MULTIPLE VIEWPOINTS MODELING OF TABLA SEQUENCES

Parag Chordia
Georgia Tech
Center for
Music Technology
Atlanta, GA, USA
ppc
@gatech.edu

Avinash Sastry
Georgia Tech
Center for
Music Technology
Atlanta, GA, USA
asastry3
@gatech.edu

Trishul Malikarjuna
Georgia Tech
Center for
Music Technology
Atlanta, GA, USA
tmallikarjuna3
@mail.gatech.edu

Aaron Albin
Georgia Tech
Center for
Music Technology
Atlanta, GA, USA
aalbin3
@mail.gatech.edu

ABSTRACT

We describe a system that attempts to predict the continuation of a symbolically encoded tabla composition at each time step using a variable-length n-gram model. Using cross-entropy as a measure of model fit, the best model attained an entropy rate of 0.780 in a cross-validation experiment, showing that symbolic tabla compositions can be effectively encoded using such a model. The choice of smoothing algorithm, which determines how information from different-order models is combined, is found to be an important factor in the models performance. We extend the basic n-gram model by adding viewpoints, other streams of information that can be used to improve predictive performance. First, we show that adding a short-term model, built on the current composition and not the entire corpus, leads to substantial improvements. Additional experiments were conducted with derived types, representations derived from the basic data type (stroke names), and cross-types, which model dependencies between parameters, such as duration and stroke name. For this database, such extensions improved performance only marginally, although this may have been due to the low entropy rate attained by the basic model.

1. INTRODUCTION AND MOTIVATION

When listening to music, humans involuntarily anticipate how it will continue [8]. Such expectations help to process information efficiently, as well as allowing complex, noisy stimuli to be accurately interpreted. For musicians, this anticipation is essential for synchronization and harmonization. In this paper, we explore a computational model of this predictive process based on an ensemble of n-gram models. Specifically, we examine whether such a model can successfully represent the structure of symbolically encoded tabla compositions. Our motivation for building a

predictive tabla model is to enable more intuitive modes of interaction between musicians and computers.

In addition to this practical goal, we hope to work towards developing a computational model of musical anticipation. Previous work [11] on Western melodies showed that human judgments of melodic continuation were highly correlated with a variable-length n-gram model. Although we will not address human subject data here, we hope to provide converging evidence from a markedly different musical tradition (tabla), that syntactic structure can be efficiently represented using an n-gram modeling approach.

2. BACKGROUND AND RELATED WORK

Markov and n-gram models have been extensively used to model temporal structure in music [1]. They have been extensively used in algorithmic composition, timbral analysis [2] [7], structure analysis [12], and music cogniton [14].

Markov models are based on a succession of states. In musical contexts, states represent discretely valued attributes, such as pitch, duration, instrument, section, etc. The Markov assumption assumes that, given the current state, the next state is independent of previous states. This can easily be generalized so that the next state depends on a fixed number of past states; a first-order Markov chain depends only on the current state, a second-order on the current and immediately preceding state, and so on. If sequences are directly observable, then most inference problems can be solved by counting transitions. An alternative formulation is the n-gram model in which all possible symbols of length n are constructed from the training sequences, and their frequency tabulated. It is easy to see that the transition probabilities for an nth-order Markov chain can be computed by forming all $n + 1$-grams.

A significant problem that arises with fixed-order models is that, as the order n increases, the number of total n-grams increases as v^n, where v is the number of symbols. In music applications, such as melody prediction, where the past ten events could easily influence the next event, and where there might be a dozen or more symbols, we are left attempting to assess the relative frequency of greater than 12^{10} n-grams. Even for large databases, most n-grams will be unseen, leading to the so-called zero frequency problem [10]. This sparsity problem leads to a

fundamental tradeoff between using the predictive power of longer context and the increasing unreliability of higher order n-gram counts. Variable-length n-gram models attempt to overcome this problem in two ways: 1) by building many fixed-order models and integrating information across orders (smoothing), 2) and by reserving a certain amount of probability mass for unseen n-grams (escape probabilities). We describe these techniques in Section 4.2.

