CLASSIFIER COMBINATION FOR CAPTURING MUSICAL VARIATION

Jeremy Pickens
Department of Computer Science
King's College London[*]
London WC2R 2LS, England
jeremy@dcs.kcl.ac.uk

ABSTRACT

At its heart, music information retrieval is characterized by the need to find the *similarity* between pieces of music. However, "similar" does not mean "the same". Therefore, techniques for approximate matching are crucial to the development of good music information retrieval systems. Yet as one increases the level of approximation, one finds not only additional similar, relevant music, but also a larger number of not-as-similar, non-relevant music. The purpose of this work is to show that if two different retrieval systems do approximate matching in different manners, and both give decent results, they can be combined to give results better than either system individually. One need not sacrifice accuracy for the sake of flexibility.

Keywords: Classifier Combination, Approximate Matching

1 INTRODUCTION

It is a well-known result, due to work such as Schapire (1990) and Tumer and Ghosh (1999) that if multiple classifiers each gives results better than random, one can achieve results better than each classifier individually by combining their classification hypotheses. In this paper, we focus on ranked list classifiers, or classifiers that make some sort of judgement about how relevant or non-relevant a piece of music is to a query and then rank by this judgement.

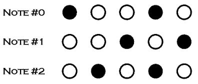
In music information retrieval, we are looking not for exact matches, but for similarity. As a result, music information retrieval systems need to be approximate in their search for matches. However, as one increases the level of approximation, one not only finds more relevant music pieces, but more non-relevant ones as well.

There are two main approaches to approximate matching. One can do "exact matching on fuzzy data, or fuzzy matching on exact data" Wiggins (2005). We present two "fuzzy" music retrieval systems: Markov Random Fields models and Harmonic models. Each of these systems does its approximation in a slightly different manner, finding many non-relevant pieces alongside the relevant ones. We show that by combining the results given by each system, we can improve upon the results available through either system.

As a result of this combination, we show that one need not sacrifice precision to obtain better recall. Thus, one can confidently build new systems that are more flexible in their approximations, knowing that through classifier combination the variations one is seeking can be successfully captured.

2 MUSIC REPRESENTATION

For these experiments, we use a 12-pitch class, octave-invariant, event-based symbolic representation. For example:



This is a "polyphonic" sequence of notes, with time along the x-axis and pitch along the y-axis. All the notes that start at the same time are arranged into the same vertical slice. For the purpose of this paper, durations of notes as well as time between notes, is ignored.

3 MARKOV RANDOM FIELD MODEL

A Markov Random Field is a model that will allow us to predict the value of a current note $n_{i,t}$ from the values of the surrounding variables (notes). In other words, it is an estimated probability distribution $P(n_{i,t}|H_{i,t})$, where $H_{i,t}$ is the set of all variables within some fixed distance previous to time t, or notes that occur at time t, but have an index lower than i.

In general, a random field framework allows arbitrary dependencies (or *features*) between the target $n_{i,t}$ and its neighborhood $H_{i,t}$. In this work, we deliberately restrict allowed dependencies to binary questions of the form: "*was note j played at time s before t?*". We also allow

[*] This work is supported by EPSRC grant GR/N63123/01

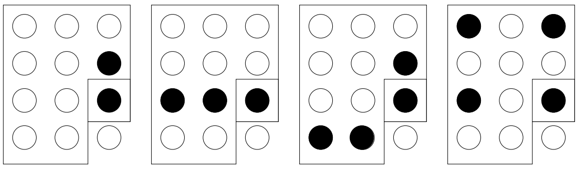


Figure 1: Examples of musical features that may be induced to predict the probability of note 2 being played at time t. Black circles represent notes that are part of the feature function. Boxed black circle denotes the note $n_{2,t}$. Boxed area represents the history $H_{2,t}$. From left to right, the features are: $\{n_{2,t}\ n_{1,t}\}$, $\{n_{2,t}\ n_{2,t-1}\ n_{2,t-2}\}$, $\{n_{2,t}\ n_{1,t}\ n_{3,t-1}\ n_{3,t-2}\}$, $\{n_{2,t}\ n_{0,t}\ n_{2,t-2}\ n_{0,t-2}\}$

