Connection Science
Vol. 00, No. 00, Month 200x, 1–22

RESEARCH ARTICLE

Predictive Models for Music

Jean-François Paiement[a,c,*], Yves Grandvalet[a,b] and Samy Bengio[c]

[a] *Idiap Research Institute, Centre du Parc, Rue Marconi 19, Case Postale 592, CH-1920 Martigny, Switzerland*; [b] *Heudiasyc, CNRS/Université de Technologie de Compiègne, Centre de Royallieu, B.P. 20529, 60205 Compiègne, France*; [c] *Google, 1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA*

(*Received 00 Month 200x; final version received 00 Month 200x*)

Modeling long-term dependencies in time series has proved very difficult to achieve with traditional machine learning methods. This problem occurs when considering music data. In this paper, we introduce predictive models for melodies. We decompose melodic modeling into two subtasks. We first propose a rhythm model based on the distributions of distances between subsequences. Then, we define a generative model for melodies given chords and rhythms based on modeling sequences of Narmour features. The rhythm model consistently outperforms a standard Hidden Markov Model in terms of conditional prediction accuracy on two different music databases. Using a similar evaluation procedure, the proposed melodic model consistently outperforms an Input/Output Hidden Markov Model. Furthermore, these models are able to generate realistic melodies given appropriate musical contexts.

Keywords: probabilistic models, time series prediction, melodies, rhythms, chord progressions

1. Introduction

Predictive models for music would be useful in a broad range of applications, from contextual music generation to on-line music recommendation and retrieval. However, modeling music involves capturing long-term dependencies in time series, which has proved very difficult to achieve with traditional statistical methods. Note that the problem of long-term dependencies is not limited to music, nor to one particular probabilistic model (Bengio et al. 1994).

In this paper we present graphical models that capture melodic structures in a given musical style using as evidence a limited amount of symbolic MIDI[1] data. A few models for music generation have already been proposed for music in general (Dubnov et al. 2003, Pachet 2003). While these models generate impressive musical results, we are not aware of proper quantitative comparisons between such models of music, that is for instance in terms of out-of-sample prediction accuracy, as it is done in Sections 3 and 5.

In the first part of this paper, we focus on modeling rhythmic sequences, ignoring for the moment other aspects of music such as pitch, timbre and dynamics. Many algorithms have been proposed for audio beat tracking (Dixon 2007, Scheirer 1998).

*Corresponding author. Email: paiment@idiap.ch

[1]MIDI stands for Musical Instrument Digital Interface, an industry-standard interface used on electronic musical keyboards and PCs for computer control of musical instruments and devices. In our work, we only consider note onsets and offsets in the MIDI signal.

ISSN: 0954-0091 print/ISSN 1360-0494 online

DOI: 10.1080/09540090xxxxxxxxxxxx
http://www.informaworld.com

Here, we consider rhythm modeling as a first step towards full melodic modeling. Our main contribution in this respect is to propose a generative model for distance patterns, specifically designed for capturing long-term dependencies in rhythms. In this work, distance patterns refer to distances between subsequences of equal length in particular positions. In Section 2, we describe the model, detail its implementation and present an algorithm using this model for rhythm prediction. The algorithm solves a constrained optimization problem, where the distance model is used to filter out rhythms that do not comply with the inferred structure. The proposed model is evaluated in terms of conditional prediction error on two distinct databases in Section 3, and a discussion follows.

With a reliable rhythm model available, we can turn our attention towards probabilistic modeling of melodies *given* rhythms and chord progressions. A chord is a group of three or more notes. A chord progression is simply a sequence of chords. In probabilistic terms, the current chord in a song can be seen as a latent variable (local in time) that conditions the probabilities of choosing particular notes in other music components, such as melodies or accompaniments. Chord changes occur at fixed time intervals in most of the musical genres, which makes them much simpler to detect (Lee and Slaney 2007) than beginnings and endings of musical notes, which can happen almost everywhere in music signal. Thus, knowing the relations between such chords and actual notes would certainly help to discover long-term musical structures in tonal music.

It is fairly easy to generate interesting chord progressions given melodies in a particular musical genre (Allan and Williams 2004, Paiement et al. 2006). However, the dual problem that we address in this paper is much more difficult. In Section 4.2, we describe melodic features derived from Narmour (1990) that will be used to put useful constraints on melodies based on musicological substantiation. We then introduce in Section 4.3 a probabilistic model of melodies *given* chords and rhythms that leads to significantly higher prediction rates than a simpler Markovian model. The combination of the rhythm model presented in Section 2 and the melodic model given chords of Section 4.3 leads to a predictive model of music that could be interesting in many applications. For instance, a good music model could help improve the poor performance of state-of-the-art transcription systems; it could as well be included in genre classifiers, automatic composition systems (Eck and Schmidhuber 2002), or algorithms for music information retrieval (Grachten et al. 2005).

2. Rhythm Model

We want to model rhythms in a dataset $\mathcal{X}$ consisting of rhythms of the same musical genre. We first quantize the database by segmenting each song in m time steps and associate each note to the nearest time step, such that all melodies have the same length m.[1] It is then possible to represent rhythms by sequences containing potentially three different symbols: 1) Note onset, 2) Note continuation, and 3) Silence. When using quantization, there is a one to one mapping between this representation and the set of all possible rhythms. Using this representation, symbol 2 can never follow symbol 3. Thus, each rhythm is represented by an m-dimensional vector $\mathbf{x}$ with discrete elements 1,2 or 3. Each rhythm sequence in the dataset is denoted by $\mathbf{x}^l \in \mathcal{X}$.