Variable length n-gram modeling is an ensemble method in which the predictions of many fixed-order models are integrated. Ensemble methods such as boosting have been shown to be effective for classification tasks [16]. Multiple viewpoint systems, introduced by Conklin [5], and developed by others such as Witten [5] and Pearce [15] can be thought of further generalizing the idea of integrating an ensemble of predictive models. The extension is based on the fact that music can be simultaneously represented in many ways. For example, a melody can be thought of in terms of chromatic pitches, intervals, scale degrees, or contour. A rhythmic pattern can be thought of in terms of onset times, durations or position-in-bar. If, for example, we are trying to predict the next note in a melody, having multiple representations is useful in capturing structure that is obvious given one representation, but less so in another. For example, a scale-degree representation of a melody might make it obvious that the chromatic pitch, say B, is actually the leading tone, making it very likely that the next note is C. However, if the training database contains many melodies in many different keys, this might not be obvious from the chromatic pitch representation. We describe the multiple viewpoints framework in Section 4.3.

Little work to date has been done on statistical modeling of tabla. Gillet [7] and Chordia [4] both used an HMM framework for tabla transcription, while Bel and Kippen [9] created a model of tabla improvisation based on a context-free grammar, one of the earliest computational tabla models.

Tabla is the most widely used percussion instrument in Indian music, both as an accompanying and solo instrument. Its two component drums are played with the fingers and hands and produce a wide variety of timbres, each of which has been named. A sophisticated repertoire of compositions and theme-based improvisations has developed over hundreds of years. Although tabla is primarily learned as part of an oral tradition, it is also notated using a system that indicates strokes and their durations. Unfortunately, the correspondence between strokes and names is not one-to-one. Depending on the context and the stylistic school, the same stroke will be given different names. And, in some cases, different strokes will be given the same name. This is unproblematic in the context of an oral tradition but requires that care be taken when interpreting symbolic notations.

3. TABLA DATABASE

The database used for training the model is a set of traditional tabla compositions compiled by tabla maestro Alok Dutta [6]. The compositions were encoded in a Humdrum-based syntax called **bol that encoded the stroke name and duration [4]. The database which is available online consists of 35 compositions in a variety of forms. Altogether there are 27,189 strokes in the dataset, composed of 42 unique symbols.

4. N-GRAM MODELING

N-gram modeling is a commonly used technique to probabilistically model sequences of elements such as phonemes in speech, letters in a word or musical notes in a phrase. [13] N-grams can be efficiently stored in a tree-shaped data structure, commonly referred to as a trie or prefix tree. Figure 1 is the trie for the sequence ABAB+C. In such a trie, branches represent the succession of certain symbols after others, and a node at a certain level of the trie holds a symbol from the sequence, along with information about the symbol such as the number of times it was seen in the sequence following the symbols above it, and the corresponding probability of occurrence. In Figure 1, the subscript below a symbol represents the symbols probability given the context, defined by the path through the trie to that node, while the superscript above it represents the count value. Thus, in the topmost level, the probabilities represent the priors for the symbols. During construction of the trie, symbols are fed sequentially into the system one-by-one. For the above example, after the sequence ABAB, the trie looks like Trie1 in figure Figure 1. When a new symbol 'C' follows, corresponding nodes are created at all levels of the trie: 5-gram node using 'ABABC', 4-gram node using 'BABC', trigram node using 'ABC', bigram node using 'BC' and a 1-gram/prior entry for 'C' at the topmost level. The corresponding probabilities are also updated resulting in Trie 2 in Figure 1.

After the trie has been built in this manner, it can be used to predict the next symbol given a test sequence. This is done by following the nodes of the trie downwards from its top, in order of the symbols in the test sequence until the last symbol in the sequence (and the corresponding node in the trie) is reached. At that point, the probabilities associated with the children nodes represent the predicitve distribution over the symbol set, given the observed context. To allow for new symbols that may appear in the test sequence and to subsequently allow for a better matching of test sequences with missing or extra symbols compared to training sequences, we incorporate the concept of escape probabilities into our trie structure, as described in [17]. The above example trie would then look like Trie3 in figure Figure 1. We describe the use of escape probabilities in section 4.2. For long training sequences, the depth of the trie can become large and is often restricted to a maximum order to limit memory usage and to speed prediction given a test sequence.