generalizations where a question is asked about some subset S of the notes in the allowed history $H_{i,t}$. The answer to a question of this form will be called the feature function f_S and S will be referred to as the *support* of f. For a given support $S \in H_{i,t}$, the feature function f_S is defined as the conjunction of answers about the individual notes in $n_{j,s}{\in}S$:

$$f_S(n_{i,t}, H_{i,t}) = n_{i,t} \prod_{n_{j,s} \in S} n_{j,s} \tag{1}$$

Defined in this manner, our feature functions are always boolean, and equal to 1 if all the notes defined by S were played before the target note $n_{i,t}$. Features are also time-invariant (does not matter in which onset time they occur) but are not index invariant (are not transposable up or down any semitones). As an illustration, Figure 1 contains some examples of features that could have an impact on note “2” at time t.

The parametric form that characterizes the random field model is a member of the exponential (or log-linear) family, expressed as:

$$\hat{P}(n_{i,t}|H_{i,t}) = \frac{1}{Z_{i,t}} \exp\left\{ \sum_{f \in \mathcal{F}} \lambda_f f(n_{i,t}, H_{i,t}) \right\} \tag{2}$$

In equation (2), the set of scalars $\Lambda = \{\lambda_f : f \in \mathcal{F}\}$ is the set of Lagrange multipliers for the set of structural constraints $\mathcal{F}$. Intuitively, the parameter λ_f ensures that our model predicts feature f as often as it should occur in reality. $Z_{i,t}$ is the normalization constant that ensures that our distribution sums to unity over all possible values of $n_{i,t}$. In statistical physics, it is known as a *partition function* and is defined as follows:

$$Z_{i,t} = \sum_n \exp\left\{ \sum_{f \in \mathcal{F}} \lambda_f f(n, H_{i,t}) \right\} \tag{3}$$

The exact manner in which the features $f \in \mathcal{F}$, scalars $\lambda \in \Lambda$ and $Z_{i,t}$ are chosen is beyond the scope of this poster, but is covered in Pickens and Iliopoulos (2005). However, the basic idea is that there is a target empirical distribution $\tilde{P}(n|H)$ given by a piece of music, and the goal is to fit the model $\hat{P}(n|H)$ to this target.

Music retrieval using these models is done by estimating a model using a given query as a target distribution, and then observing how well that query model predicts the notes in each document in the collection.

$$\sum_H \tilde{P}_D(H) \sum_n \tilde{P}_D(n|H) \log \hat{P}_Q(n|H) \tag{4}$$

In other words, our similarity measure is the expected *cross-entropy* between the empirical distribution $\tilde{P}_D(n|H)$ of the document and the estimate $\hat{P}_Q(n|H)$ produced by the query model, as given by equation 4.

4 HARMONIC MODEL

While the random field models operate on note conjunction features, the harmonic models developed by Pickens and Crawford (2002) work by mapping each 12-dimensional note onset vector s onto a 24-dimensional chord vector of the 12 major and 12 minor triads. This ad hoc mapping takes into account not only the size of the overlap between the note and the chord, but also the total number of notes in the simultaneity, and the Krumhansl and Shepard (1979) perceptual distance between the chords in the lexicon:

$$Context(s, c) = \frac{|s \cap c|}{|s|} \sum_{c' \in lexicon} \frac{|s \cap c'|}{(|s| * Krum(c', c)) + 1} \tag{5}$$

This context score is computed for every chord c in the lexicon. Additionally, inter-vector smoothing is performed, whereby neighboring vectors are allowed to contribute to the partial observation of the current vector. A vector of partial observations is then obtained by normalizing by the sum total:

$$PartialObs(s, c) = \frac{Context(s, c)}{\sum_{c' \in lexicon} Context(s, c')} \tag{6}$$