[1] This hypothesis is not fundamental in the proposed model and could easily be avoided if one would have to deal with more general datasets.

Hidden Markov Models (HMMs) are commonly used to model temporal data (Rabiner 1989). In principle, an HMM as described in Section 2.4 is able to capture complex regularities in patterns between subsequences of data, provided its number of hidden states is large enough. Thus, the HMM could be thought as a valid candidate for rhythm prediction. However, when dealing with music, such a model would lead to a learning process requiring a prohibitive amount of data: in order to learn long range interactions, the training set should be representative of the joint distribution of subsequences. To overcome this problem, we propose in Section 2.2 to summarize the joint distribution of subsequences by the distribution of their pairwise distances. This summary is clearly not a sufficient statistic for the distribution of subsequences, but its distribution can be learned from a limited number of examples. The resulting model, which generates distances, is then used to constrain the generation of subsequences. Moreover, empirical results obtained in Section 3 show that constraining the HMM with distributions over distances between subsequences significantly improves prediction accuracy.

2.1. *Graphical Models and EM*

The probabilistic models used in this paper are described using the graphical model framework. Graphical models (Lauritzen 1996) are useful to define probability distributions where graphs are used as representations for a particular factorization of joint probabilities. Vertices are associated with random variables. A directed edge going from the vertex associated with variable A to the one corresponding to variable B accounts for the presence of the term $P(B|A)$ in the factorization of the joint distribution of all the variables in the model. The process of computing probability distributions for a subset of the variables of the model given the joint distribution of all the variables is called *marginalization* (e.g. deriving $P(A, B)$ from $P(A, B, C)$). The graphical model framework provides efficient algorithms for marginalization and various learning algorithms can be used to learn the parameters of a model, given an appropriate dataset.

The Expectation-Maximization (EM) algorithm (Dempster et al. 1977) can be used to estimate the conditional probabilities of the hidden variables in a graphical model. Hidden variables are neither observed during training nor during evaluation of the models. These variables represent underlying phenomena that have an impact on the actual observations, but that cannot be observed directly. The EM algorithm proceeds in two steps applied iteratively over a dataset until convergence of the parameters. Firstly, the E step computes the expectation of the hidden variables, given the current parameters of the model and the observations of the dataset. Secondly, the M step updates the values of the parameters in order to maximize the joint likelihood of the observations, given the expected values of the hidden variables.

2.2. *Distance Model*

Music is characterized by strong hierarchical dependencies determined in large part by *meter*, the sense of strong and weak beats that arises from the interaction among hierarchical levels of sequences having nested periodic components. Such a hierarchy is implied in western music notation, where different levels are indicated by kinds of notes (whole notes, half notes, quarter notes, etc.) and where bars[1] establish measures of an equal number of beats. Meter and rhythm provide a framework

[1]Bars can also be called *measures.*

for developing musical melody. For example, a long melody is often composed by repeating with variation shorter sequences that fit into the metrical hierarchy (e.g. sequences of 4, 8 or 16 bars). It is well known in music theory that *distance patterns* are more important than the actual choice of notes in order to create coherent music (Handel 1993). For instance, measure 1 may always be similar to measure 5 in a particular musical genre. In fact, even random music can sound structured and melodic if it is built by repeating random subsequences with slight variations.

Traditionally, musicologists refer to repetition patterns in music with sequences of letters (e.g. AABA). Let us consider the simple pattern "AB". This notation does not tell to what extent the second part differs from the first. Instead of just stating if the second part is similar or not to the first one, we want to quantify the distances between the two parts in a corpus of music data. We can even go further and repeat this process hierarchically with various partition lengths. To do so, we introduce in this section a generative model for distance patterns and its application to rhythm sequences. Such a model is appropriate for most music data, where distances between subsequences of data exhibit strong regularities.

Suppose that we construct a *partition* of each sequence $\mathbf{x}^l$ by dividing it into ρ parts defined by $y_i^l = (x^l_{1+(i-1)m/\rho}, \ldots, x^l_{im/\rho})$ with $i \in \{1, \ldots, \rho\}$. We are interested in modeling the distances between these subsequences, given a suitable metric $d(y_i, y_j) : \mathbb{R}^{m/\rho} \times \mathbb{R}^{m/\rho} \to \mathbb{R}$. As was pointed out in the beginning of Section 2, the distribution of $d(y_i, y_j)$ for each specific choice of i and j may be more important when modeling rhythms (and music in general) than the actual choice of subsequences y_i.