The modeling and evaluation framework was implemented in C++ as an external object in Max/MSP along with supporting patches.
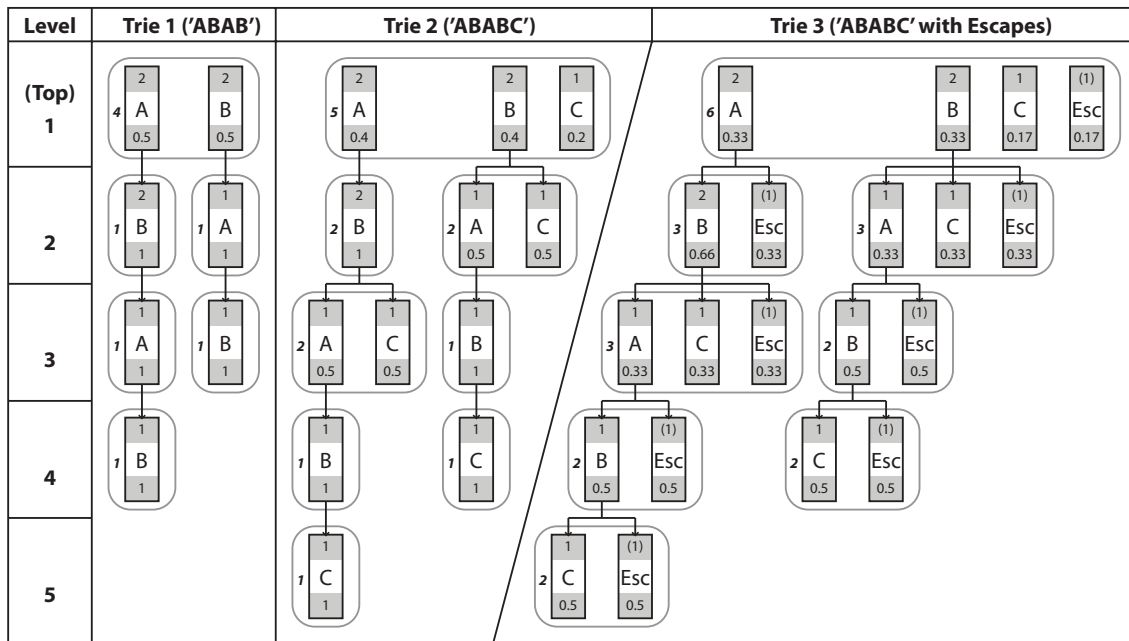
Figure 1. Illustration of tries built for the sequence 'ABAB' followed by the symbol 'C'. Superscripts represent count values, and subscripts represent probability values. Rounded boxes represent siblings, while italicized number at the left of a rounded box represents the total count among the siblings, which is used to calculate the 'probability' values. Trie 3 includes escape probabilities.

4.1 Escape Probabilities

As noted above, the zero frequency problem occurs because, in high-order models, most n-grams will never have been observed [10]. Using a simple counting scheme, the model would assume zero probability for these unseen events, thereby returning infinite entropy should they occur in the test sequence. The solution is to reserve a small amount of probability mass for events that haven't occurred yet. This is done by reserving an escape proabability for each level of the trie. Whenever an event returns zero probability, it returns the escape probability instead. There are many ways to assign the escape probability. Based on the results of Bell and Witten [17], we have implemented the Poisson distribution method. The escape probability for each level is assigned by $e(n) = \frac{T_1(n)}{N(n)}$, where T_1 is the number of tokens that have occurred exactly once and N is the total number of tokens seen by the model so far.

