Predictive Tabla Modeling Using Variable-length Markov and Hidden Markov Models

Parag Chordia, Avinash Sastry, Sertan Şentürk

Georgia Institute of Technology, Georgia Tech Center for Music Technology

`ppc@gatech.edu, asastry@gatech.edu, sertansenturk@gatech.edu`

Abstract

Tabla is a sophisticated, centuries-old percussion tradition from North India based on timbral sequences. We model these sequences in a predictive framework with variable-length Markov models (VLMMs). Using a database containing nearly 30,000 strokes in 35 compositions, we show that VLMMs have high predictive accuracy, with an average perplexity of 1.80, and median perplexity of 1.19, on a task with 42 distinct symbols. This basic framework is extended by the introduction of several new smoothing techniques that determine how to integrate predictions from the different order models. The model is then extended to include parallel representations of the sequence, a technique known as Multiple Viewpoint modeling.

The work is then extended to the problem of recognizing strokes from audio. In this hidden context, the identity of the previous stroke is not revealed at each time step. A variable-length Hidden Markov model (VLHMM) is used to determine the next-symbol distribution that is used in computing the perplexity. We detail how the forward probabilities can be efficiently computed for the VLHMM using by traversing a prediction suffix tree (PST) that is used to represent sequences. Using a VLHMM with a maximum order of 3, we obtain an average perplexity of 2.31, with a median of 1.16 on a 9 target task. To the best of our knowledge, this is the first use of variable-length Hidden Markov models for music modeling or prediction.

1 Introduction

Anticipation is an essential component of listening and performing. Regularities in the structure of music lead to expectations that focus attention and guide perception [30]. For musicians, anticipation is essential for synchronization. Markov and n-gram models have been used extensively to model temporal structure in music [1]; they have been successfully applied to algorithmic composition, timbral analysis [5] [29], and music cognition [36].

In these experiments, we are concerned with discrete, mid-level predictions, what might be colloquially called "note-level". This contrasts with low-level modeling of audio frames, which is commonly done for music information retrieval tasks such as genre-recognition or music similarity [43]. To date, most note-level modeling has focused on symbolic music, due to the difficulty of transcribing music from audio. Here we begin with a symbolic prediction task and then generalize to the more difficult task of prediction from audio. Our motivation for this research is to develop systems that can interact naturally with human performers, even in improvised settings. This requires the system to anticipate what will come next. Predictive accuracy is one significant measure of model quality. By developing more highly predictive models we can develop computational models of music theory that help to describe and elucidate musical traditions.

In our first experiment, we attempt to predict how symbolic tabla compositions, which essentially consist of sequences of named strokes, will continue. Our basic task is to predict the next stroke (i.e. to compute the next-symbol distribution), given a context that represents the previous sequence of strokes. A fundamental observation about musical patterns is that they are often of wildly different lengths — this happens both for aesthetic reasons, and because patterns are frequently combined hierarchically to form longer patterns. This is particularly true in solo tabla music, where musically significant patterns can range in length from a couple to hundreds of strokes.

We show that variable-length Markov models (VLMMs) are an effective modeling technique in these situations, with high predictive accuracy. We introduce new smoothing techniques for integrating the fixed-order Markov models that comprise the VLMM, based on a family of exponential curves. Further, building upon previous research [14], we show that performance can be improved by using an ensemble of VLMMs, where each component is used to represent different aspects of the surface structure (described in Section 2.4), a technique called Multiple Viewpoint (MV) modeling.

The second experiment attempts a similar prediction task, but beginning with an audio recording. Rather than a discrete sequence of symbols representing stroke names, after the audio is segmented we have a sequence of feature vectors computed from the audio. Given a context, which in this case consists of the past several feature vectors, our task is to compute the next-stroke distribution. In the symbolic prediction task, as time advances, the true identity of the stroke is revealed. By contrast, when we are working from audio, the label remains hidden, making it much harder to compute the next-stroke distribution. Hidden Markov Models are extensively used to model such situations, with hidden states representing the information we would like to know (i.e. strokes), and observations modeled as samples drawn from probability distributions that depend on the hidden states. Our main contribution is to extend the VLMM to the hidden domain. We show that such variable-length Hidden Markov models (VLHMMs) outperform low-order HMMs as well as fixed, high-order HMMs. High-order VLHMMs are better able to utilize long patterns to disambiguate observations and predict continuations. Integrating low- and high-order models, as variable-length models do, further improves prediction by allowing low-order information, which is less specific but more reliable, to be incorporated as well.

2 Background and Related Work

2.1 Tabla Solo

Tabla is the most widely used percussion instrument in Indian music, both as an accompanying and solo instrument. It is used extensively in classical, folk, film, popular, and devotional music, and its distinctive timbre forms one of the ubiquitous signifying elements of the music of North India. Unlike Western classical music, Indian classical music makes extensive use of percussion.

Its two component drums are played with the fingers and hands and produce a wide variety of timbres, each of which has been named (Table 2.1). A sophisticated repertoire of compositions and theme-based improvisations has developed over hundreds of years. Tabla is a natural candidate for Markov modeling because it consists, at a basic level, of a sequence of discrete states that are temporally structured. Tabla is particularly interesting because of the complex patterns and dependencies that are present in typical compositions. Although tabla is primarily learned as part of an oral tradition, it is also notated using a system that indicates strokes and their durations.

The repertoire of tabla can be divided into two basic categories, structured improvisations, and through-composed material. In the former category are *qaidas*, a theme and variation form, and *peshkar* another type of thematic improvisation, and *relas*. *Relas* are very dense, fast drum-roll-type textures, that are played in a smooth, flowing manner. In the latter category are *gats*, *tukdas*, and *chakradhars*, various types of fixed compositions.

Little work to date has been done on statistical modeling of tabla. Gillet [29] and Chordia [12] focused on tabla transcription using a Bayesian classifier and HMMs. Bel and Kippen [7] created a symbolic model of tabla improvisation based on a context-free grammar, one of the earliest computational tabla models. The current work builds on the generative tabla system of Rae and Chordia [41]. Compared with the authors previous work, the novelty of this work lies in 1) application of a Multiple Viewpoint framework 2) extension of the variable-length Markov framework to the hidden domain for use with audio 3) use of a prediction (cross-entropy) rather than classification (recall / precision) for evaluation.

Table 1: Tabla strokes used in the current work. The drum used is indicated, along with basic timbral information. “Ringing” strokes are resonant and pitched; “modulated pitch” means that the pitch of the stroke is altered by palm pressure on the drum; “closed” strokes are short, sharp, and unpitched.