Let $D(\mathbf{x}^l) = (d^l_{i,j})_{1\le i\le\rho, 1\le j\le\rho}$ be the distance matrix associated with each sequence $\mathbf{x}^l$, where $d^l_{i,j} = d(y_i^l, y_j^l)$. Since $D(\mathbf{x}^l)$ is symmetric and contains only zeros on the diagonal, it is completely characterized by the upper triangular matrix of distances *without* the diagonal. Hence,

$$p(D(\mathbf{x}^l)) = \prod_{i=1}^{\rho-1} \prod_{j=i+1}^{\rho} p(d^l_{i,j} | S_{l,i,j}) \tag{1}$$

where

$$S_{l,i,j} = \{d^l_{r,s} | (1 < s < j \text{ and } 1 \le r < s) \text{ or } (s = j \text{ and } 1 \le r < i)\} \ . \tag{2}$$

In words, we order the elements column-wise and do a standard factorization, where each random variable depends on the previous elements in the ordering. Hence, we do not assume any conditional independence between the distances.

Since $d(y_i, y_j)$ is a metric, we have that $d(y_i, y_j) \le d(y_i, y_k) + d(y_k, y_j)$ for all $i, j, k \in \{1, \ldots, \rho\}$. This inequality is usually referred to as the *triangle inequality*. Defining

$$\begin{aligned} \alpha^l_{i,j} &= \min_{k\in\{1,\ldots,(i-1)\}} (d^l_{k,j} + d^l_{i,k}) \text{ and} \\ \beta^l_{i,j} &= \max_{k\in\{1,\ldots,(i-1)\}} (|d^l_{k,j} - d^l_{i,k}|) \ , \end{aligned} \tag{3}$$

we know that given previously observed (or sampled) distances, constraints imposed by the triangle inequality on $d^l_{i,j}$ are simply

$$\beta^l_{i,j} \le d^l_{i,j} \le \alpha^l_{i,j} \ . \tag{4}$$

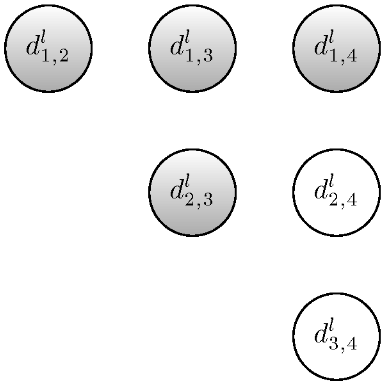

Figure 1. Each circle represents the random variable associated with the corresponding factor in Eq. (1), when $\rho = 4$. For instance, the conditional distribution for $d^l_{2,4}$ possibly depends on the variables associated to the grey circles.

One may observe that the boundaries given in Eq. (3) contain a subset of the distances that are on the conditioning side of each factor in Eq. (1) for each indices i and j. Thus, constraints imposed by the triangle inequality can be taken into account when modeling each factor of $p(D(\mathbf{x}^l))$: each $d^l_{i,j}$ must lie in the interval imposed by previously observed/sampled distances given in Eq. (4). Figure 1 shows an example where $\rho = 4$. Using Eq. (1), the distribution of $d^l_{2,4}$ would be conditioned on $d^l_{1,2}$, $d^l_{1,3}$, $d^l_{2,3}$, and $d^l_{1,4}$, and Eq. (4) reads $|d^l_{1,2} - d^l_{1,4}| \leq d^l_{2,4} \leq d^l_{1,2} + d^l_{1,4}$. Then, if subsequences y^l_1 and y^l_2 are close and y^l_1 and y^l_4 are also close, we know that y^l_2 and y^l_4 cannot be far. Conversely, if subsequences y^l_1 and y^l_2 are far and y^l_1 and y^l_4 are close, we know that y^l_2 and y^l_4 cannot be close.

2.3. *Modeling Relative Distances Between Rhythms*

When using the rhythm representation introduced in the beginning of Section 2, $d^l_{i,j}$ can simply be chosen to be the Hamming distance (i.e. counting the number of positions on which corresponding symbols are different). One could think of using more general edit distance such as the Levenshtein distance. However, this approach would not make sense psycho-acoustically: doing an insertion or a deletion in a rhythm produces a translation that alters dramatically the nature of the sequence. Putting it another way, rhythm perception heavily depends on the *position* on which rhythmic events occur. In the remainder of this paper, $d^l_{i,j}$ is the Hamming distance between subsequences y_i and y_j.

We now have to encode our belief that rhythms of the same musical genre have a common distance structure. For instance, drum beats in rock music can be very repetitive, except in the endings of every four measures, without regard to the actual beats being played. This should be accounted for in the distributions of the corresponding $d^l_{i,j}$.

With Hamming distances, the conditional distributions of $d^l_{i,j}$ in Eq. (1) should be modeled by discrete distributions, whose range of possible values *must* obey Eq. (4). Hence, we assume that the random variables $(d^l_{i,j} - \beta^l_{i,j})/(\alpha^l_{i,j} - \beta^l_{i,j})$ should be identically distributed for $l = 1, \ldots, n$. Empirical inspection of data supports this assumption. As an example, suppose that measures 1 and 4 always tend to be far away, that measures 1 and 3 are close, and that measures 3 and 4 are close; Triangle inequality states that 1 and 4 should be close in this case, but the desired model would still favor a solution with the greatest distance complying with the constraints imposed by triangle inequalities.