4.2 n-gram Smoothing

Smoothing addresses the tradeoff between the specificity of higher-order models (if a match can be found) and the reliability of the n-gram counts for lower-order models. Since higher order models are much sparser, many n-grams will be assigned zero probability, and counts for n-grams that have been observed will tend to vary greatly based on the particular training database. This variance can be reduced by incorporating information from lower order models. There are two basic types of smoothing algorithms: backoff models and interpolation models. Given a test sequence, a backoff model will search for the entire sequence, and if no match is found in the trie, the process

continues recursively after dropping the first element of the sequence. The process stops once a positive match is found and the count for that n-gram count is greater then some threshold. Interpolated smoothing, by contrast, always incoporates lower order information even if the n-gram count in question is non-zero.

For this study, two smoothing methods were primarily used, Kneser-Ney (KN) and an averaging method we term $1/N$. These were also compared to a simple backoff procedure. KN was adopted because earlier work showed it to be a superior smoothing method in the context of natural language processing [3]. The basic idea of KN is to ensure that lower order distributions are only used when there are few, or no, counts in the higher order models. When incorporating lower information, the probability is related not to the true count of the n-grams but rather is proportional to the number of different n-grams that it follows. An example in music might be as follows: given a bigram consisting of two rhythmic durations, where the second duration is transitional and not typically used on its own, we would not assign a high unigram probability since it is only used in association with the first duration. Implementation details can be found in [3].

Given a model, with M as the maximum order, the weights for each model are given by $w(n) = \frac{1}{m(maxOrder-n)}$. In other words, the higher orders receive greater weight than the lower orders. It is worth noting what happens in the case where a higher-order model has not seen a particular n-gram. In that case, even though the weight for that model will be relatively higher than for a lower order model, the probability of the n-gram, which will be determined by the escape probability, will be very small, and

typically much smaller than the weight.

4.3 Multiple Viewpoints

Our focus in the work so far has been to implement a multiple viewpoints system for music analysis, apply these principles to traditional North Indian tabla compositions, and identify the set of parameters that work best on this kind of music. Though we will touch upon the basics of the multiple viewpoint system, a more detailed explanation can be found here [5].

Conventional context-dependent predictive models track only the most basic aspects of music, like pitch, rhythm and onset times. Moreover, these variables are tracked together, so that finding an exact match for every possible context is practically impossible. A multiple viewpoints system, however, tracks each variable independently, maintaining many predictive models simultaneously. The final prediction is obtained by combining all these predictions into a meaningful set of basic parameters (such as pitch and duration). Such a system not only incorporates information from different variables but can also model complex relationships between two or more of these variables and make use of that information to strengthen its prediction. Furthermore, a multiple viewpoint system can make much better predictions in rare event cases, because of its ability to find context matches in at least one of its many models.

A viewpoint is nothing more than a set of events of a particular type. For example, a set of all pitch classes (C, C#, D, D# and so on until B) would consititue a viewpoint for a melody. Similarly, a viewpoint for rhythm would consist of the set of all onset times within a measure. These two viewpoints, pitch and rhythm, can be directly extracted from the music, are independent of each other and are called *basic types*. *Cross types* are formed when two or more basic types are combined and tracked simultaneously (T1 x T2). A cross type formed using Notes and Onset Times would consist of all elements in the Notes viewpoint in combination with all elements in the Onset Times viewpoint. Each element of this viewpoint is represented as a tuple {Note, OnsetTime}, instead of a single value. The number of all possible elements in a cross type is equal to the product of the number of elements in each basic type. A *derived type* depends on information extracted from a basic type. A simple example of this is melodic intervals, which are extracted from pitches. Derived types can use information from more than one viewpoint, and this can lead to the formation of cross types derived from derived types. Selection of appropriate representations is domain dependent and often uses prior knowledge of the music.