This vector of partial observations over the chord lexicon is then used as the raw feature set for model estimation. For example, suppose we have a lexicon of three chords, P, Q, and R. A sequence of partial observation vectors might appear as follows:

	Partial observation vectors				
Chord	1	2	3	4	5
P	0.2	0.1	0.7	0.5	0
Q	0.5	0.1	0.1	0.5	0.1
R	0.3	0.8	0.2	0	0.9
Total	1.0	1.0	1.0	1.0	1.0

We simply count the number of length m sequences through a piece of music, each count weighted by the fractional observation amount. Continuing our example, suppose $m = 2$. We begin with the window from timestep 1 to timestep 2. The sequence P$\rightsquigarrow$P is observed in proportion to the amount in which we observe P at timestep 1 and also observe P at timestep 2 (0.2 * 0.1 = 0.02). The sequence Q$\rightsquigarrow$R is observed in proportion to the amount in which we observe Q at timestep 1 and then R at timestep 2 (0.5 * 0.8 = 0.4), and so on.

We next divide these chains into two parts, the “previous state”, or history, and the “current state”. We define the history H as the first m-1 chords in the sequence, and the current state c as the final chord in the sequence. For example, with an m=2 chain "P⇝Q", the history is the state "P" and the current state is "Q". With an m=3 chain "P⇝Q⇝P", the history is the state "P⇝Q" and the current state is "P"

We obtain parameters for the conditional probability distribution $\hat{P}(c|H)$ by doing maximum likelihood estimation using the complete set of extracted chains, where $|H, c|$ is the number of times the sequence with history H followed by chord c is observed.

$$\hat{P}(c|H) = \frac{|H, c|}{\sum_{H_i} |H_i, c|} \quad (7)$$

Prior to retrieval, at indexing time, we estimate $\hat{P}(c|H)$ for every piece of music in the collection. At retrieval time, when presented with a query, we estimate a model for the query in the exact same manner. Similarity is calculated between the query model and every document model in the collection using the Kullback-Leibler (KL) divergence measure, also known as relative entropy. Pieces are ranked by increasing dissimilarity to the query.

5 CLASSIFIER COMBINATION

Our approach is among the simplest possible. Combining the actual similarity score given by each system is difficult, because this score means something completely different depending on the system. Instead, we combine the ranks that each system gives, by giving each piece of music a new score equal to the average of the two ranks given by each system. Pieces are then reranked by this average score. Ties, if any, are broken by randomly ordering all pieces with that same score.

If both systems rank a piece highly, it will continue to be ranked highly in the combined ranking. If one system gives a high rank and another a low rank, it will not fare as well. The intuition is that, because the two systems differ in their approximation methods, non-relevant pieces that happen to be given a high rank under the peculiarities of one system will not fare as well under the other. Pieces which truly are relevant will receive decent, though not perfect, rankings under both methods, and will therefore percolate to the top, in the combination.

6 EXPERIMENTS

For our project, we have four collections. The first is a set of approximately 3000 polyphonic music pieces from the CCARH at Stanford. These are mostly baroque and classical pieces from Bach, Beethoven, Corelli, Handel, Haydn, Mozart and Vivaldi. Our remaining three sets of music, on the other hand, are pieces which were intentionally composed as variations on some theme. These are 26 variations on the tune known to English speakers as 'Twinkle, twinkle, little star', 75 versions of John Dowland's 'Lachrimae Pavan' from different 16th and 17th-century sources, and 50 variations by four different composers on the well-known baroque tune 'Les Folies d'Espagne'.

For retrieval, we select a piece from the three sets of variations and use that as the query. All other pieces from that same variation set are judged relevant to the query, and the rest of the collection is judged non-relevant. This process is repeated for all pieces in all three sets of variations, for a total of 151 queries.