Stroke name	drum used	timbre
dha	compound	ringing *bayan*
dhe	compound	ringing *bayan*
dhec	*dayan*	closed
dhen	*dayan*	ringing *bayan*
dhin	compound	ringing *bayan* and *dayan*
dun	compound	ringing *bayan* and *dayan*
ge	*bayan*	ringing *bayan*
geM	*bayan*	ringing *bayan*, modulated pitch
ke	*bayan*	closed
na	*dayan*	ringing *dayan*
nec	*dayan*	closed
nen	*dayan*	ringing *dayan*
rec	*dayan*	closed
te	*dayan*	closed
tin	*dayan*	ringing *dayan*
tun	*dayan*	ringin *dayan*
tunke	compound	closed *bayan*, ringing *dayan*

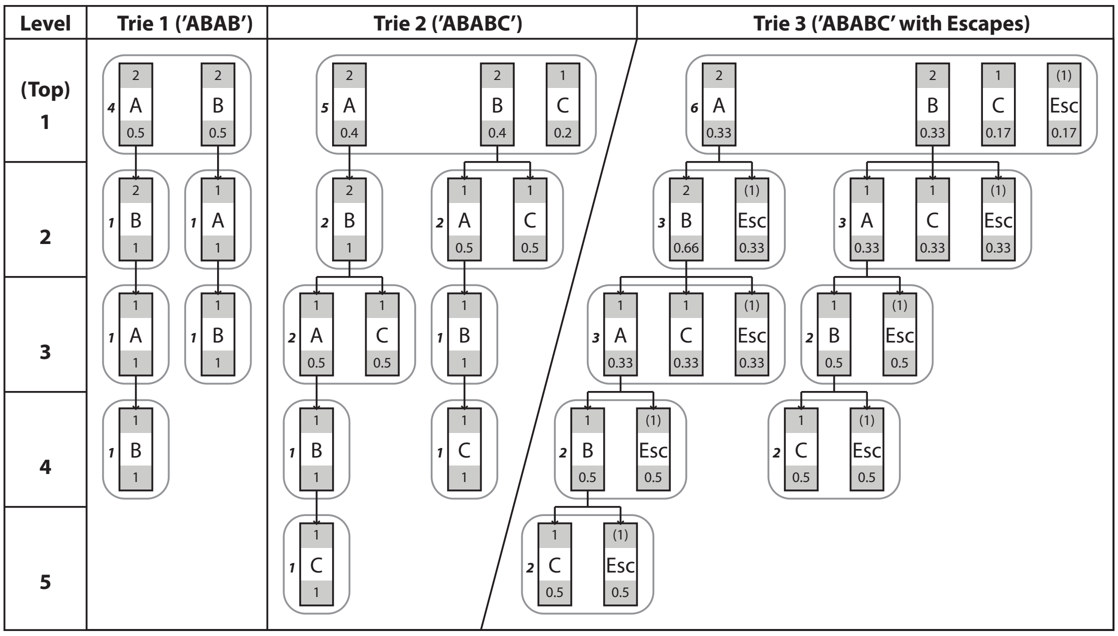


Figure 1: Illustration of tries built for the sequence {ABAB} followed by the symbol {C}. Superscripts represent count values, and subscripts represent probability values. Rounded boxes represent siblings, while italicized number at the left of a rounded box represents the total count among the siblings, which is used to calculate the 'probability' values. Trie 3 includes escape probabilities.

2.2 Visible Markov Models (VMMs)

Markov models are arguably the most successful and widely used methods for modeling the temporal structure of sequences with discrete state spaces. They are the basis for state-of-the-art algorithms for lossless compression [8], speech recognition [31], language modeling [10], biological sequence analysis [28], and financial models [9]. In music, VMMs and HMMs have been used for algorithmic composition [1], timbral analysis [5, 29], structure analysis [33], and music cognition [30].

The basic prediction problem can be stated as follows: given a sequence of discretely valued observations, $\{x_1, \ldots, x_{t-1}\}$, compute the next-symbol distribution $P(x_t|x_1, \ldots, x_{t-1})$. In music, $\{x_1, \ldots, x_{t-1}\}$ might be a series of piano notes constituting a melody, a series of drum-hits in a rhythmic pattern, or a series of chords in a song. Given what has been heard, we would like to be able to predict the next event – both what will happen, and when it will happen. A musical event can be described by various attributes, such as pitch, duration, timbre, and loudness, that we might wish to predict.

Markov models are generative, probabilistic models based on a succession of discrete states. A state is represented by one of an alphabet of symbols ($x_i \in S$). Sequences of symbols are often referred to as strings, reflecting their use in language models. Markov models assume that, given the current state, the next state is independent of previous states: $P(x_t|x_1, \ldots, x_{t-1}) = P(x_t|x_{t-1})$. This can easily be generalized so that the next state depends on a fixed number, n, of past states: $P(x_t|x_1, \ldots, x_{t-1}) = P(x_t|x_{t-n} \ldots x_{t-1})$. If we have sequences generated by such a model, then this conditional probability can be calculated by counting how often the symbol x_t follows the context. Strings of length n are often referred to as

n-grams.

If we want to model long-term dependencies, we must increase the order n. However, the total number of possible n-grams increases exponentially: 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 pitches, we are left attempting to assess the relative frequency of greater than 12^{10} (12 billion) n-grams. Even for large databases, most n-grams will never have been seen, leading to the so-called zero frequency problem [45, 15]. In a test sequence, when we encounter an n-gram that has not been previously seen, our fixed-order model assigns it a probability of zero. Due to this sparsity problem, there is a fundamental trade-off between the predictive power of longer contexts, which have longer memories, and the unreliability of higher order n-gram counts.

Variable-length Markov models (VLMMs) address this problem by using an ensemble of fixed-order models, up to order n, to smooth probability estimates. The basic idea is to use the high-order count, if it can be found, and to otherwise recursively back off to lower-order models until a match for the sequence is found [45]. A variation, called interpolation smoothing, does not stop when a match is made, but computes a weighted average of probability estimates across all orders [11]. Both these techniques utilize escape probabilities that reserve a certain amount of probability mass for unseen n-grams [11]. Smoothing addresses the trade-off 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. Smoothing can have a substantial impact on predictive performance, and a variety of smoothing techniques, such as Laplace, Katz backoff, Good-Turing, Kneser-Ney, 1/N, and parametric smoothing have been previously introduced [34, 11, 14, 13].

Rather than naively storing counts for all n-grams in a table, to avoid space complexity that increases exponentially with model order, and to make it easy to search for a sequence, n-grams and counts are stored in a partial k-ary tree called a prediction suffix tree (PST) [44]. Figure 1 is the PST for the sequence {ABAB}+{C}. In the PST, branches represent the succession of certain symbols after others, and a node at a certain level of the PST 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 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 PST, symbols are fed sequentially into the system one-by-one. For the above example, after the sequence {ABAB}, the PST looks like Trie1 in 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 1.

VLMMs form the basis for state-of-the-art music prediction and generation systems [26, 23, 2, 3, 4, 32, 25, 24, 21, 35]. These systems start with symbolic input, typically MIDI (musical instrument digital interface), and learn pitch and duration sequences using a VLMM. Some of these systems have tried to improve prediction by using abstractions of the basic data types by, for example, representing pitches

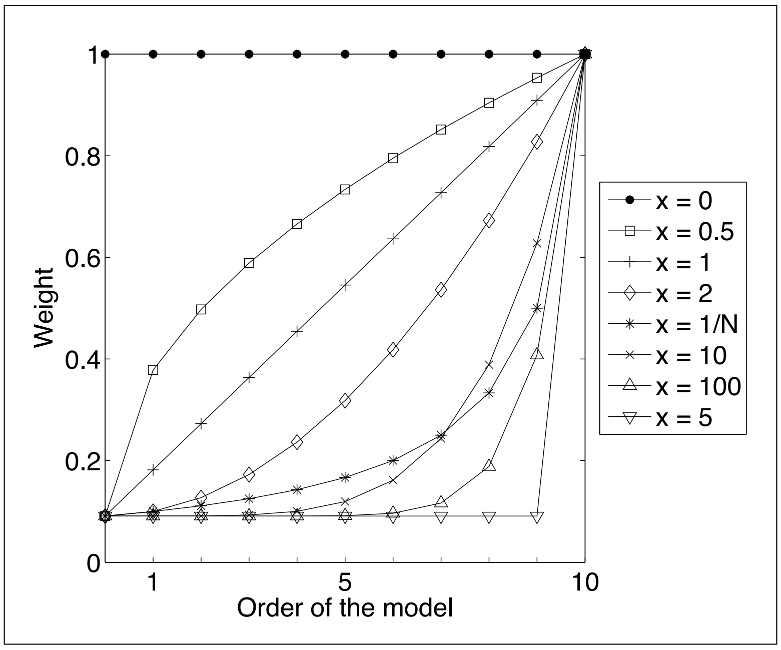


Figure 2: Family of exponential curves used for parametric smoothing model. Each curve represents a given exponent coefficient and shows relative weight assigned to each order while taking weighted average.

in terms of broader pitch-regions [35]. The practical motivation for this is to be able to find higher-order matches when a more specific representation would yield none.