All these requirements are fulfilled if we model $d_{i,j} - \beta_{i,j}$ by a binomial distribution of parameters $(\alpha_{i,j} - \beta_{i,j}, p_{i,j})$, where $p_{i,j}$ is the probability that two symbols of subsequences y_i and y_j differ. With this choice, the conditional probability of getting $d_{i,j} = \beta_{i,j} + \delta$ would be

$$B(\delta, \alpha_{i,j}, \beta_{i,j}, p_{i,j}) = \binom{\alpha_{i,j} - \beta_{i,j}}{\delta} (p_{i,j})^{\delta} (1 - p_{i,j})^{(\alpha_{i,j} - \beta_{i,j} - \delta)} , \qquad (5)$$

with $0 \leq p_{i,j} \leq 1$. If $p_{i,j}$ is close to zero/one, the relative distance between subsequences y_i and y_j is small/large. However, the binomial distribution is not flexible enough since there is no indication that the distribution of $d_{i,j} - \beta_{i,j}$ is unimodal. We thus model each $d_{i,j} - \beta_{i,j}$ with a binomial *mixture* distribution in order to allow multiple modes. We thus use

$$p(d_{i,j} = \beta_{i,j} + \delta | S_{i,j}) = \sum_{k=1}^{c} w_{i,j}^{(k)} B(\delta, \alpha_{i,j}, \beta_{i,j}, p_{i,j}^{(k)}) \qquad (6)$$

with $w_{i,j}^{(k)} \geq 0$, $\sum_{k=1}^{c} w_{i,j}^{(k)} = 1$ for every indices i and j, and $S_{i,j}$ defined similarly as in Eq. (2). Parameters

$$\theta_{i,j} = \{w_{i,j}^{(1)}, \ldots, w_{i,j}^{(c-1)}\} \cup \{p_{i,j}^{(1)}, \ldots, p_{i,j}^{(c)}\}$$

can be learned with the EM algorithm on rhythm data for a specific music style.

In words, we model the *difference* between the observed distance $d_{i,j}^{l}$ between two subsequences and the minimum possible value $\beta_{i,j}$ for such a difference by a binomial mixture.

The parameters $\theta_{i,j}$ can be initialized to arbitrary values before applying the EM algorithm. However, as the likelihood of mixture models is not a convex function, one may get better models and speed up the learning process by choosing sensible values for the initial parameters. In the experiments reported in Section 3, the k-means algorithm for clustering (Duda et al. 2000) was used. More precisely, k-means was used to partition the values $(d_{i,j}^{l} - \beta_{i,j}^{l})/(\alpha_{i,j}^{l} - \beta_{i,j}^{l})$ into c clusters corresponding to each component of the mixture in Eq. (6). Let $\{\mu_{i,j}^{(1)}, \ldots, \mu_{i,j}^{(c)}\}$ be the centroids and $\{n_{i,j}^{(1)}, \ldots, n_{i,j}^{(c)}\}$ the number of elements in each of these clusters. We initialize the parameters $\theta_{i,j}$ with

$$w_{i,j}^{(k)} = \frac{n_{i,j}^{(k)}}{n} \quad \text{and} \quad p_{i,j}^{(k)} = \mu_{i,j}^{(k)}.$$

We then follow a standard approach (Bilmes 1997) to apply the EM algorithm to the binomial mixture in Eq. (6). Let $z_{i,j}^{l} \in \{1, \ldots, c\}$ be a hidden variable telling which component density generated $d_{i,j}^{l}$. For every iteration of the EM algorithm, we first compute

$$p(z_{i,j}^{l} = k | d_{i,j}^{l}, \alpha_{i,j}^{l}, \beta_{i,j}^{l}, \hat{\theta}_{i,j}) = \frac{\psi_{k,i,j,l}}{\sum_{t=1}^{c} \psi_{t,i,j,l}}$$

where $\hat{\theta}_{i,j}$ are the parameters estimated in the previous iteration, or the parameters guessed with k-means on the first iteration of EM, and

$$\psi_{k,i,j,l} = \hat{w}_{i,j}^{(k)} B(d_{i,j}^l, \alpha_{i,j}^l, \beta_{i,j}^l, p_{i,j}^{(k)}) \ .$$

Then, the parameters can be updated with

$$p_{i,j}^{(k)} = \frac{\sum_{l=1}^{n}(d_{i,j}^l - \beta_{i,j}^l)p(z_{i,j}^l = k|d_{i,j}^l, \alpha_{i,j}^l, \beta_{i,j}^l, \hat{\theta}_{i,j})}{\sum_{l=1}^{n}(\alpha_{i,j}^l - \beta_{i,j}^l)p(z_{i,j}^l = k|d_{i,j}^l, \alpha_{i,j}^l, \beta_{i,j}^l, \hat{\theta}_{i,j})}$$

and

$$w_{i,j}^{(k)} = \frac{1}{n}\sum_{l=1}^{n} p(z_{i,j}^l = k|d_{i,j}^l, \alpha_{i,j}^l, \beta_{i,j}^l, \hat{\theta}_{i,j}).$$

This process is repeated until convergence.

Note that using mixture models for discrete data is known to lead to *identifiability* problems. Identifiability refers here to the uniqueness of the representation (up to an irrelevant permutation of parameters) of any distribution that can be modeled by a mixture.