We define Θ_{MRF} as the retrieval system based on random fields. $\Theta_{HARM=2}$ denotes a retrieval system based on harmonic models with the chord sequence length set to 2 (and a small smoothing window), while $\Theta_{HARM=3}$ is a length 3 chord sequence and a larger smoothing window. Finally, $\Theta_{MERGE=2}$ and $\Theta_{MERGE=3}$ denote the combined ranked lists of Θ_{MRF} and Θ_{HARM}, with the chain set to 2 or 3, respectively. Figure 2 shows the results. Percentage improvements are given for the Θ_{MERGE} lists over each of the other two lists. An asterisk indicates statistical significance (t-test at a 0.05 level).

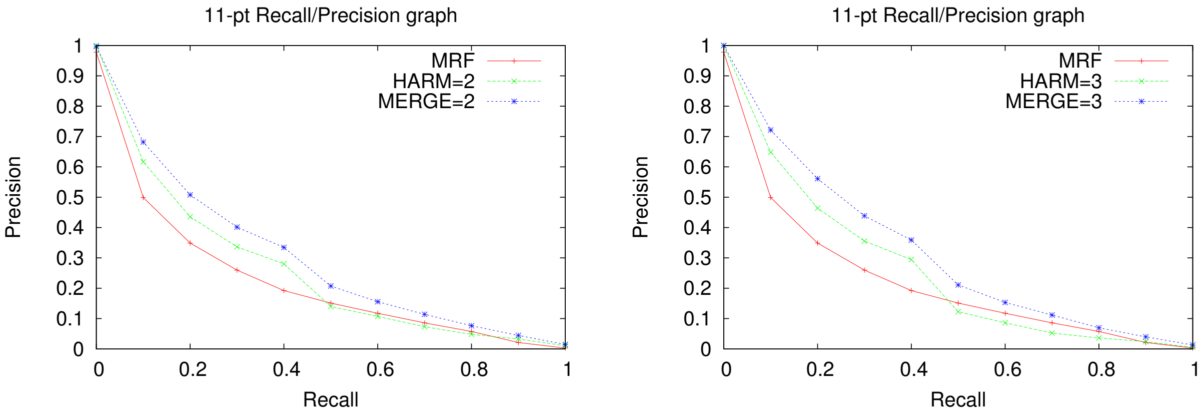
7 ANALYSIS

The first thing we note is that the classifier combination works. For both chain lengths, the combination yields anywhere from a 20% to a 40% or more average improvement over either the harmonic model or the random field model alone, and the improvement is statistically significant at every level of recall. Total number of relevant documents in the top 1000 decreases slightly (≈5%) when compared against Θ_{MRF}, but it still an ≈20% improvement over the Θ_{HARM} models.

But why does it work? MRF model estimation involves adding thousands of features to the model, each of which pinpoints some exact subset of notes and gives a weight to the relative importance of that subset. The idea is that if a variation has a few notes missing, hundreds of these feature functions will no longer be activated. Yet thousands more features will still be activated, along with their weights, and thus the overall prediction accuracy of the current note will still be good. However, the system still makes mistakes in that spurious occurances of notes sometimes activate highly-weighted feature functions, and thus pull in non-relevant pieces.

Harmonic models, on the other hand, smear out note observations into chords. Relevant variations which do not use the exact same notes might use notes from the exact same chord. By using triads as features, rather than the notes themselves, one can find these variations. However, the system still makes mistakes in that there are non-relevant pieces that are harmonically similar, without actually being a "true" variation on the query.

Each of these systems makes its relevance-classification decision using a different technique. Both do a good job of finding relevant variations, but also make mistakes. However, they make mistakes for different reasons. If the mistakes were as consistent as the successes, i.e. if the same non-relevant pieces were always ranked highly by both systems, we would not expect to see the consistent improvement in precision over both systems at all levels of recall that the merged list provides.