Here we use two basic types – strokes and durations. We also look at the following cross types: 1) Strokes x Durations and 2) Strokes x PositionInBar (PIB), where PIB refers to the onset position of a stroke as a fraction of the bar. Finally we introduce three derived types into the model. These were constructed by mapping the stroke names to a reduced set. Reduced Set 1 was made by eliminating different names for the same stroke, reducing the number of symbols from 41 to 32. Reduced Set 2 extended this idea by mapping acoustically similar strokes to the same name, which further reduced the number of symbols to 10. The open/closed mapping was made by classifying each stroke as resonant or non-resonant.

4.4 Merging Model Predictions

An important point here is the actual process of merging the predictions of each of the models. Though there are many different ways to do this, we use a weighted average as described in [15]. Each viewpoint model is assigned a weight depending on its cross-entropy at each time step. The weight for each model is given by $w_{\mathrm{m}} = H(p_{\mathrm{m}})/H_{\max}(p_{\mathrm{m}})$, where $H(p_{\mathrm{m}})$ is the entropy of the probability distribution and $H_{\max}(p_{\mathrm{m}})$ is the maximum entropy for a prediction in the distribution. Higher entropy values result in lower weights. In this way, models that are uncertain (i.e., have higher entropy) make a lesser contribution to the final distribution. The distributions are then combined by taking their weighted average.

4.5 Long Term and Short Term Models

A common limitation of such predictive models built on large databases is that the model is usually unaware of any patterns specific to a particular song. The model becomes too general to be effective, and very often patterns and predictions which seem obvious to humans are missed because they are infrequent in the global training database. To solve this problem, we used two models: a long-term model (LTM) built on the entire training database, and a short-term model that starts out empty and is built up as a particular composition is processed. In this work, the LTM is not updated as the test composition is processed.

When a composition is read, both models return a distribution over the symbol set at each time step. The predictions are merged into a final prediction using a weighted average as described above. Whenever the STM is uncertain, such as the beginning of a composition or new section, the system gives more weight to the LTM. In other sections, such as the end of a song, where the STM is more certain, the weighting scheme assigns more weight to the STM. A comparison of the cross-entropy measure for each model is presented in Table 1.

5. EVALUATION

Cross-validation was performed using a leave-one-out design. For each of the 35 compositions, training of the LTM was performed on the remaining 34. Reported results were averaged over all 35 trials. A common domain-independent approach for evaluating the quality of the models' predictions is cross-entropy [11]. If the true distribution is unknown, the cross entropy can be approximated by $-\frac{1}{n}\sum_{i=1}^{n}\log_2(p_i)$, which is the mean of the entropy values for a given set of predictions. To illustrate, at a given step t, we note the true symbol. We then look at the predictive distribution for symbols at step $t-1$ and calculate the entropy for the true symbol at step t. After running through

	Durations				Strokes				Stroke-Duration			
Order	Priors	Comb.	STM	LTM	Priors	Comb.	STM	LTM	Priors	Comb.	STM	LTM
1	1.891	1.049	0.916	1.890	3.614	3.184	2.958	3.613	4.965	3.883	3.397	4.962
5	1.891	0.563	0.510	0.897	3.614	1.117	0.994	1.780	4.965	1.294	1.162	2.354
10	1.891	0.469	0.429	0.814	3.614	0.868	0.814	1.609	4.965	1.060	0.995	2.220
20	1.891	0.383	0.356	0.736	3.614	0.805	0.780	1.584	4.965	1.007	0.966	2.201

Table 1. Summary of cross-entropy results for LTM, STM, and combined models for order 1-20
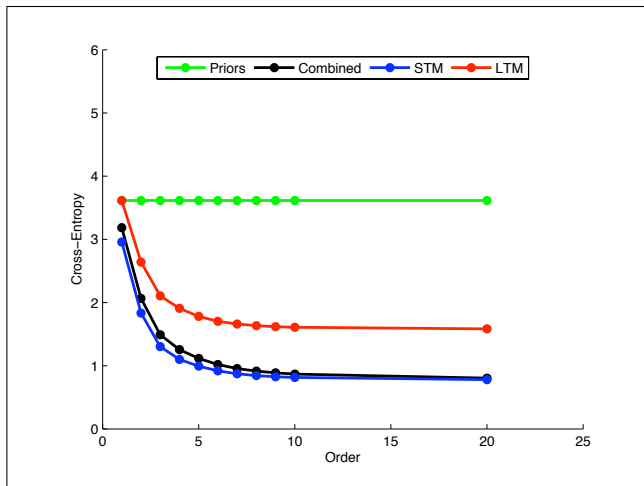

Figure 2. Cross-entropy for stroke prediction using LTM, STM, and combined models for orders 1-20

all the symbols in the test set, these entropies are averaged, giving a cross-entropy result for that particular test set.