Estimation procedures may not be well-defined and asymptotic theory may not hold if a model is not identifiable. However, the model defined in Eq. (6) is identifiable if $\alpha_{i,j} - \beta_{i,j} > 2c - 1$ (Titterington et al. 1985, p.40). While this is the case for most $d_{i,j}$, we observed that this condition is sometimes violated. Whatever happens, there is no impact on the estimation because we only care about what happens at the distribution level: there may be several parameters leading to the same distribution, some components may vanish in the fitting process, but this is easily remedied, and EM behaves well.

As stated in Section 2.2, musical patterns form hierarchical structures closely related to meter (Handel 1993). Thus, the distribution of $p(D(\mathbf{x}^l))$ can be computed for many numbers of partitions within each rhythmic sequence. Let $\mathcal{P} = \{\rho_1, \ldots \rho_h\}$ be a set of numbers of partitions to be considered by our model, where h is the number of such numbers of partitions. The choice of $\mathcal{P}$ depends on the domain of application. Following meter, $\mathcal{P}$ may have dyadic[1] tree-like structure when modeling most music genres (e.g. $\mathcal{P} = \{2, 4, 8, 16\}$). Let $D_{\rho_r}(\mathbf{x}^l)$ be the distance matrix associated with sequence $\mathbf{x}^l$ divided into ρ_r parts. Estimating the joint probability $\prod_{r=1}^{h} p(D_{\rho_r}(\mathbf{x}^l))$ with the EM algorithm as described in this section leads to a model of the distance structures in rhythms datasets. Suppose we consider 16 bar songs with four beats per bar. Using $\mathcal{P} = \{8, 16\}$ would mean that we consider pairs of distances between every group of two measures ($\rho = 8$), and every single measures ($\rho = 16$).

One may argue that our proposed model for long-term dependencies is rather unorthodox. However, simpler models like Poisson or Bernoulli process (we are working in discrete time) defined over the whole sequence would not be flexible enough to represent the particular long-term structures in music.

2.4. *HMMs*

The HMM model (Rabiner 1989) is a well known probabilistic model for time series. Let $\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^n\}$ be a dataset of rhythm sequences, where all the

[1]Even when considering non-dyadic measures (e.g. a three-beat waltz), the *very large* majority of the hierarchical levels in metric structures follow dyadic patterns in most tonal music (Handel 1993).

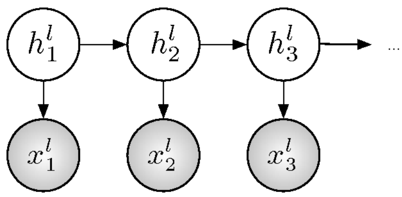

Figure 2. Hidden Markov Model. Each node is associated to a random variable and arrows denote conditional dependencies. When learning the parameters of the model, white nodes are hidden whereas grey nodes are observed.

sequences contain m elements: $\mathbf{x}^l = (x_1^l, \ldots, x_m^l)$, $l = 1, \ldots, n$. Furthermore, let $\mathbf{h}^l = (h_1^l, \ldots, h_m^l)$ be the corresponding sequence of states for a discrete hidden variable synchronized with $\mathbf{x}^l$.The joint probability of the rhythm sequence $\mathbf{x}^l$ and hidden states $\mathbf{h}^l$ estimated by an HMM is given by

$$p_{\text{HMM}}(\mathbf{x}^l, \mathbf{h}^l) = p_\pi(h_1^l) p_o(x_1^l | h_1^l) \prod_{t=2}^{m} p_{\bar{o}}(h_t^l | h_{t-1}^l) p_o(x_t^l | h_t^l) \ , \tag{7}$$

where the $p_{\bar{o}}(.|.)$ terms are called transition probabilities, the $p_o(.|.)$ terms are called emission probabilities, and the $p_\pi(.)$ is the initial probability of the first state of the hidden variable. This model is presented in Figure 2, following standard graphical model formalism. Each node is associated to a random variable and arrows denote conditional dependencies. The probability distributions p_π, $p_{\bar{o}}$, and p_o are multinomials, whose parameters can be learned by the EM algorithm.

2.5. *Conditional Prediction*

For most music applications, it would be particularly helpful to know which sequence $\hat{x}_s, \ldots, \hat{x}_m$ maximizes $p(\hat{x}_s, \ldots, \hat{x}_m | x_1, \ldots, x_{s-1})$. Knowing which musical events are the most likely given the past $s-1$ observations would be useful both for prediction and generation. Note that in the remainder of the paper, we refer to prediction of musical events given past observations only for notational simplicity. All the models presented in this paper could be used to predict any part of a music sequence given any other part with only minor modifications.

While the described modeling approach captures long range interactions in the music signal, it has two shortcomings. First, it does not model local dependencies: it does not predict how the distances in the smallest subsequences (i.e. with length smaller than $m/\max(\mathcal{P})$) are distributed on the events contained in these subsequences. Second, as the mapping from sequences to distances is many to one, there exists several admissible sequences $\mathbf{x}^l$ for a given set of distances. These limitations are addressed by using another sequence learner designed to capture short-term dependencies between musical events. Here, we use a standard HMM (Rabiner 1989), as described in Section 2.4.

