CONVENTIONAL AND PERIODIC *N*-GRAMS IN THE TRANSCRIPTION OF DRUM SEQUENCES

Jouni K. Paulus, Anssi P. Klapuri

Tampere University of Technology, P.O.Box 553, FIN-33101 Tampere, Finland
{paulus,klap}@cs.tut.fi

ABSTRACT

In this paper, we describe a system for transcribing polyphonic drum sequences from an acoustic signal to a symbolic representation. Low-level signal analysis is done with an acoustic model consisting of a Gaussian mixture model and a support vector machine. For higher-level modelling, periodic *N*-grams are proposed to construct a "language model" for music, based on the repetitive nature of musical structure. Also, a technique for estimating relatively long *N*-grams is introduced. The performance of *N*-grams in the transcription was evaluated using a database of realistic drum sequences from different genres and yielded a performance increase of 7.6 % compared to a the use of only prior (unigram) probabilities with the acoustic model.

1. INTRODUCTION

Drums and percussive instruments are an essential part of contemporary music and especially of popular music. As a consequence, the recognition and transcription of rhythm sequences from acoustic signals to MIDI has become a topic of interest. Applications of this comprise e.g. automatic music transcription, light effects control, and music information retrieval in general. However, there has been relatively little previous research in this area (for an overview, see [2]). In most of the work in this field, e.g. in [2], the research has focused on the recognition of individual drum sounds without considering mixtures of simultaneous sounds. Also, typically no higher-level modelling of temporal dependencies in rhythm sequences has been attempted.

The purpose in this paper is to transcribe drum sound mixtures that appear in real rhythm sequences. The task has proven to be very difficult with low-level signal processing only. We propose to improve the recognition ability by using higher-level musicological "language models", based on conventional and periodic *N*-grams. In the periodic *N*-gram, the units that are used to predict the probability of a "word" at time *n* are picked at multiples of interval *L* before the word to be predicted, i.e., at time instances $k-(N-1)L, \ldots, k-2L, k-L$. The conventional *N*-gram is a special case of this where $L = 1$. Many basic elements of music exhibit periodically repeating patterns that vary over time. This observation can be made for harmonic and rhythmic elements at a wide range of different time scales. Conventional *N*-grams can be quite successfully used to predict melodies. However, when it comes to the other parts of music, such as accompaniment and rhythm section, instruments usually follows a periodically regular rather than a locally predictable pattern. The applicability of such models is evaluated and a technique which allows the estimation for relatively large values of *N* is proposed.

2. PROBABILISTIC MODEL FOR RHYTHM SEQUENCES

2.1. Notation

Each individual drum sound is associated with a code which uniquely identifies it. For example, the General MIDI standard defines codes for 47 different drum sounds. Let Ψ be a set of drum codes and *Y* the size of this set. Drum sounds are further classified into broader categories by defining a mapping from the set Ψ to a finite alphabet of symbols Σ which represents the drum categories. The size of the alphabet Σ is typically significantly smaller than that of Ψ. Although the size of Σ could be equal to that of Ψ, this kind of very extensive alphabet would limit the ability of the system to generalize or it would require huge amounts of training data in later steps. In this paper, we chose to use an alphabet size *S*=7, where the symbols represent bass drums, snare drums, hi-hats, cymbals, ride cymbals, tom toms, and percussion instruments, respectively. The percussion symbol class operates as a kind of left-over class which contains all the sounds that could not be fitted to any other category.

Words are defined to be unordered subsets of Σ, where each symbol may occur only once. A "word" is interpreted to represent a set of drum categories that are played simultaneously at a given instant of time, the term "simultaneously" to be defined more exactly in Sec. 2.2. A word can be written as a string of symbols $w_i=\{s_1,s_2,\ldots,s_l\}$, where $l \le S$. For example, $\{s_1,s_2\}$ is a word where sounds from two different categories play simultaneously. An empty word which does not contain any symbols is called *silence*.

V is the total number of word types in the language, that is, the *vocabulary size*. This can be calculated as $V = 2^S$. I.e. each word can be represented as a binary number, where one bit per one symbol indicates whether on not the symbol belongs to the word.

2.2. Rhythm sequences

A percussive music performance is modelled as follows. First, musical time is discretized by finding the *tatum* of the incoming musical performance. The term *tatum*, or, time quantum, refers to the shortest durational value in a musical composition that is still more than incidentally encountered. The other durational values (with few exceptions) are integer multiples of the tatum. Tatum is relative to tempo, and its value may gradually change over time, reflecting tempo fluctuations.