2.3 Smoothing

Smoothing addresses the tradeoff between the specificity of higher-order models 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: back-off models and interpolation models. Given a test sequence, a back-off 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, stopping once a positive match is found and the count for that n-gram count is greater then some threshold. Interpolated smoothing, by contrast, always incorporates lower order information even if the n-gram count in question is non-zero.

In our previous work [14], two smoothing methods were studied, Kneser-Ney (KN) and an averaging method we termed $1/N$. KN was adopted directly from language processing because earlier work showed it to be a superior smoothing method in the context of natural language processing [11]. In the 1/N smoothing method, weights for the n-th order model are given by $\frac{1}{(maxOrder-n+1)}$, giving greater relative weight to higher-order models We found that the 1/N model outperformed KN.

We introduce a back-off model and a novel, parametric approach based on generalizing the 1/N technique. For the back-off model, the match threshold was set to two; if the pattern was observed only once at a given level, we continued to back-off. The motivation for setting the threshold greater than one was to ensure that, particularly for higher-order models, that a match was more likely to be a genuine pattern. The parametric model is based on a family of exponential curves given by

$$\begin{gathered} w(n) = a\left\{\left(1-\frac{c}{a}\right)\left(\frac{n}{maxOrder}\right)^{x}\right\}+c \\ n = 0,1,2,3...maxOrder \end{gathered} \tag{1}$$

As x increases, the curve rises more steeply, giving more importance to higher-order models as can be seen in Figure 2. The parameters a and c are used to set the minimum and maximum allowable weights.

2.4 Multiple Viewpoints

MVMs, introduced by Conklin [19, 16, 20, 18, 17, 46], and developed by others such as Pearce [38, 37], further generalize 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 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.

In a conventional context-dependent predictive model, correlations can only be modeled by tracking a new variable, which is formed by taking the Cartesian product of the domains of basic attributes like pitch, duration, and loudness [16]. This makes finding an exact match for a context practically impossible. By contrast, a multiple viewpoints system maintains an ensemble of predictive models of varying degrees of specificity. This allows it to match specific, complex patterns when the information is available in the trained model, and gracefully fall back on simpler, more abstract models when it is not. MVMs merge the predictions of their models using a weighted average, according to each model's uncertainty at a given time step. This is quantified using the entropy of the next-symbol distribution (Section

2.5) [20].

2.5 Merging Model Predictions

An important point here is the 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 [38]. 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.

3 Experiment 1: Symbolic Tabla Prediction

3.1 Viewpoints

As described in [20], a type is an abstract property of a musical event, such as start time, note duration or melodic contour. A simple type uses one musical property, such as the "Pitch Class" type described above. In contrast, a cross-type consists of two or more simple types considered together and represented as a tuple T1-T2. An example would be a cross-type modeling the relationship between pitch class and note duration. Another possibility is a derived type. A derived type is dependent on the relationship between one or more types, like melodic contour, which depends directly on the pitch class. The number and complexity of the viewpoints chosen depends on the nature of the sequence being modeled and the problem in question.

In addition to the basic viewpoints of strokes and durations, we also introduced a cross-type which modeled stroke-duration pairs, and another which modeled the strokes metric position in the rhythmic cycle. The stroke-duration viewpoint would allow as to distinguish patterns such as *dha-tete* and *dhatete.* In the first, the stroke *dha* is twice as long. The stroke-PositionInCycle viewpoint was based on the intuition that certain stroke tend to appear at strong positions in the metric cycle. For example *dha* is often used to resolve phrases and thus has a greater likelihood of appearing on the first beat, where most long phrases conclude. The following table summarizes these viewpoints.

1. Strokes: Numbers representing the unique tabla strokes
2. Durations: Interval between two strokes as a ratio of each bar
3. Stroke-Duration: a cross-type consisting of a stroke and its duration
4. Stroke-PIC: a cross-type relating a stroke to its metric position in a complete rhythmic cycle

3.2 Long Term and Short Term Models

One problem that we had to address was the frequent repetition of themes and phrases. Because a large part of music on the tabla is built upon a theme-variation structure, the theme for a specific song is repeated quite often within that song. In addition, some phrases and their improvisations make sense only in the right context - in such cases, a prediction based on a large generic database will almost

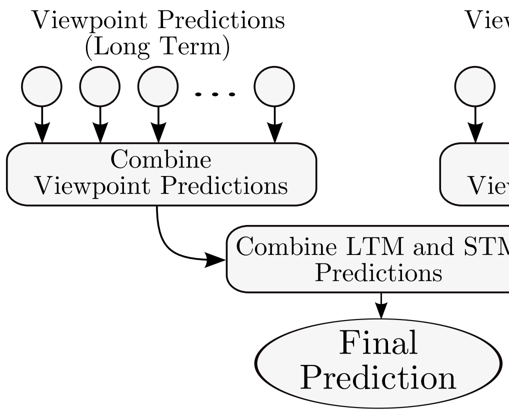

Viewpoint Predictions (Long Term)
Viewpoint Predictions (Short Term)
Combine Viewpoint Predictions
Combine Viewpoint Predictions
Combine LTM and STM Predictions
Final Prediction

Figure 3: Block diagram illustrating the how the models are combined in the VLMM to make the next-stroke prediction.

always sound out of place. A common limitation of a predictive model built on a large database is that the model is usually unaware of any patterns specific to a particular song. The model becomes too general to be effective; patterns and predictions which seem obvious to humans are missed because they are infrequent in the training database or because there are simply too many possibilities to choose from. We needed a system that would make very specific predictions in some parts of a song, and fall back on the general rules in other parts. 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 [20]. Each of our viewpoints has a long term model (LTM), which is fed with a database of tabla compositions, and a short term model (STM), which is trained only upon the current song being evaluated. When a decision needs to be made, predictions from both models are combined using a perplexity-based merging scheme (Figure 3). More weight is given to the model that has the lower perplexity.

3.3 Merging Viewpoints

When a prediction is to be made at a given time-step, the LTM and STM are combined into a single distribution for each of three PSTs (Strokes, Durations, and Stroke-Durations). Additionally, the Strokes-Durations cross-type is combined with the basic stroke and duration types by marginalizing the Strokes-Durations predictive distribution. The Strokes-PositionInCycle viewpoint is only used in the STM. The LTM cannot be used for this viewpoint because there are many different rhythmic cycles in the corpus leading to very different relations between strokes and metric position. At each time-step, the model checks the current position in the cycle (the length of the cycle is defined by the length of the theme for that song – if the theme is not complete yet, then this PST is not used) and calculates the probabilities for strokes at this position. The resultant distribution is then combined with Strokes distribution. The Strokes and Durations models are then normalized leading to a single predictive distribution for each that can be used to calculate the cross-entropy.

