in his life and after his death, at the same time that music was beginning to be printed more widely. As such, we cannot determine which works are his by the time of their publishing.

The Josquin Research Project in the Stanford Music Department has a large collection of music from the early Renaissance, including both verified and unverified works by Josquin and his contemporaries. There are 50 works identified through manuscript sources that are nearly guaranteed to belong to Josquin, and the remaining works range from known misattributions to likely to be by Josquin.

The goal of this project is to analyze the music of Josquin and his contemporaries (Johannes Ockeghem and Pierre de la Rue) to find a model that can classify the unknown works as Josquin's or not. In doing so, we hope to shed light on some of Josquin's controversial pieces and provide Renaissance music scholars with paths for further investigation.

Data

Our data set was provided by the Josquin Research Project, and can be found online.

Of the verified works in our data set, 130 were by Josquin, 93 were by Ockeghem, 183 were by la Rue. There were 287 unverified works by Josquin. Of these, Professor Rodin identified 10 examples as being most probably written by Josquin, and 10 as being most probably not written by Josquin. These 20 examples became our "test bench," by which we evaluated the performance of different models and sets of features.

From here, we will go through each step of our processing pipeline and explain why we made the decisions we did, then discuss the results of the classification problem.

Initial Thoughts

Before beginning to talk about the various feature extraction methods, there are many aspects of the project that deserve mention as they separate it from seemingly similar classification problems like genre classification, composer attribution in other periods of classical music, modern popular music, or across genres.

First, harmony, and specifically chords and their usages is a common metric used in later periods of music. However, in the late 16th century, when Josquin's music was popular, functional harmony as it is interpreted and used now had not been invented yet. Composers rarely thought "vertically" about the music they were writing, and felt that each voice had to have enough variance and color to it that it would be pleasing to listen to by itself. This though process is quite different from modern music, where especially the bass holds one note as a pedal for multiple measures, or employs ostinato for rhythmic effect.

Second, Renaissance music was extremely strict in its control over the use of intervals. The rules of counterpoint severely limited the range of a composer's artistic freedom (as compared to the almost limitless possibilities of today's music). It's fascinating that composers managed to create such beautiful music even in within the structure of counterpoint, but it makes telling apart two composers of the time period very difficult, even for a human being.

Finally, music is an incredibly dense informational system. A lot happens within one bar of music, and the simplest of analyses has hundreds, or even thousands of features. Compounded to this fact is that composers write for different modes, each with its own characteristics. We only have a few hundred examples, so the danger of over fitting is quite acute in this classification problem. The works themselves also tend to be shorter, unlike the symphonies and operas of later times, leading to more variance in the data, and very, very sparse features.

Features

After some debate, we decided to pose the problem as two independent problems, as we felt each composer was distinct enough that grouping them together would only "dilute" any hypothesis our learning algorithms would make. So, we had two classification problems: Josquin vs Ockeghem and Josquin vs Pierre La Rue.

With the knowledge that we had to keep our feature vectors relatively small, we decided to limit the extraction to four categories: frequencies of the individual notes, frequencies of pairwise interval combinations between each of the voices, markov transition matrices for the rhythms of the pieces, and markov transition matrices of the pitches in each piece. Effectively, these can be though of as 0-order and 1-order markov chains of pitch and rhythm events in the pieces.

One difference in our frequency histograms from most of the literature we found was that rather than consider the number of attacks of each note (i.e., count the number of times it appears in the music), we used the percentage duration of that pitch in the piece as our feature. We felt this corrected for the variance caused by different themes in a piece (which would highlight different notes), and generally said more about the "sound" being utilized by a composer than the number of appearances did.

When comparing Josquin and Ockeghem, the pitch histogram had 248 features, the interval histogram had 682, and the Markov rhythm matrix had 303. The Markov pitch features had ∼2700 features.

As the music of the Renaissance followed strict rules, many of our extracted features were virtually the same across composers. For example, it's simply a rule that the 7th tone of the scale always resolve to the tonic. This was reflected in the data as $p(\text{tonic}|\text{7th}) = 1$ for all composers (markov pitch feature). The goal, then, was to find the features that best *separated* Josquin from his contemporaries.