After the tatum of a performance has been estimated, a grid of equidistant tatum pulses is aligned with the performance, and each drum event is associated with the nearest grid point. The events that are associated to a same tatum grid point are considered as simultaneous. Following a common notation [4], the rhythm sequence can then be written as a sequence of words

$w_1 w_2 ... w_K$ as w_1^K, where exactly one word is generated per each grid point. If no drum events occur in the vicinity of a grid point, an empty word is generated.

2.3. Prior probabilities for words

Given a representative database of rhythm sequences, prior probabilities for different word types, $P(w_k), w_k = 0, ..., 127$ can be estimated by performing tatum estimation for each performance in the database and by counting the number of occurrences of each word type in the whole database in relation to the total number of all word occurrences in the database. From the point of view of *N*-grams these can be called *unigram* probabilities.

2.4. Conventional *N*-grams

N-grams have been found to be a convenient way of modelling the sequential dependencies in natural languages. An *N*-gram uses $N-1$ previous words to predict what the next word would be. This involves an $(N–1)^{th}$ order *Markov assumption*, i.e., that the probability of the next word depends only on the *N*–1 immediately preceding words. Applying the *N*–gram model, the probability of a word sequence can be calculated as

$$P(w_1^K) = \prod_{k=1}^{K} P(w_k | w_{k-N+1}^{k-1}) . \qquad (1)$$

Given a representative database, the *N*-gram probabilities $P(w_k | w_{k-N+1}^{k-1})$ can be estimated by counting the number of times a certain word occurs after a certain prefix, and dividing this by the count how many times the prefix occurs in the database:

$$P(w_k | w_{k-N+1}^{k-1}) = \frac{C(w_{k-N+1}^{k})}{C(w_{k-N+1}^{k-1})} . \qquad (2)$$

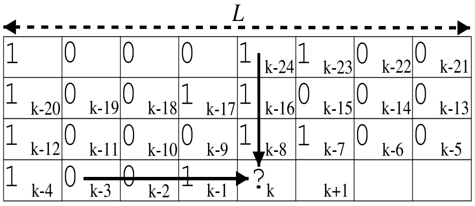
A typical problem with *N*-grams is that, even for a modest vocabulary size *V*, the size of the database does not suffice to estimate the probabilities for large *N*. The number of probabilities to be estimated for a given *V* and *N* is V^N, which requires a rather large database already for *N*=3. Because the training set is usually not extensive enough to estimate all possible *N*-gram probabilities reliably, some smoothing method is applied to the word counts before transforming them to probabilities. Here Witten-Bell smoothing was used [9].

In the case of rhythm sequences, a potential solution to the problem of insufficient data is to estimate *N*-gram probabilities for each individual symbol separately, instead of estimating them for words. Equations are the same as those given above, but with w_k getting only binary values (symbol does or does not occur in the word). Now the size of the vocabulary in the *N*-gram modelling shrinks to *V*=2, with “1” indicating that the symbol occurs at a given point, and “0” vice versa. In this case, *N*-gram probabilities for large values of *N* can be estimated. In our example, the number of probabilities to be estimated for word bigrams is 128^2, which is equal to 2^{14}, i.e., 14-grams for binary data.

Symbol *N*-grams can be used to predict words by combining the symbol-by-symbol predictions as

$$P(w_k | w_1^{k-1}) = \prod_{s_n \in w_k} P(s_n | w_1^{k-1}) \prod_{s_m \notin w_k} (1 - P(s_m | w_1^{k-1})) , \qquad (3)$$

where $s_n \cup s_m = \Sigma$. In this way, the training corpus can be utilized more efficiently. However, it is not memory-efficient to store long word *N*-grams, but only the symbol *N*-grams which are then combined according to Eq. (3) when processing files.



1	0	0	0	1 k-24	1 k-23	0 k-22	0 k-21
1 k-20	0 k-19	0 k-18	1 k-17	1 k-16	0 k-15	0 k-14	0 k-13
1 k-12	0 k-11	0 k-10	0 k-9	1 k-8	1 k-7	0 k-6	0 k-5
1 k-4	0 k-3	0 k-2	1 k-1	? k	k+1		

Fig. 1. The idea of periodic *N*-grams illustrated. Quantized time indices are represented with the smaller font. Horizontal arrow represents conventional quadrigram prediction and the vertical arrow periodic quadrigram prediction with *L* = 8.