3.4 Tabla Database

The database used for training the model is a set of traditional tabla compositions compiled by tabla maestro Alok Dutta [27]. A Humdrum-based syntax called **bol [12] was developed to provide a notation for representing the compositions. Each **bol file encodes the name and duration of every stroke in the song along with metadata describing details about the song like its form, meter and tempo. The online database consists of 34 such compositions comprising a variety of forms: 13 Qaidas, 5 Relas, 4 Tukdas, Gats and chakradhars, 2 Keharva thekas, 1 Dadra and 1 Laggi. Altogether, there are 27,189 strokes in the dataset composed of 42 unique symbols. In addition to the symbolic database, an audio database of these compositions was also synthesized. This is described in more detail in Section 4.1.

3.5 Evaluation

A common domain-independent approach for evaluating the quality of model predictions is cross-entropy [34]. If the true distribution is unknown, the cross entropy can be approximated by $H = -\frac{1}{n}\sum_{i=1}^{n} \log_2(p_i)$, which is the mean of the entropy values for a given set of predictions and is expressed in bits. 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 all the symbols in the test set, these entropies are averaged, giving a cross-entropy result for that particular test set. A closely related concept, often used in natural language modeling is perplexity per symbol [34], defined to be $P = 2^H$, where H is the cross-entropy as described above. Perplexity has a simple interpretation, it is the number of choices that the model is confused between and would be equivalent to the model choosing uniformly between P choices.

Cross-validation was performed using a leave-one-out design at the song-level. For each of the 35 compositions, the LTM was trained on the remaining 34. The STM was trained on the remaining song. Reported results were averaged over all 35 trials. [41].

Model	**Strokes**		**Strokes MV**		**Durations**		**Durations MV**		**Stroke-Duration**		**Stroke-PIC**	
	Av.	Med.	Av.	Med.	Av.	Med.	Av.	Med.	Av.	Med.	Av.	Med.
BO	2.71	1.08	2.41	1.07	2.11	1.014	1.83	1.01	3.84	1.29	2.58	1.40
1/N	2.08	1.37	1.80	1.19	1.44	1.11	1.30	1.06	2.69	1.58	2.55	1.37
P (x=1)	2.00	1.33	2.03	1.27	1.55	1.09	1.42	1.07	2.60	1.43	2.57	1.37

Table 2: Average and median of perplexity results for for back-off, 1/N, and parametric (with exponent coefficient equal to 1) smoothing methods. Results for combined models using a maximum order of 10. MV refers to the multiple-viewpoints model in which the Stroke-Duration and Stroke-PIC viewpoints have been incorporated.

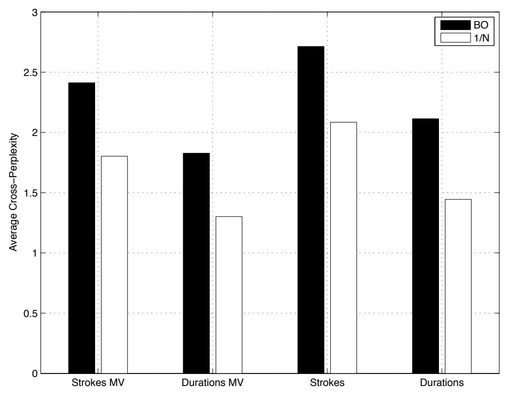
	Durations	**Strokes**	**Stroke-Duration**
Order 10	1.76	3.05	4.66

Table 3: Summary of perplexity results for LTM for order 10.

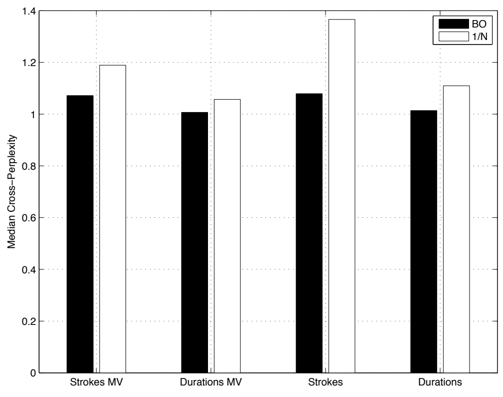
3.6 Results and Discussion

Table 2 summarizes the perplexity results for the different smoothing models and different prediction types. For all tasks the interpolation-based smoothing methods have lower average perplexity than the back-off method. Perplexity for stroke prediction using the multiple viewpoints (MV) is 2.03 for the parametric model (using an exponent of 1) and 2.41 for the back-off model, with similar differences for Durations and Stroke-Duration. This compares favorably to a baseline perplexity of 12.24 when using only using prior probabilities for strokes. When predicting Stroke-PIC there are no significant differences between smoothing methods.

Incorporating the Stroke-Duration and Stroke-PIC viewpoints improves performance when predicting strokes (2.41 vs 2.71) and durations (1.83 vs 2.11) for the back-off model. For the parametric model, these additional viewpoints do not seem to significantly improve performance. Table 3 shows the results when only the LTM is used with 1/N smoothing. It can be seen that incorporation of the



(a) Comparison of average perplexity between BO and 1/N for single and multiple viewpoints.



(b) Comparison of median perplexity between BO and 1/N for single and multiple viewpoints.

Figure 4: Comparison of perplexity between BO and 1/N models.

STM signficantly reduces perplexity for stroke prediction (3.05 to 1.37).

Median entropy is less than average entropy for all models (Table 2). For example, when predicting strokes using 1/N smoothing, average perplexity is 2.08 and median perplexity is 1.37. Interestingly, while average entropy is lower for interpolation models compared with BO, median entropy is lower for BO. This is true for both strokes and durations. As can be seen in Figure 6, the distribution of entropy values showed a larger peak near zero and fatter tail for BO vs. interpolation methods. This is perhaps due to the fact that back-off methods do well when a high-order match is made but do not degrade as well because lower-order information is not fully integrated. More specifically, if a high-order model does not find match at a given order the escape probability is used. If several high-order models are combined then the multiplication of escape probabilities can lead to very small probability values. If that stroke then occurs it will lead to a high entropy value. Outliers can clearly be seen in Figure 6, giving support to this idea. Thus it seems that the appropriate choice of smoothing algorithm depends

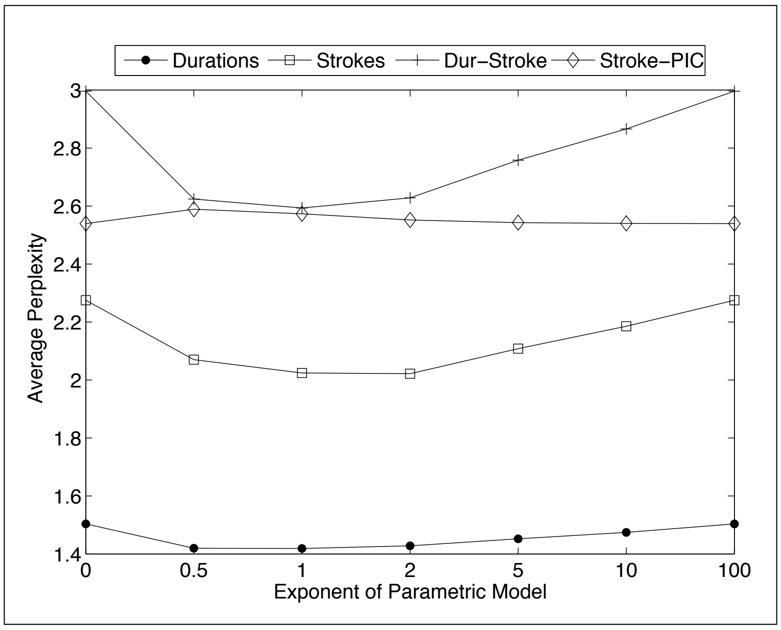


Figure 5: Perplexity as a function of exponent coefficient in parametric model. There seems to be an optimal range between 1 and 2.

on ones risk function. If we wish to minimize very bad misses (a sort of minimax strategy), then interpolation methods are preferable, whereas if we can tolerate such occasional bad misses BO will lead to lower median entropy.