Feature Processing

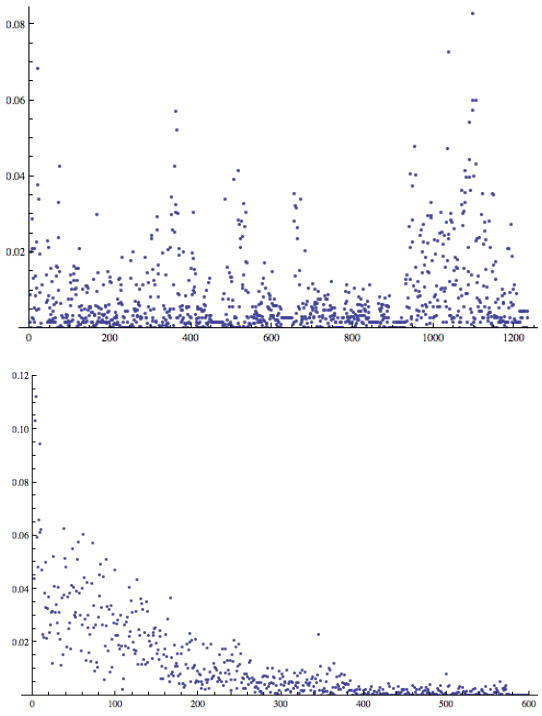

Figure 1: top: raw scores, bottom: pca scores

In order to discover which features were the most relevant in telling apart Josquin and his contemporaries, and to reduce the number of features we needed to consider, we set up a heuristic to score all 3000 of our features. Our choice of heuristic was the mutual information of each feature and the class (0 or 1). There was one slight twist - each $x^{(i)}$ fell into more or less an exponential distribution, being a normalized percentage, since having a high percentage of the note G precludes the presence of any other note. Similarly, transitioning from quarter note to whole note more often automatically means there's less transitions between quarter note and half note. Furthermore, being a percentage, each $x^{(i)}$ was continuous. Rather than fit a gaussian function to the 1 and 0 classes and integrate over them, we decided to discretize the distribution by binning using the value of sqrt(x) rather than just x, in order to make the bins a little more even.

After running feature selection, and using the top 50 features (we just picked this number, no fancy cross validation involved, yet), we saw that the top 50 features were full with rhythmic movement of the bass voice (so markov transitions of rhythm of the bass voice) and intervals between the bass and the soprano (or in other words the melody). Our findings with regards to the most important features leave much analysis and investigation to be desired - there's a lot of intuition to be gained from what the computer thought the most important aspects of the piece were.

Running GDA with feature selection gave us 10% training error and 3/20 wrong on the test bench (note, this was a post-process run; an ablative test of sorts; the later PCA results compare to before

we implemented the entire pipeline).

At this point, Principal Components Analysis seemed like the next step - there's a natural dependency and correlation between most of the features we extracted, and PCA tries its best to remove those dependencies and present uncorrelated, independent features. We ran PCA and selected the top 50 features (PCA gives us features in order of maximum variance). The benefits of PCA are very visible upon viewing Figure 1.

The PCA algorithm rotates the features so that the best scoring features are now present in the top 50. Furthermore, while only a few of the top features break the 0.05 boundary in feature analysis, points in the PCA space break even 0.1 or 0.12, which is a 30% score increase. Given that overfitting is a critical issue, this feature "condensation" was extremely beneficial. Upon performing PCA analysis, our error dropped from 30-40% to 12-13%. The full results are detailed in the next section.

Algorithms and Cross-Validation

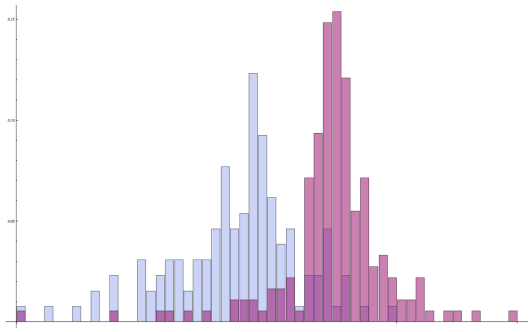

Figure 2: the 7th highest variance feature - blue:Josquin, red:La Rue

Our initial learning algorithm was Naive Bayes, since it was easy to implement. However, Naive Bayes exhibited about 30-40% training error, which was unacceptable. However, the results aren't surprising, since the independence assumption is horrible for something so interdependent as music.

To select the best model to represent our data, we used 10-fold cross validation on both the Support Vector Machine (SVM) Gaussian discriminant analysis (GDA) learning algorithms. The models with the best training error were used to calculate testing error and run on the unknown data. We chose to use SVM with a Gaussian kernel because the distribution of the features tended to be Gaussian after PCA analysis (see Figure 2). We generally thought that the support vector model would be a good way to separate the two composers - looking for a subset of the pieces that were "difficult" and then forming a margin between them would ignore the extraneous outliers in the feature set. GDA seemed natural considering the shape of the features.