Constructing *N*-grams separately for each symbol may lead to a situation where the predicted symbols at a given point of time together constitute a very improbable word. This problem can be solved by combining the use of symbol *N*-grams with word priors.

2.5. Periodic *N*-grams

Percussive rhythms exhibit periodicity at different time scales. This observation can be utilized by constructing periodic *N*-grams, as illustrated in Figure 1. Instead of the conventional *N*-gram model given in Eq. (1), the events that are used to predict the word w_k are taken at multiples of interval *L* earlier, at $w_{k-(N-1)L}^{k-L}$. The conventional *N*-gram is a special case of this where $L = 1$. Preferably, *L* is set to be the length of a relatively prominent repetition period, such as the rhythmic pattern, or the musical measure length. In Figure 1, *L* is set to be the length of the musical measure, and the bass drum is being predicted. The horizontal arrow represents a conventional quadrigram prediction, and the vertical arrow a periodic quadrigram prediction with period *L* = 8.

To apply periodic *N*-grams efficiently, a musical meter estimation process is required to find a suitable *L*. Musical meter estimation is a difficult problem in itself, and is above the scope of this paper. We have earlier presented a model which estimates the tatum, tactus (foot tapping rate), and the musical measure length from an acoustic musical signal [6]. We suggest setting *L* to correspond to the musical measure length.

3. ACOUSTIC MODELING

The foundation of signal analysis is in reliable low-level observations. Without being able to reliable extract information at the lowest level, no amount of higher level modelling is going to lead to a correct analysis.

Two independent acoustic models were constructed which together constitute the overall model which provides low-level observations for further statistical processing. The first model calculates the probability that a given audio segment (word) represents an empty word, i.e., silence. The second model assumes that the given segment is not silence and computes the probabilities for each of the 127 non-empty words to have generated the acoustic signal. The latter model is considered first.

3.1. Word recognition

The acoustic model for recognizing non-empty words is based on a Gaussian mixture model (GMM) classifier and Mel-frequency cepstral coefficients (MFCCs) and ΔMFCCs as features [8]. The model was constructed using the following steps:

- 100 random instances of each of the 127 non-empty words were synthesized by allotting each constituent sound (symbol) from a drum sound database and by mixing the sounds. This acoustic database was *not* used in the subsequent testing stage.
- For each sample, six MFCCs and six ΔMFCCs (after discarding the zeroth coefficient) were calculated in successive 20 ms frames with 75 % overlap up to 150 ms after the onset of the sounds. Feature vectors from all the 100 samples were catenated to a single matrix. These matrices were then mean and variance normalized.
- A GMM with two components was used to model the distribution of the feature values for each of the 127 non-empty signals.

3.2. Silence detection

In tests it proved out that the concept of an empty word or the silence was difficult to model, and preceding sounds which continue ringing at an empty grid point are easily confused with hi-hat sounds. Since silence and hi-hat are the two most often appearing word types (see Fig. 2.), this was a serious problem. It was approached with support vector machines, normally used in binary classification. For a detailed description, refer to [1]. The used implementation was *SVMlight*-toolkit [3] and a radial basis function (RBF) kernel was used.

The output of the SVM is a distance from the decision surface. Normally just the sign of the output matters. In our case, a probability value was needed because the *N*-grams are based on probabilistic computation. In [5] a sigmoid function is used for modifying the SVM output. As the distance from the hyperplane increases, the more probable it is that the sample is classified correctly. When choosing the “silence” class to be positive and the “sound” negative, the probability of this word being empty is then acquired from manipulating the SVM output $f(x)$ with sigmoid function:

$$P(w = \{\varnothing\}) = \frac{1}{1 + e^{-Kf(x)}} \tag{4}$$

where K is a constant determining the steepness of the sigmoid. Finally the result is scaled as

$$P'(w = \{\varnothing\}) = \max\{P(w \neq \{\varnothing\})\}P(w = \{\varnothing\}) \tag{5}$$

and catenated to the vector of non-empty word probabilities, which are scaled with $(1 - P(w = \{\varnothing\}))$.

The features that are used for silence detection are crest factor, kurtosis, skewness, zero-crossing rate, RMS-value and temporal centroid [2]. They are extracted from the whole length of the grid point. In the SVM training the samples for class “sound” are collected from extracting the features from 150 ms part of all generated word samples, whereas the samples for class “silence” are collected by extracting the same features at the time range 150-300 ms from all of the generated words. This mimics the situation that some sounds are still echoing in the background at the moments of silence.