Figure 5 shows results for different values of the exponent for the parametric model. It seems that there is an optimum range for the exponent between 0.5 and 2, with values outside of this range leading to worse performance. While these results are not statistically significant, they suggest that for interpolation methods the relative weights of models is important. More specifically, it seems that while higher-order models should receive greater weight, increasing their weight beyond a certain point decreases performance.

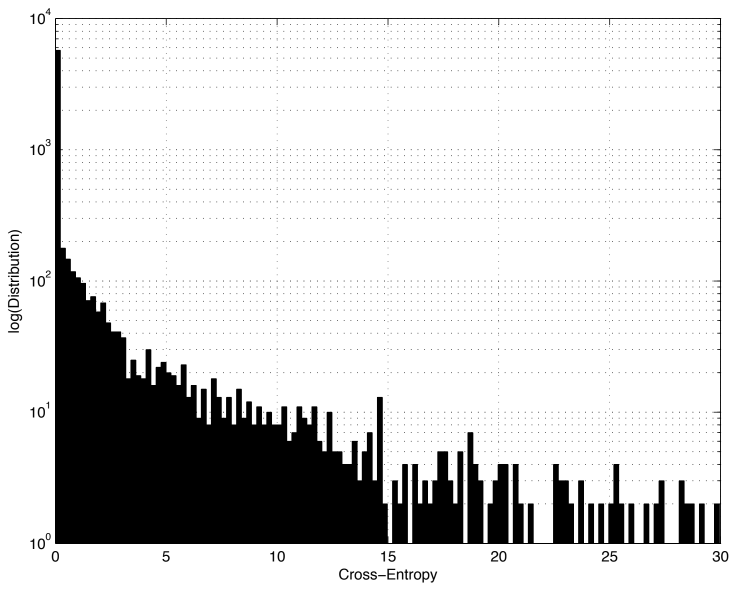


Figure 6: Distribution of perplexity values when predicting strokes for the Back-Off model

4 Experiment 2: Tabla Prediction from Audio

Taking inspiration from the ensemble methods used for symbolic tabla prediction, here we use an ensemble of HMMs of different orders to predict the next-stroke distribution, given a sequence of features vectors computed from a recording of a synthesized tabla performance.

4.1 Method

We created synthetic training and test audio databases based on the 35 symbolically encoded compositions, allowing us to use a much larger database (27,000 strokes) than has been previously been undertaken. The compositions were synthesized using a sample-based technique described in [40] and each was saved as a WAV file (44.1 kHz, 16 bit). First, each nominal stroke name was mapped to an acoustic category. In other words, different names for the same timbre were mapped to a single stroke type. The strokes used for synthesis are given in Table 2.1. For occurrence of each stroke type, the synthesis algorithm chose from among approximately 10 samples recorded by a professional tabla player, preserving some of the natural variability of a real performance. Each composition was also synthesized using a completely different set of samples that came from a different performer and drum to make the recognition task more difficult and realistic. The synthesized compositions can be heard at http://paragchordia.com/jnmrTabla. The time stamp of each stroke onset was also saved in a text file, and was later used for segmentation.

Next, the audio files and their corresponding time-stamps were read into MATLAB. The audio was segmented into individual strokes using the time-stamps. It remains for future work to automatically detect the onsets. From previous work we know that the main difficulty of detecting onsets is masking due to loud ringing strokes. Nevertheless, in non-archival recordings, recall and precision rates for tabla onsets are typically high. For each stroke, the first 21 MFCCs were computed and the 0th coefficient, representing signal energy, was discarded. Features were calculated on the entire stroke, without breaking it into smaller frames; the FFT length was equal to the size of the stroke, resulting in a high-resolution spectrum as nearly all strokes were longer than 4096 sample frames at a sampling rate of 44.1kHz.

The testing and training feature vectors for each composition, along with the parameters of the multivariate Gaussian (MVG) for each stroke type, were passed to the VLHMM, which is described next.

4.2 Variable-length Hidden Markov Model

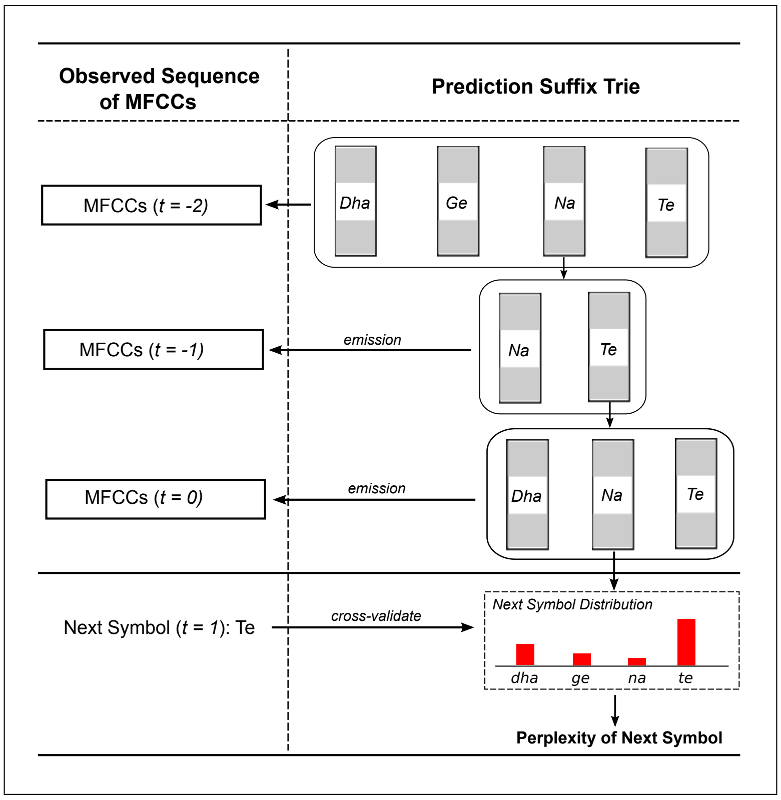


Figure 7: A portion of the Variable-Length Hidden Markov Model is shown unfolded for several time-steps

The models described in Section 2.2 can be used if symbols can be directly observed or easily derived from the representation. The situation is more complicated when dealing with audio. Information, such as when an event begins and what its pitch is, must be extracted using digital signal processing (DSP) techniques. There are two basic approaches: prior to modeling, DSP can be used to create symbol sequences by segmenting and labeling events, or a HMM that jointly considers the state sequence and observations can be applied.