6. RESULTS

Figure 2 shows cross-entropy as a function of model order using $1/N$ smoothing. The cross-entropy for the order 20 stroke LTM is 1.584, which is a surprisingly good result given the number of symbols (42). Compared with using a predictive distribution based on the prior probability of each stroke, cross-entropy was reduced by 2.030 (from 3.614 to 1.584). The STM entropy rate for strokes was a remarkable 0.780. Combining the LTM and STM based on entropy, as described in section 4.3, did not improve performance over the STM alone. The STM also outperformed the LTM when predicting durations (0.365 vs. 0.736) and when jointly predicting the stroke and duration (0.966 vs. 2.201). In both cases, the combined model offered no overall performance increase. Not surprisingly, in some cases the LTM outperformed the STM at the beginning of the composition, before the STM model had seen much data.

As expected, cross-entropy decreases monotonically as a function of model order. The curve decays roughly exponentially, with performance improving dramatically from order 1 to order 5, significantly between 5 and 10, and little between 10 and 20. This suggests that, for these compositions, memorizing more than the past 10 strokes does
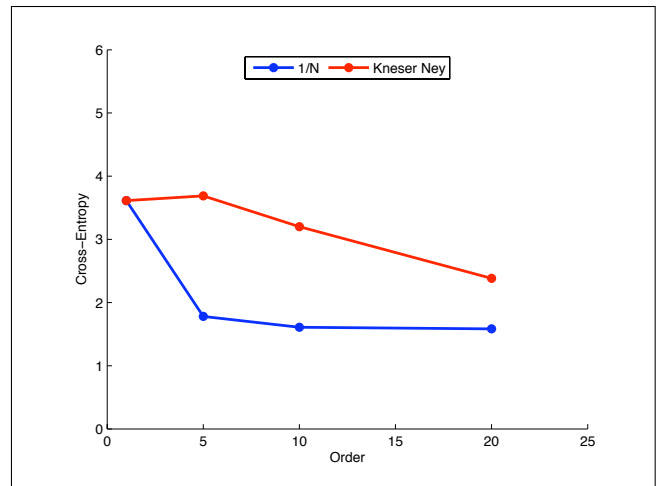

Figure 3. Comparison of Kneser-Ney and 1/N smoothing

Model	Durations	Strokes
Basic	0.814	1.609
Basic + SD	0.756	1.557
Basic + SD entropy	0.744	1.546
SD + PIB entropy	0.744	1.522

Table 2. Cross-types LTM where SD is stroke X duration, and PIB is position-in-bar. Merging of models was done using a weighted average with entropy-based weights, except for Basic + SD, which took the simple mean

little to improve predictions. This is true for the prediction of durations, strokes, and joint prediction of the stroke and duration.

Figure 3 shows the effect of different smoothing techniques on performance for the LTM. Both $1/N$ and Kneser-Ney smoothing significantly outperform a simple backoff method. $1/N$ is the clear winner for strokes, durations, and joint prediction. The difference in the quality of prediction decreases as the model size increases but is large throughout. Unusually, Kneser-Ney smoothing decreases slightly in performance as the model order is increased from 1 to 5.

Table 2 shows that cross-types had a small impact on performance with the addition of the stroke X duration type having the most impact.

In Table 3, we show the results for several LTM using derived stroke types, essentially more abstract sound categories based on the stroke name. For the LTM, Reduced

Derived Types	Strokes
Strokes only	1.60858
Strokes + Reduced Set 1	1.83609
Strokes + Reduced Set 2	1.81989
Strokes + Open/Close Set	1.61803

Table 3. Cross-entropy for stroke prediction using derived-types with LTM

Set 1 and 2 decreased performance by approximately 0.2, whereas open/closed marginally improved performance.