4. VALIDATION EXPERIMENTS

The proposed methods were validated by applying it to the automatic transcription of drum sequences. Input to the system was presented as an acoustic signal, and the output consisted of a sequence of recognized words, i.e. a list of drum categories playing at each time instance. This can be written to a MIDI file or displayed to the user in a symbolic form.

We assume that the temporal framework, i.e. the tatum lengths T_0 and musical measure length LT_0, are given along with the acoustic input signals. A method which can automatically estimate these in acoustic input signals has been presented in [6]. However, in order to make the results in this paper as unambiguous as possible, we use manually determined tatum and musical measure length values in the following simulations. Thus the only remaining task is to decide what drum sounds play at each given time instant.

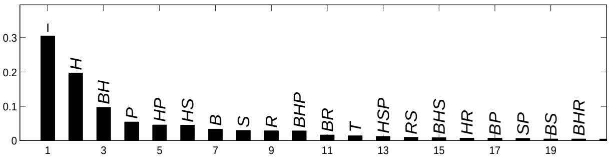


Fig. 2. Occurrence frequencies of the 20 most probable words in the Drumtrax database.

4.1. Estimation of the prior and *N*-gram probabilities

A commercial database, *Drumtrax Library* 3.0, was used to estimate the *N*-gram probabilities and prior probabilities for words. The database consists of 359 drum performances recorded in real time by studio drummers and organized into 14 different categories (genres). The performances are stored as MIDI files, the average length of which is 140 seconds. From each category five performances were taken to test set and the rest were used in calculating the *N*-grams. The test sequences were synthesized using *Timidity* program from MIDI files into monophonic audio files with sample rate of 44100 Hz.

Word *N*-gram probabilities were estimated by converting MIDI performances into a sequence of words as described in Sec. 2.2, and then by using Eq. (2). *N*-grams for each of the seven individual symbols were calculated by converting word sequences into binary sequences (see Sec. 2.2) before applying Eq. (2). For symbols, *N*-grams for *N*={5,10} were estimated, and for words, unigrams (*a priori* probabilities), bigrams and trigrams could be meaningfully estimated. Witten-Bell smoothing was applied to the probabilities to account for data sparseness [9]. Even though the training corpus was quite extensive not even all bigram probabilities were able to be approximated reliably, even such words exist that were not at all present at the training corpus. This is yet another motivation to use symbol *N*-grams and calculate word probabilities from them.

Figure 2 shows the prior probabilities of the 20 most frequently occurring words in the Drumtrax database. In the figure, characters, *B, S, H, C, T, R, P* refer to bass drum, snare drum, hi-hat, cymbal, tom tom, ride cymbal, and percussion, respectively, and *BH*, for example, means a bass drum and a hi-hat occurring simultaneously. The most probable word is the empty word (silence). As can be seen, the distribution is heavily concentrated to the few most probable words.

4.2. Models in comparison

The simplest recognition experiment was done using only the acoustic models. More exactly, the events at each tatum point of an incoming acoustic signal were classified to the most likely word model. When the word classification has been performed, all the symbols (drum sounds) at each time point are known.

In the second experiment, only the word priors (unigrams)

Table 1: Performance of different *N*-gram systems.

method	symbol error rate (%)	improvement from baseline (%)
acoustic model	76.1	
baseline	49.5	
1. conv. word bigram	47.2	4.7
2. per. word bigram	46.6	5.8
3. conv+per word bigram	47.1	4.9
4. conv. word trigram	46.9	5.2
5. per. word trigram	46.8	5.5
6. conv+per word trigram	46.5	6.0
7. conv. sym. quintagram	46.8	5.5
8. per. sym. quintagram	45.9	7.3
9. conv. sym decagram	45.7	7.6
10. per. sym. decagram	46.0	7.1

were used along with the acoustic model. An as could be predicted from the highly compacted distribution of Fig. 2, this makes a significant improvement to the acoustical model performance. This result operates also as the baseline result to which the *N*-gram enhancements are then added.

Several different *N*-gram types were added to the baseline: (1) conventional bigrams for words, (2) periodic bigrams for words, (3) both conventional and periodic bigrams for words, (4) conventional trigrams for words, (5) periodic trigrams for words, (6) both conventional and periodic trigrams for words, (7) conventional quintagrams (*N*=5) for symbols, (8) periodic quintagrams for symbols, (9) conventional decagrams (*N*=10) for symbols and (10) periodic decagrams for symbols.