Generative models, such as HMMs, are based on computing the joint probability $P(X_t, S_t)$, where $X_t = \{x_1, \ldots, x_t\}$ is a sequence of observations, and

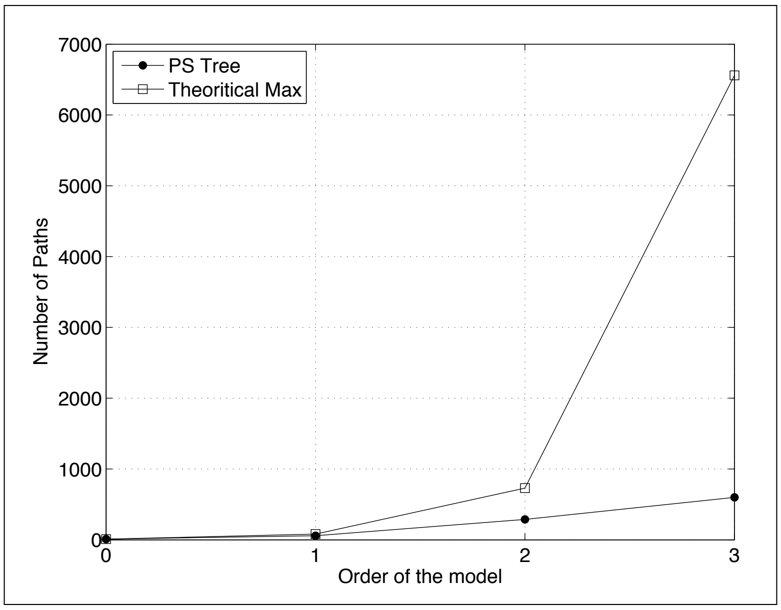


Figure 8: Comparison between number of nodes in the PST at a given level vs. the theoretical maximum

$S_t = \{s_1, \ldots, s_t\}$ is the corresponding sequence of discrete labels or states. In a HMM, to simplify inference, two independence assumptions are made: 1) x_i is independent of all other observations, given s_i, and 2) s_t depends only on s_{t-1}. With these assumptions, the joint distribution can be factored as follows: $P(X_t, S_t) = P(x_1|s_1)P(s_1)\prod_{i=1}^{T} P(s_i|s_{i-1})P(x_i|s_i)$. The relationship between observation and hidden state is given by an observation distribution $P_i(x_t|s_t)(i \in \{1, \ldots, N\}$, where N is the number of states). A HMM, then, is defined by a transition distribution $A = [a_{ij}]$ $(a_{ij} = P(s_t = j|s_{t-1} = i))$, a set of observation probability measures $B = \{b_i(S_t)\}_{i=1}^{N}$ $(b_i(S_t) = P_i(x_t|s_t))$, and an initial-state distribution π $(\pi_i = P(s_1) = i)$. Observations may be continuous or discrete. Prediction, in this context, means computing the next-symbol distribution given the observed sequence: $P(s_{t+1}|X_t, A, B)$. This can be solved efficiently by using the well-known forward algorithm (with a trivial modification to omit the emission probability for the unknown observation at time $t + 1$) [31] . If A and B are unknown, they can be estimated from training sequences using the Baum-Welch algorithm [6], an instance of the generalized expectation-maximization algorithm [22].

Next we describe the extension of the HMM to variable-order framework for tabla prediciton. The VLHMM consists of an ensemble of fixed-order HMMs, up to some maximum order. In practice, VLHMMs or mixed-order HMMs, become useful by considering a reduced set of transitions and contexts, specified using a PST [39]. The underlying PST might be learned from symbolic data (as we do here), or might be inferred from unlabeled sequences [42].

Here, the hidden states of the HMM correspond to tabla strokes, and the observation are the features vectors (MFCCs) computed from the audio. For a given fixed-order HMM, the next-stroke distribution can easily be computed by using the forward probabilities, which can be interpreted as the likelihood of reaching a given state at a given time-step, having correctly emitted all the features vectors up to that time. The fundamental complexity, compared to the visible case is that we must now consider all possible paths to that node, since we cannot be sure which path is correct. Fortunately, we do not need to exhaustively enumerate all the paths but can recursively compute the forward probabilities, as we describe below.

Still, such an algorithm has a time complexity that increases as a square of the number of states. This problem is particularly severe for high-order HMMs. A high-order HMM can be made into an equivalent first-order HMM by increasing the state space. For example, if we have 10 states then, the second-order HMM is equivalent to a 100-state first-order HMM, representing all the bi-gram state combinations. Thus the time-complexity can quickly become unmanageable, and for this reason high-order HMMs and VLHMMs are not widely used. Our approach is to consider a much reduced set of paths given in the PST. We use our symbolic data to learn which paths are in fact valid, and only compute the forward probabilities for these paths.

Each node in the PST corresponds to a given stroke type and has an emission distribution associated with it. Here we use a MVG with a covariance matrix that is pooled across all strokes. In a high-order HMM, the emission distribution might depend not just on the current state, but on several previous states. This would allow us to capture differences in the feature vectors due to interactions between adjacent strokes. For example the timbre of a stroke might differ depending on the previous stroke, both because the previous stroke might still be ringing, and because the fingering might be altered by the previous stroke. We leave this for future work, and here we simply use one MVG per stroke.

The mean vector and covariance matrix for each class, were estimated on the labeled training data and are passed directly to the HMM for use in the emission distributions. These were left clamped and were not adjusted by the forward-backward algorithm. The PST (Section 2.2) also stores the probability of arriving at a given node by a certain path; we simply multiply the probabilities at each node along the path. Thus the transition probabilites are efficiently stored in the PST. Note that because the PST is sparse, i.e. most high-order n-grams are unseen, this is much more efficient than storing all possible n-grams of a given length. Notice also that we are exploiting the fact that symbolic data is readily available, a key advantage in music problems compared with speech and other HMM application areas. Further, because hidden states correspond to strokes, and thus have a simple interpretation, estimating the parameters of the MVG can be done beforehand without using the forward-backward algorithm.

Formally our problem is the following. Given a sequence of observations $O_t = \{x_1, x_2, x_3, ..., x_t\}$, what is $P(\omega_{t+1}^j | O_t)$ for $j = 1, ..., c$, where c is the number of categories, and $P(\omega_{t+1}^j)$ represent the probability of the jth stroke at time $t + 1$. If we know the forward probability at time t for each stroke category, then this would simply be $P(\omega_{t+1}^j | O_t) = \sum_{i=1}^{c} \alpha_i^t a_{ij}$, where α_i^t is the probability of being in stroke category i at time t having emitted O_t. If a_{ij} is the probablity of making a transition from i to j then the forward probabilities can be calculated recursively as follows:

$$\alpha_j^t = \sum_{i=1}^{c} \alpha_i^{t-1} a_{ij} P(x_t | \omega_j) \tag{2}$$

However, as noted earlier this quickly becomes intractable as the state space is expanded to deal with high-order transitions. The PST allows us to consider a much smaller number of terms in this sum. In particular, to calculate $P(\omega_{t+1}^j | O_t)$, for an HMM of order m, we look for all nodes at level $m + 1$ that correspond to category j. Let the set of such nodes be A. The total number of nodes that fit this definition will be the number of terms in our sum, which will in general be much less than the theoretical maximum. Since each node has only one parent, we simply multiply the forward probability of the parent node by the transition probability from the parent to child. These partial probabilities are then summed for all $i \in A$ to get $P(\omega_{t+1}^j)$. In other words, we now have $\alpha_j^t = \sum_{i \in A} \alpha_i^{t-1} a_{ij} P(x_t | \omega_j)$. Figure 8 shows how the number of nodes grows with the depth of the tree for this data set compared with the theoretical maximum given by c^m. Between order 2 and 3 the number of possible nodes explodes, however the actual number of nodes in the

	Order 0	**Order 1**	**Order 2**	**Order 3**
Average	2.64	2.45	2.50	2.31
Median	1.23	1.20	1.20	1.16

Table 4: Graph showing the change in perplexity with increasing order of the VLHMM

PST is approximately an order of magnitude less than this.