Our algorithms' training errors are detailed in the following tables:

Legend: La Rue		Frequency Histograms		Markov Transitions	
True Negative	False Positive				
False Negative	True Positive				
SVM		86.81%	13.19%	87.36%	12.64%
		10.00%	90.00%	10.00%	90.00%
		error: 11.86%		error: 11.54%	
Gaussian Discriminant Analysis		96.15%	3.85%	87.91%	12.09%
		11.54%	88.46%	15.38%	84.62%
		error: 7.05%		error: 13.46%	

(a) La Rue training error

Legend: ockeghem		Frequency Histograms		Markov Transitions	
True Negative	False Positive				
False Negative	True Positive				
SVM		94.57%	5.43%	97.87%	2.17%
		14.62%	85.38%	3.08%	96.92%
		error: 10.76%		error: 2.70%	
Gaussian Discriminant Analysis		94.57%	5.43%	96.47%	3.26%
		7.69%	92.31%	4.62%	95.38%
		error: 6.76%		error: 4.05%	

(b) Ockeghem training error

The results seem very promising, if a little confusing. Detailed here are the values of error obtained by running the frequency histograms (first column) on Ockeghem and La Rue, respectively, and the error values obtained with the first order markov transitions along with the frequency histograms. The significance of both of these composers is as follows: Ockeghem is a composer from the time period before Josquin Some aspects of the results are baffling -

for example adding markov transition features improved the results for Ockeghem, but hurt the results for La Rue. Part of the inconsistencies in behavior can be attributed to the fact that we only took the top 50 variance features from PCA, and not the top 50 scoring features.

Testing Benchmark Results

We ran the algorithms on the 20 pieces provided by Professor Rodin.

When trained on La Rue's data:

GDA correctly labeled the 10 incorrect examples as incorrect. However, it classified 3 of the 10 examples that were probably Josquin's pieces as being Josquin. In all cases, GDA's decisions were very heavy-handed. The probabilities with which the decisions were made were either 80-90+% Josquin or 70-90+% not Josquin.

SVM actually performed the exact same classifications as GDA, but didn't output any probabilities. We assume that the three pieces both algorithms were getting wrong constitute irregularities of some kind. Either way, they deserve investigation.

When trained on Ockeghem's data:

GDA got 4 of the not-Josquin examples wrong, and SVM got 5 of them wrong. The probabilities were closer to the middle and more varied, indicating the algorithm isn't as sure about its decisions. Not surprisingly, GDA and SVM actually classified all but 1 of the Josquin examples as being Josquin. However, this isn't surprising, as Ockenghem was very different from Josquin and any classifier with only knowledge of pieces that came from a previous time period would be much more aggressive about classifying something as Josquin.

Although not perfect, the results inspire confidence in our choice of methods. We think with a little more coaxing, machine learning could prove to be useful tool in deciding which composer wrote which Renaissance piece.

Future Steps

One thing that would be simple to implement and add to the pipeline would be feature selection *after* pca. While the top 50 PCA points have on average high scores, it would be much more beneficial to choose the 50 highest scoring PCA variables. PCA only increases the variance of the variable as a whole, and disregards the distance in between the two gaussian functions of the variables. It's pretty clear from the graph (which we unfortunately generated much later than we should have) that the top 50 variance $\neq$ top 50 scores.

Alternatively, it may have been a better idea to run PCA on only the training features, then recover the original vectors, and perform the same transformation on the test data set. It's unclear whether maximizing the variance of the test set helped us distinguish the unknown pieces.

Then, we could have run the SVM with some "play" (as shown in class by adding a constant) so that outliers would be ignored. However, it didn't seem prudent to start there, since we know of only 130 verified Josquin sources, and they ALL should be classified as being Josquin's, even if they are outliers. It was also difficult to know that any outliers existed or that they should be labeled as such given the complexity and variance of pieces produced by any one composer.

On the features side, other papers on similar subjects have achieved good results using 3rd order markov transitions of pitch and rhythm considered *together* as opposed to separately. However, the number of features this generates is absolutely astronomical and the authors of that paper did a lot of work to trim down the number of features they were dealing with. This extended treatment of the subject was outside the scope of a one quarter class, but deserves future investigation.

Also, Professor Craig Sapp has a method of storing the counterpoint transitions between each set of intervals in a melody. Again, this method has been shown to provide great insights into the structure of Renaissance music, but also makes for an extremely large amount of features, and it wasn't clear we'd have the time to implement the processing tools to trim down those features in time. Also, we felt that the insight gained from these simple features was a little more tractable - the frequency of G makes more sense than the probability of the chain: m3, P4, m3 $\rightarrow$ P5. The latter is a little harder to make sense of.

Acknowledgments

We'd like to thank the Stanford CCARH Center and the Josquin Research Project for their support throughout this process, from giving us the data and initial insights about the pieces we needed to culling an appropriate set of scores as a good test bench. Professors Jesse Rodin and Craig Sapp were instrumental in the success of this project.

We'd also like to thank Professor Andrew Ng and the CS229 staff for an enriching quarter, and for giving us the toolset to accomplish so much in such a little time. We never thought we'd have the knowledge to figure out which Renaissance composer wrote which piece in only 10 weeks of exposure to the material.

Sources

Rodin, J. (n.d.). Josquin and epistemology.

Wolkowicz, J., Kulka, Z., & Keselj, V. (n.d.). N-gram-based approach to composer recognition.

Liu, Y.-W. (2002, June). Modeling music as Markov chains: Composer identification.