The two models are trained separately using their respective version of the EM algorithm. For predicting the continuation of new sequences, they are combined by choosing the sequence that is most likely according to the local HMM model, provided it is also plausible regarding the model of long-term dependencies. Let $p_{\text{HMM}}(\mathbf{x}^l)$ be the probability of observing sequence $\mathbf{x}^l$ estimated by the HMM after training. The final predicted sequence is the solution of the following optimization

(1) Initialize $\hat{x}_s, \ldots, \hat{x}_m$ using Eq. (10);
(2) Set $j = s$ and set `end` = `true`;
(3) Set $\hat{x}_j = \arg\max_{a \in A}[\log p_{\text{HMM}}(\mathbf{x}^*|x_1, \ldots, x_{s-1}) + \lambda \sum_{r=1}^{h} \log p(D_{\rho_r}(x_1, \ldots, x_{s-1}, \mathbf{x}^*))]$
where $\mathbf{x}^* = (\hat{x}_s, \ldots, \hat{x}_{j-1}, a, \hat{x}_{j+1}, \ldots, \hat{x}_m)$
(4) If $\hat{x}_j$ has been modified in the last step, set `end` = `false`.
(5) If $j = m$ and `end` = `false`, go to 2;
(6) If $j < m$, set $j = j + 1$ and go to 3;
(7) Return $\hat{x}_s, \ldots, \hat{x}_m$.

Figure 3. Simple optimization algorithm to maximize $p(\hat{x}_i, \ldots, \hat{x}_m|x_1, \ldots, x_{i-1})$

problem:

$$\begin{cases} \max\limits_{\tilde{x}_s, \ldots, \tilde{x}_m} & p_{\text{HMM}}(\tilde{x}_s, \ldots, \tilde{x}_m|x_1, \ldots, x_{s-1}) \\ \text{subject to} & \prod_{r=1}^{h} p(D_{\rho_r}(\mathbf{x}^l)) \geq P_0 \end{cases}, \tag{8}$$

where P_0 is a threshold. In practice, one solves a Lagrangian formulation of problem (8), where we use log-probabilities for computational reasons:

$$\max_{\tilde{x}_s, \ldots, \tilde{x}_m} [\log p_{\text{HMM}}(\tilde{x}_s, \ldots, \tilde{x}_m|x_1, \ldots, x_{s-1}) + \lambda \sum_{r=1}^{h} \log p(D_{\rho_r}(\mathbf{x}^l))] , \tag{9}$$

where tuning λ has the same effect as choosing a threshold P_0 in problem (8) and can be done by cross-validation.

Multidimensional Scaling (MDS) is an algorithm that tries to embed points (here "local" subsequences) into a potentially lower dimensional space while trying to be faithful to the pairwise affinities given by a "global" distance matrix. Here, we propose to consider the prediction problem as finding sequences that maximize the likelihood of a "local" model of subsequences under the constraints imposed by a "global" generative model of distances between subsequences. In other words, solving problem (8) is similar to finding points such that their pairwise distances are as close as possible to a given set of distances (i.e. minimizing a stress function in MDS). Naively trying all possible subsequences to maximize (9) leads to $O(|A|^{(m-s+1)})$ computations. Instead, we propose to search the space of sequences using a variant of the Greedy Max Cut (GMC) method (Rohde 2002) that has proven to be optimal in terms of running time and performance for binary MDS optimization.

The subsequence $\hat{x}_s, \ldots, \hat{x}_m$ can be simply initialized with

$$(\hat{x}_s, \ldots, \hat{x}_m) = \max_{\tilde{x}_s, \ldots, \tilde{x}_m} p_{\text{HMM}}(\tilde{x}_s, \ldots, \tilde{x}_m|x_1, \ldots, x_{s-1}) \tag{10}$$

using the local HMM model. The complete optimization algorithm is described in Figure 3. For each position, we try every admissible symbol of the alphabet and test if a change increases the probability of the sequence. We stop when no further change can increase the value of the utility function. Obviously, many other methods could have been used to search the space of possible sequences $\hat{x}_s, \ldots, \hat{x}_m$, such as simulated annealing (Kirkpatrick et al. 1983). Our choice is motivated by simplicity and the fact that it yields excellent results, as reported in the following section.

Note that the algorithm in Figure 3 does not define by itself a generative model of rhythms. It only provides an approximation of the most probable sequence according to an implicitly defined probabilistic model. However, the conditioning set of variables in the objective value of problem (8) may be empty if we set the value of s to 1 (i.e. the HMM term becomes a joint distribution of all the observations), in which case problem (8) defines a proper generative model of observed data. A naive sampling algorithm for this model would be to sample the HMM distribution, and to use the distance constraint as a rejection criterion. The parameter P_0 would thus determine indirectly the proportion of rejected samples during the sampling process.

3. Rhythm Prediction Experiments

Two databases from different musical genres were used to evaluate the proposed model. Firstly, 47 jazz standards melodies (Sher 1988) were interpreted and recorded by the first author in MIDI format. Appropriate rhythmic representations as described in Section 2.3 have been extracted from these files. The complexity of the rhythm sequences found in this corpus is representative of the complexity of common jazz and pop music. We used the last 16 bars of each song to train the models, with four beats per bar. Two rhythmic observations were made for each beat, yielding observed sequences of length 128. We also used a subset of the Nottingham database[1] consisting of 53 traditional British folk dance tunes called "hornpipes". In this case, we used the first 16 bars of each song to train the models, with four beats per bar. Three rhythmic observations were made for each beat, yielding observed sequences of length 192. The sequences from this second database contain no silence (or rest), leading to sequences with binary states.