Integration of the *N*-grams to the acoustic model was done simply by multiplying the probabilities of the acoustic model and the *N*-gram prediction. In the systems where conventional and periodic *N*-grams were used simultaneously, the periods-behind words were taken from along the path which was decoded up to that point. The final words to the transcription were chosen with greedy decoding. Globally optimizing Viterbi decoding was also tested for methods 1-6, but it had no major effect on the results.

4.3. Results

To be able to evaluate system transcription results automatically, a symbol error rate calculation formula was introduced. There exists three different error types that are omission (certain symbol should be present at word, but it is not), insertion (certain symbol should not be present at word, but it is) and substitution (combination of two previous error types, but counted only as one error). The final error rate is defined by

$$e = \sum_i \frac{(\aleph(w_i^R \cap w_i^T) + \max(0, \aleph(w_i^T) - \aleph(w_i^R)))}{\sum_i \aleph(w_i^R)} \qquad (6)$$

where i goes through all the test grid points, w_i^R is the set of symbols which are present in the word at that point in reference transcription, w_i^T is similarly the set of symbols in the word found by the system and $\aleph(\ )$ means the cardinality of the set.

The final error rates for all of the systems and improvement of *N*-gram methods from the baseline can be seen in Table 1. An overview of the errors for each of the symbols can be seen in Fig. 3. A large portion of the snare drum omissions are related to bass drum insertions. Another significant source for bass drum insertions is the omission of percussions. Also about half of the hi-hat omissions are related to insertion of silence. These are the main sources for errors and should be noted in future work.

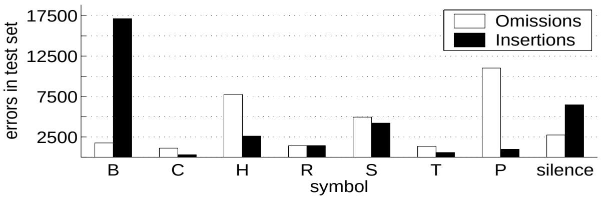


Fig. 3. Omission and insertion errors for each symbol.

5. CONCLUSIONS

We have presented a system for transcribing long sequences of drum sound mixtures. Higher-level statistical modelling improved the performance of acoustic models. Both periodic *N*-grams for words and *N*-grams for symbols have the ability to use information from a longer portion of the past events when predicting the words. The presented statistical models are not limited to percussive part only but can be used for other musical structures, too.

As suggested in [7], there are cases where increased memory length does not increase the prediction accuracy in same ratio. In those cases it would be useful to estimate the gained performance improvement when increasing the *N* and if it is below some predetermined threshold, adhere to the smaller one.

6. REFERENCES

[1] C. Cortes, V. Vapnik, "Support-Vector Networks," Machine Learning, vol. 20 no. 3, pp. 273-297, 1995.

[2] P. Herrera, A. Yeterian, F. Gouyon, "Automatic classification of drum sounds: a comparison of feature selection methods and classification techniques," Proc of 2nd International Conference on Music and Artificial Intelligence, pp. 69-80, 2002.

[3] T. Joachims, "Making Large-Scale SVM Learning Practical," In B. Schölkopf, C. Burges, A. Smola, (ed), "Advances in Kernel Methods-Support Vector Learning," MIT-Press, 1999.

[4] D. S. Jurafsky, J. H. Martin, "Speech and Language Processing," Prentice-Hall, 2000.

[5] A. Madevska-Bogdanova, D. Nikolic, "A new approach of modifying SVM outputs," Proc.of IEEE-INNS-ENNS International Joint Conference on Neural Networks 2000, vol. 6, pp. 395-398, 2000.

[6] J. Paulus, A. Klapuri, "Measuring the similarity of rhythmic patterns," Proc. of 3rd International Conference on Music Information Retrieval 2002, pp. 150-156, 2002.

[7] D. Ron, Y. Singer, N. Tishby, "The power of amnesia: learning probabilistic automata with variable memory length," Machine Learning, vol. 25, no. 2-3, pp. 117-149, 1996.

[8] L. Rabiner, B. H. Juang, "Fundamentals of Speech Recognition," Prentice-Hall, New Jersey, 1993.

[9] I. H. Witten, T. C. Bell, "The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression," IEEE Trans. on Information Theory, vol. 37, no. 4, pp. 1085-1094, 1991.