What we have described so far is still a fixed order HMM. We can now extend this to a variable-length model by incorporating the smoothing techniques described in Section 2.3. Based on the results of the previous experiment, we decided to use the 1/N weighting scheme for smoothing. This weighted average across model orders gives us the combined next symbol distribution.

4.3 Results and Discussion

Evaluation was done using the same leave-one-out framework at the song-level. As noted above, the test song was synthesized using different samples than the training songs. For each test song, the entropy (and hence perplexity) was computed at each time step. The results are given in Table 5. It can be seen that the perplexity shows a small downward trend with the maximum order of the VLHMM. A multiple comparison of means test, using the Tukey-Cramer statistic, shows that differences between the various order VLHMMs are all significant at the .01 level, with the exception of the difference between order 1 and 2. One important point to note when comparing with the symbolic results is that here only 9 symbols were used compared with 35 for the symbolic data. Thus the same perplexity values for the symbolic data represent a better performance relative to the random baseline.

	Order 0	**Order 1**	**Order 2**	**Order 3**	**Composition Type**
Song 1	1.82	1.71	1.75	1.72	qaida
Song 2	4.79	4.38	4.36	4.3	laggi
Song 3	2.06	1.95	1.89	1.83	qaida
Song 4	2.77	2.47	2.4	2.35	qaida
Song 5	2.9	2.67	2.53	2.44	qaida
Song 6	2.22	2.06	2.01	2.02	qaida
Song 7	4.3	4.08	3.95	3.78	qaida
Song 8	1.59	1.47	1.44	1.41	rela
Song 9	2.06	1.97	2.9	1.84	rela
Song 10	2.03	1.97	1.93	1.89	tukda
Song 11	2.33	2.26	2.26	2.28	gat
Song 12	2.34	2.34	2.33	2.29	chakradhar
Song 13	1.94	1.83	1.82	1.8	qaida
Song 14	2.61	2.41	2.35	2.27	qaida
Song 15	3.06	2.85	2.81	2.74	qaida
Song 16	1.66	1.53	1.45	1.4	rela
Song 17	1.92	1.87	1.83	1.79	tukda
Song 18	3.4	3.11	2.97	2.83	gat
Song 19	2.31	2.19	2.13	2.14	chakradhar
Song 20	2.54	2.23	2.13	2.1	qaida
Song 21	2.66	2.32	2.23	2.16	qaida
Song 22	1.63	1.55	1.68	1.39	rela
Song 23	2.66	2.53	2.48	2.4	tukda
Song 24	3.95	3.53	3.39	3.23	gat
Song 25	2.89	2.77	2.73	2.6	chakradhar
Song 26	2.24	2.07	2.07	2.03	qaida
Song 27	2.26	2.11	2.05	2.03	qaida
Song 28	2.19	1.98	1.88	1.79	rela
Song 29	3.05	2.9	2.82	2.7	tukda
Song 30	31.24	28.35	27.29	26.73	gat
Song 31	4.42	4.46	4.39	4.31	chakradhar
Song 32	3.81	3.74	3.65	3.68	keharva
Song 33	21.21	15.16	13.73	13.55	keharva
Song 34	9.19	8.15	8.07	7.86	dadra
Median	2.575	2.33	2.34	2.275	

Table 5: Average perplexity for each test song, and the median perplexity across all test songs for each order.

The order 0 model is a special case and represents a lower bound on the perplexity rather than a true estimate. This is because we do not attempt to predict the next stroke. We simply calculate the posterior distribution given the current feature vector and use this distribution to compute the entropy and perplexity. In the other models there is no evidence to at the $t + 1$ step to guide the prediction. The order 0 model has much more information, and its perplexity is thus overly optimistic.

There is substantial variation of perplexity between pieces, with the VLHMM performing significantly better on some compositions than others. For example, the order 3 VLHMM ranges from a perplexity of 1.71 (Song 1) to 26.73 (Song 30)! Certain songs, such as 30, 33, and 34 proved difficult to model. However the models failure was instructive. Song 30 was a complex fixed composition (*gat*) with *bol* patterns quite different from the rest of the songs. Moreover it was composed in a rhythmic cycle of 12, whereas most of the composition are set to a rhythmic cycle of 16. This leads to patterns that are grouped in triples rather than duples. Song 33 was a *keharva theka*, and song 34 was a *dadra theka*. Ironically, while these are amongst the most internally repetitive songs, they are outliers in terms of their structure. Thus the most severe failures seem to stem from the VLHMMs inability to adapt to song-specific patterns.

Another source of error is the lack of the models knowledge about exact repetition. Nearly 40% of the compositions are *qaidas*. As we briefly noted in Section 1, these compositions are structured using almost exact repetition. Future models could be improved by augmenting the VLHMM with explicit, high-level formal knowledge, and by creating an STM.

Interestingly, Table 5 shows that there are systematic differences between types of compositions. The average perplexity for *qaidas* is 2.25, 1.57 for emphrelas, 4.97 for fixed compositions such as emphgats and emphtukdas, and 7.34 for the remaining, which are *thekas*. This is what would be expected from music theory. *Relas*, compositions that have long-sequences of repeated closed strokes, giving them their drum-roll feel, tend to be smooth and predictable. *Qaidas* are also somewhat predictable because of the strict way in which variations are formed from the theme. Fixed compositions on the other hand are often described as unpredictable, with unusual turns and juxtapositions of strokes. It is satisfying that the statistical model was able to quantify these qualitative assessments.

Despite the significant reductions in computation due to the PST, high-order models for a large database are still costly to compute. Computing the full VLH-MMs for order 1, 2 and 3 took .066, .198, and .534 seconds per stroke on a 2X3 GHz Mac. With our current implementation, orders 2 and 3 are too slow for re-altime use. We are currently exploring methods for further pruning the PST to speed up computation.

5 Conclusions and Future Work

We have developed a model of tabla sequences based on VLMMs and VLHMMs. To the best of our knowledge this is the first use of VLHMMs for music prediction. When predicting the next stroke or duration of a tabla composition VLMMs are highly predictive, with a minimum perplexity of 1.80 using a median perplexity of 1.07 using Multiple Viewpoint modeling. Incorporating a short-term model substantially improves performance compared with only using a corpus-wide long-term model. The reflects the fact that patterns in musical pieces often differ substantially from corpus-wide patterns, while being internally quite consistent. Moreover we showed that the incorporation of additional rhythmic viewpoints leads to small, but statistically significant improvements in the entropy of stroke predictions.

We have also shown that VLHMMs can be used to predict stroke continuations from audio. Our approach was based on computing the forward probabilities, for the high-order HMMs that constitute the VLHMM, efficiently using a PST that had been learned from stroke patterns in the symbolic data. Increasing the maximum model order from 1 to 3 decreases perplexity by a small but highly statistically significant, difference. Depending on the application, the computational burden of computing the high-order HMM is probably not justifiable given this performance improvement.