	Θ_{MRF}	$\Theta_{HARM=2}$	$\Theta_{HARM=3}$	$\Theta_{MERGE=2}$			$\Theta_{MERGE=3}$		
					%Change (Θ_{MRF})	%Change ($\Theta_{HARM=2}$)		%Change (Θ_{MRF})	%Change ($\Theta_{HARM=3}$)
Retrieved:	151000	151000	151000	151000			151000		
Relevant:	8801	8801	8801	8801			8801		
Rel|ret:	7239	5818	5650	6884	-4.90*	+18.32*	6832	-5.62*	+20.92*
Interpolated Recall - Precision									
at 0.00	0.9781	1.0000	1.0000	0.9967	+1.9*	-0.3	1.0000	+2.2*	0.0
at 0.10	0.4990	0.6166	0.6484	0.6818	+36.6*	+10.6*	0.7215	+44.6*	+11.3*
at 0.20	0.3488	0.4352	0.4638	0.5078	+45.6*	+16.7*	0.5610	+60.8*	+20.9*
at 0.30	0.2597	0.3363	0.3553	0.4015	+54.6*	+19.4*	0.4387	+69.0*	+23.5*
at 0.40	0.1923	0.2803	0.2945	0.3344	+73.9*	+19.3*	0.3589	+86.6*	+21.9*
at 0.50	0.1510	0.1398	0.1226	0.2072	+37.3*	+48.2*	0.2109	+39.7*	+72.0*
at 0.60	0.1174	0.1073	0.0861	0.1555	+32.4*	+45.0*	0.1531	+30.3*	+77.8*
at 0.70	0.0857	0.0730	0.0523	0.1140	+33.0*	+56.1*	0.1116	+30.2*	+113.4*
at 0.80	0.0575	0.0480	0.0362	0.0763	+32.6*	+58.8*	0.0698	+21.5	+92.7*
at 0.90	0.0210	0.0328	0.0239	0.0445	+111.6*	+35.8*	0.0397	+88.7*	+65.8*
at 1.00	0.0029	0.0106	0.0053	0.0147	+401.5*	+37.8*	0.0130	+343.4*	+146.6*
Average precision (non-interpolated) over all rel docs									
	0.2054	0.2360	0.2476	0.2859	+39.22*	+21.7*	0.3061	+49.05*	+23.61*

Figure 2: Recall-Precision results. In the Θ_{MERGE} columns, results are given for the combination of the Θ_{MRF} and corresponding-sized Θ_{HARM} model, followed by the percentage improvement over both individual systems.

8 CONCLUSION

We have shown in this paper that two good polyphonic retrieval systems can be combined to become even better. Though the systems evaluated were for symbolic data, this has serious implications for much of music retrieval. Music retrieval systems typically do not work by finding exact matches, no matter if the data is symbolic or audio. Music is fluid and changing, and there are no hard rules about what constitutes a “variation”. Some degree of approximation is always going to be needed.

In the future we hope to test an assortment of other boosting and classifier combination techniques, not just ranked list averaging. We also hope to test a number of other matching systems beyond random fields and harmonic models. However, the point of this short paper is not the specifics of any one boosting technique, or even any one retrieval technique. It is simply to show that we need not build all-encompassing retrieval systems to do approximate matching or variation finding. By building a host of systems, each of which tackles the variation problem from a slightly different angle, and then combining them, future music retrieval systems will be able to gain the flexibility of finding more pieces without sacrificing accuracy, and thus better capture musical variation.

REFERENCES

C. L. Krumhansl and R. N. Shepard. Quantification of the heirarchy of tonal functions within a diatonic context. *Journal of Experimental Psychology: Human Perception and Performance*, 5:579–594, 1979.

J. Pickens and T. Crawford. Harmonic models for polyphonic music retrieval. In *Proceedings of the ACM Conference in Information Knowledge and Management (CIKM)*, McLean, Virginia, November 2002.

J. Pickens and C. Iliopoulos. Markov random fields and maximum entropy for music information retrieval. In *6th Annual International Conference on Music Information Retrieval (ISMIR)*, September 2005.

R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

K. Tumer and J. Ghosh. Linear and order statistics combiners for pattern classification. In A. Sharkey, editor, *Combining Artificial Neural Nets*, pages 127–162. Springer Verlag, 1999.

G. Wiggins. Personal conversation, London SE14, January, 2005.