7. DISCUSSION

These results suggest that tabla compositions can be effectively encoded using a variable length n-gram model. Given a set of 42 stroke symbols, the best model's cross-entropy was 0.780, essentially meaning that it was on average uncertain between 2 strokes, a dramatic reduction from the 42 strokes in the vocabulary, as well as from the prior distribution which corresponded to approximately 12 strokes. Interestingly, the results suggest that tabla compositions exhibit strong local patterns that can be effectively captured using a STM, providing significantly better performance when compared with the LTM alone. Because many tabla compositions consist of a theme and variations, this result is not surprising. These data also suggest that it is almost always better to only use the STM, except for the very initial portion of the composition. Cross types seem to lead to small improvements, whereas derived types lead to small decreases. More experiments are needed in order to determine whether these changes are significant.

Another important result is that smoothing can have a large impact on predictive performance and seems to be highly domain dependent, with $1/N$ outperforming KN, a technique that had been shown to be amongst the best in another area. It is likely that the correct balancing of model order will depend on the database size and of course the distribution of n-grams. It would be interesting if further work could elucidate a clear theoretical basis for choosing a given smoothing method. In the absence of this, it is likely that performance could be improved by using a validation set and by adjusting how quickly weights fall off for interpolated smoothing techniques as the model order decreases.

8. FUTURE WORK

As always, we plan to continue to encode more tabla compositions to see if these results generalize. Additionally, we hope to test other merging methods such as geometric combinaton, a technique shown to be superior to additive combination in the context of melodies [11], as well as implementing cross and derived types for the STM. We also hope to use our trained models to generate novel tabla compositions and to use human evaluators to judge their quality. Lastly, we hope to use these results in an interactive tabla system that can anticipate and respond to a tabla improvisation.

9. REFERENCES

[1] C Ames. *The Markov Process as a Compositional Model: A Survey and Tutorial*. 1989.

[2] Jean-Julien Aucouturier, Franois Pachet, and M. Sandler. The way it sounds: timbre models for analysis and retrieval of music signals. 2005.

[3] Stanley Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *PROCEEDINGS OF THE 34TH ANNUAL MEETING OF THE ACL*, pages 310–318, 1996.

[4] Parag Chordia. *Automatic Transcription of Solo Tabla Music*. PhD thesis, Stanford University, December 2005.

[5] Darrell Conklin and Ian H. Witten. Multiple viewpoint systems for music prediction. 1995.

[6] Alok E Dutta. *Tabla: Lessons and Practice*.

[7] Olivier Gillet and Gael Richard. Supervised and unsupervised sequence modeling for drum transcription. In *Proceedings of International Conference on Music Information Retrieval*, 2007.

[8] David Huron. *Sweet Anticipation: Music and the Psychology of Expectation*. MIT Press, 2006.

[9] Bel B Kippen J. *Bol Processor Grammars In Understanding Music with AI*. AAAI Press, 1992.

[10] W J Teahan John G Cleary. Experiments on the zero frequency problem. 1995.

[11] Marcus Pearce Johnston. *The construction and evaluation of statistical models of melodic structure in music perception and cognition*. PhD thesis, City University, London, 2005.

[12] Kyogu Lee. Automatic chord recognition from audio using an hmm with supervised learning. In *In Proc. ISMIR*, 2006.

[13] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 2002.

[14] Pearce, Herrojo Ruiz, Kapasi, Wiggins, and Bhattacharya. Unsupervised statistical learning underpins computational, behavioural and neural manifestations of musical expectation. 2010.

[15] Marcus Pearce, Darrell Conklin, and Geraint Wiggins. Methods for combining statistical models of music. 2004.

[16] Dietterich T.G. Ensemble methods in machine learning. 2000.

[17] Ian H. Witten and Timothy C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. 1991.