However, currently there is no analog to the STM in the hidden framework. Since the strokes are not visible it is more difficult to learn the song-specific n-grams. One approach that we plan to try is to allow the transition probabilities to adapt within a song, using the forward-backward algorithm. Results from the symbolic domain suggest that incorporating such song-level information could lead to dramatic improvements. For high-order HMMs, it may be the case that a song is not sufficiently long to adapt the background (LTM) model. A hybrid approach might be to cluster together compositions in the database that are similar, to form what might be called medium-term models (MTMs). When processing a song, the perplexity for each MTM could be calculated for the first few phrases, and the best MTM could then be used as in addition to the LTM, just as the STM is used in the symbolic prediction case.

6 Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant no. IIS-0855758.

References

[1] C. Ames. The Markov process as a compositional model: A survey and tutorial. *Leonardo*, 22(2):175–187, 1989.

[2] G. Assayag and S. Dubnov. Universal prediction applied to stylistic music generation. In *Mathematics and Music*, pages 147–160. Springer-Verlag, 2002.

[3] G. Assayag and S. Dubnov. Using factor oracles for machine improvisation. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 8(9):604–610, 2004.

[4] G. Assayag, S. Dubnov, and O. Delerue. Guessing the composer's mind: applying universal prediction to musical style. In *Proceedings of the 1999 International Computer Music Conference*, pages 496–499. San Francisco: ICMA, 1999.

[5] J. Aucouturier, F. Pachet, and M. Sandler. The way it sounds: Timbre models for analysis and retrieval of polyphonic music signals. *IEEE Transactions on Multimedia*, 7(6):1028–1035, 2005.

[6] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41:164–171, 1970.

[7] B. Bell and J. Kippen. Bol processor grammars. *Understanding music with AI: perspectives on music cognition*, pages 366–400, 1992.

[8] T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression.* Prentice Hall, 1990.

[9] Y. Bengio, V.-P. Lauzon, and R. Ducharme. Experiments on the application of IOHMMs to model financial returns series. *IEEE Transactions on Neural Networks*, 12(1):113 – 123, 2001.

[10] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[11] S. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the ACL*, pages 310–318. Association for Computational Linguistics, 1996.

[12] P. Chordia. *Automatic Transcription of Solo Tabla Music.* PhD thesis, Stanford University, Dec. 2005.

[13] P. Chordia, A. Albin, and A. Sastry. Evaluating multiple viewpoints models of tabla sequences. In *ACM Multimedia Workshop on Music and Machine Learning*, 2010.

[14] P. Chordia, A. Sastry, T. Mallikarjuna, and A. Albin. Multiple viewpoints modeling of tabla sequences. In *Proceedings of International Conference on Music Information Retrieval*, 2010.

[15] J. Cleary and W. Teahan. Experiments on the zero frequency problem. In *Proceedings of Data Compression Conference, DCC '95*, page 480, 28-30 1995.

[16] D. Conklin. Prediction and entropy of music. Master's thesis, University of Calgary (Canada), 1990.

[17] D. Conklin. Music generation from statistical models. In *Proceedings of the AISB 2003 Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, pages 30–35, 2003.

[18] D. Conklin and C. Anagnostopoulou. Representation and discovery of multiple viewpoint patterns. In *Proceedings of the 2001 International Computer Music Conference*, pages 479–485. International Computer Music Association, 2001.

[19] D. Conklin and J. G. Cleary. Modelling and generating music using multiple viewpoints. In *Proceedings of the First Workshop on AI and Music*, pages 125–137, Menlo Park, CA, 1988. AAAI Press.

[20] D. Conklin and I. H. Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24:51–73, 1995.

[21] A. Cont, S. Dubnov, and G. Assayag. Guidage: A fast audio query guided assemblage. In *Proceedings of International Computer Music Conference (ICMC)*. Copenhagen, September 2007.

[22] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[23] S. Dubnov. Analysis of musical structure in audio and midi using information rate. In *Proceedings of International Computer Music Conference*. ICMC, 2006.

[24] S. Dubnov, G. Assayag, and A. Cont. Audio oracle: A new algorithm for fast learning of audio structures. In *Proceedings of International Computer Music Conference (ICMC)*. Copenhagen, September 2007.

[25] S. Dubnov, G. Assayag, and R. El-Yaniv. Universal classification applied to musical sequences. In *Proceedings of the 1998 International Computer Music Conference*, pages 332–340. San Francisco: ICMA, 1998.

[26] S. Dubnov, G. Assayag, O. Lartillot, and G. Bejerano. Using machine-learning methods for musical style modeling. *IEEE Computers*, 36(10):73–80, 2003.

[27] A. E. Dutta. *Tabla: Lessons and Practice.* Ali Akbar College, 1995.

[28] S. R. Eddy. Hidden Markov models. *Current Opinion in Structural Biology*, 6(3):361 – 365, 1996.

[29] O. Gillet and G. Richard. Supervised and unsupervised sequence modeling for drum transcription. In *Proceedings of International Conference on Music Information Retrieval*, pages 219–224, 2007.

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

[31] B. H. Juang and L. R. Rabiner. Hidden Markov models for speech recognition. *Technometrics*, 33(3):251–272, 1991.

[32] O. Lartillot, S. Dubnov, G. Assayag, and G. Bejerano. Automatic modelling of musical style. In *Proceedings of the 2001 International Computer Music Conference*, pages 447–454. San Francisco: ICMA, 2001.

[33] K. Lee and M. Slaney. A unified system for chord transcription and key extraction using hidden Markov models. In *Proceedings of International Conference on Music Information Retrieval*, 2007.

[34] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*, pages 60–78. MIT Press, 2002.

[35] F. Pachet. The continuator: Musical interaction with style. In *Proceedings of International Computer Music Conference, Gotheborg (Sweden), ICMA*, pages 211–218, 2002.

[36] Pearce, H. Ruiz, Kapasi, Wiggins, and Bhattacharya. Unsupervised statistical learning underpins computational, behavioural and neural manifestations of musical expectation. *NeuroImage*, 50(1):302–313, 2010.

[37] M. Pearce. *The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Cognition.* PhD thesis, City University, London, 2005.

[38] M. Pearce, D. Conklin, and G. Wiggins. *Methods for Combining Statistical Models of Music*, volume 3310, pages 295–312. Springer Berlin, 2005.

[39] J. A. D. Preez, P. D. E. Barnard, and D. D. M. Weber. Efficient high-order hidden markov modelling. In *Proceedings of the International Conference on Spoken Language Processing*, pages 2911–2914, 1998.

[40] A. Rae. Generative rhythmic models. Master's thesis, Georgia Institute of Technology (Georgia), 2008.

[41] A. Rae and P. Chordia. Tabla gyan: An artificial tabla improviser. In *First International Conference on Computational Creativity (ICCCX)*, 2010.

[42] L. Schwardt and J. D. Preez. Efficient mixed-order hidden markov model inference. In *International Conference on Spoken Language Processing*, 2000.

[43] J. Serra, E. Gómez, and P. Herrera. Audio cover song identification and similarity: background, approaches, evaluation, and beyond. *Advances in Music Information Retrieval*, pages 307–332, 2010.

[44] J. L. Triviño-Rodriguez and R. Morales-Bueno. Using multiattribute prediction suffix graphs to predict and generate music. *Computer Music Journal*, 25(3):62–79, 2001.

[45] I. H. Witten and T. C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094, 1991.

[46] I. H. Witten, L. C. Manzara, and D. Conklin. Comparing human and computational models of music prediction. *Computer Music Journal*, 18(1):70–80, 1994.