The goal of the proposed model is to predict or generate rhythms *given* previously observed rhythm patterns. As pointed out in Section 1, such a model could be particularly useful for music information retrieval, transcription, or music generation applications. Let $\varepsilon_i^j = 1$ if $\hat{x}_i^j = x_i^j$, and 0 otherwise, with $\mathbf{x}^j = (x_1^j, \ldots, x_m^j)$ a test sequence, and $\hat{x}_i^j$ the output of the evaluated prediction model on the i-th position when given $(x_1^j, \ldots, x_s^j)$ with $s < i$. Hence, we fix the value of s in each experiment to be the number of rhythm symbols observed by the model before trying to make predictions. Assume that the dataset is divided into K folds $T_1, \ldots, T_K$ (each containing different sequences), and that the k-th fold T_k contains n_k test sequences. When using cross-validation, the accuracy Acc of an evaluated model is given by

$$Acc = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{n_k} \sum_{j \in T_k} \frac{1}{m-s} \sum_{i=s+1}^{m} \varepsilon_i^j \ . \tag{11}$$

Note that, while the prediction accuracy is simple to compute and to apprehend, other performance criteria, such as ratings provided by a panel of experts, should be more appropriate to evaluate the relevance of music models. We plan to define such an evaluation protocol in future work. We used 5-fold double cross-validation to estimate the accuracies. Double cross-validation is a recursive application of cross-validation that enables to jointly optimize the hyper-parameters of the model and evaluate its generalization performance. Standard cross-validation is applied

[1] `http://www.cs.nott.ac.uk/~ef/music/database.htm`.

Table 1. Accuracy (the higher the better) for best models on the jazz standards database.

Observed	Predicted	HMM	Global
32	96	34.5%	54.6%
64	64	34.5%	55.6%
96	32	41.6%	47.2%

Table 2. Accuracy (the higher the better) for best models on the hornpipes database.

Observed	Predicted	HMM	Global
48	144	75.1%	83.0%
96	96	75.6%	82.1%
144	48	76.6%	80.1%

to each subset of $K-1$ folds with each hyper-parameter setting and tested with the best estimated setting on the remaining hold-out fold. The reported accuracies are the averages of the results of each of the K applications of simple cross-validation during this process.

For the baseline HMM model, double cross-validation optimizes the number of possible states for the hidden variables. In the reported experiments, we tried 2 to 20 possible states. In the case of the model with distance constraints, referred to as the global model, the hyper-parameters that were optimized are the number of possible states for hidden variables in the local HMM model (i.e. 2 to 20), the Lagrange multiplier λ, the number of components c (common to all distances) for each binomial mixture, and the choice of $\mathcal{P}$, i.e. which partitions of the sequences to consider. Values of λ ranging between 0.1 and 4 and values of c ranging between 2 and 5 were tried during double cross-validation. Since music data commonly shows strong dyadic structure following meter, many subsets of $\mathcal{P} = \{2, 4, 8, 16\}$ were allowed during double cross-validation.

Note that the baseline HMM model is a poor benchmark on this task, since the predicted sequence, when prediction consists in choosing the most probable subsequence given previous observations, only depends on the state of the hidden variable at time s. This observation implies that the number of possible states for the hidden variables of the HMM upper-bounds the number of different sequences that the HMM can predict. However, this behavior of the HMM does not question the validity of the reported experiments. The main goal of this quantitative study is to measure to what extent distance patterns are present in music data and how well these dependencies can be captured by the proposed model. What we really want to measure is how much gain we observe in terms of out-of-sample prediction accuracy when using an arbitrary model if we impose additional constraints based on distance patterns. That being said, it would be interesting to measure the effect of appending distance constraints to more complex music prediction models (Dubnov et al. 2003, Pachet 2003).

Results in Table 1 for the jazz standards database show that considering distance patterns significantly improves the HMM model. One can observe that the baseline HMM model performs much better when trying to predict the last 32 symbols. This is due to the fact that this database contains song endings. Such endings contain many silences and, in terms of accuracy, a useless model predicting silence at any position performs already well. On the other hand, the endings are generally different from the rest of the rhythm structures, thus harming the performance of the global model when just trying to predict the last 32 symbols. Results in Table 2 for the hornpipes database again show that the prediction accuracy of the global model is consistently better than the prediction accuracy of the HMM, but the difference is less marked. This is mainly due to the fact that this dataset only

Table 3. Accuracy over the last 64 positions for many sets of partitions $\mathcal{P}$ on the jazz database, given the first 64 observations. The higher the better.

$\mathcal{P}$	Global
$\{2\}$	49.3%
$\{2,4\}$	49.3%
$\{2,4,8\}$	51.4%
$\{2,4,8,16\}$	55.6%

contains two symbols, associated to note onset and note continuation. Moreover, the frequency of these symbols is quite unbalanced, making the HMM model much more accurate when almost always predicting the most common symbol.

In Table 3, the set of partitions $\mathcal{P}$ is not optimized by double cross-validation. Results are shown for different fixed sets of partitions. The best results are reached with "deeper" dyadic structure. This is a good indication that the basic hypothesis underlying the proposed model is well-suited to music data, namely that dyadic distance patterns exhibit strong regularities in music data. We did not compute accuracies for $\rho > 16$ because it makes no sense to estimate distribution of distances between too short subsequences.

One could wonder why we restricted our proposed model to metric distances. It would indeed be possible to design a model for non metric distances, such as subjective human similarity judgments. However, we want to emphasize that specifically dealing with metrics has many advantages. First, we do not have subjective measures between our rhythm subsequences available. In contrast, the Hamming distance is an easily computed objective dissimilarity that is psychoacoustically relevant and intuitive given our proposed representation for rhythms. Since Hamming distance is a metric, not modeling the metric constraints would add useless degrees of freedom to a model of such distances, since we know how to model these constraints directly. Second, an embedding of rhythm subsequences could be built using the proposed metric (i.e. Hamming distances) that could be used as a preprocessing step to other models of subsequences. One could design algorithms that would first draw samples from a generative model of metric distances like ours, and then generate actual observations from the sampled metrics. This would be much more difficult if sampling from a model of non metric distances.

4. Melodic Model

With a reliable predictive model of rhythms available, we can now turn our attention to a more difficult problem, which is to model melody notes, *given* rhythm and chord progressions. In order to do so, we proceed iteratively in three steps depicted schematically in Figure 4. We first model rhythms with the model presented in Section 2. Then, we model features that represent the plausibility of sequences of notes. These "Narmour" features, introduced in Section 4.2, are computed for each sequence of three consecutive notes. Their prediction is an interesting intermediate problem since the cardinality of such features is much lower than the number of sequences of three notes. Moreover, such features are descriptive of the perceptual expectancy of a particular group of three notes. As stated in Section 1, chords can be seen as latent variables (local in time) that condition the probabilities of choosing particular notes in a melody. However, chords do not describe longer term melodic structure. This is why we propose to use Narmour features as sequences of constraints on the choices of melody notes. In Section 4.3, we describe a proba-

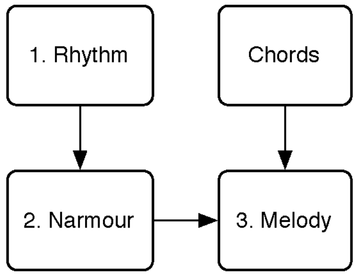

Figure 4. Schematic overview of the proposed melodic model. We first model rhythms. Then, we model Narmour features sequences given rhythms. Finally, we model actual melodies given Narmour features and chords.

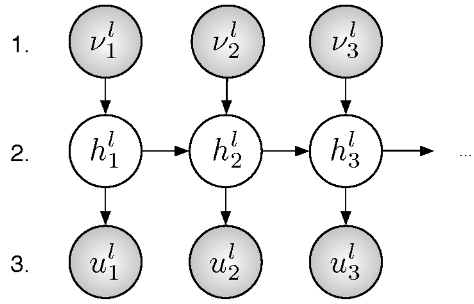

Figure 5. Variant of an IOHMM model for MIDI notes given chords. The variables in level 1 are always observed and correspond to chords. Variables in level 2 are hidden, while variables in level 3 correspond to melodic notes. All variables in grey are observed during training.

bilistic model for melody notes given Narmour features *and* chord progressions.

Results reported in Section 5 show that using sequences of Narmour features as constraints leads to much better prediction accuracy than the direct baseline approach using the IOHMM model described in the following section.

4.1. *IOHMMs*

A simple probabilistic model for melodies given chords can be designed by adding input variables to the HMM of Section 2.4. Let $\mathcal{U} = \{\mathbf{u}^1, \ldots, \mathbf{u}^n\}$ be a dataset of varying length melodies, where melody $\mathbf{u}^l$ has length g_l, $\mathbf{u}^l = (u_1^l, \ldots, u_{g_l}^l)$. Each melodic line is composed of notes u_i^l in the MIDI standard, $u_i^l \in \{0, \ldots, 127\}$. The melodies in dataset $\mathcal{U}$ are synchronized with rhythms in the dataset $\mathcal{X}$ defined as in Section 2. The length g_l of melodic line $\mathbf{u}^l$ corresponds thus to the number of note onsets (symbol 1) in rhythm sequence $\mathbf{x}^l$. In addition, let $\boldsymbol{\nu}^l = (\nu_1^l, \ldots, \nu_{g_l}^l)$ be the chord progression corresponding to the l-th melody. Here, each ν_t^l takes a discrete value within the number of different chords in the dataset. The joint probability of each sequence $\mathbf{u}^l$, its associated chord progression $\boldsymbol{\nu}^l$, and hidden states $\mathbf{h}^l$ can be modeled by

$$p_{\text{IOHMM}}(\mathbf{u}^l, \boldsymbol{\nu}^l, \mathbf{h}^l) = p_i(\nu_1^l) p_\pi(h_1^l|\nu_1^l) p_o(u_1^l|h_1^l) \prod_{t=2}^{g_l} p_i(\nu_t^l) p_{\bar{o}}(h_t^l|h_{t-1}^l, \nu_t^l) p_o(u_t^l|h_t^l) \ . \tag{12}$$

This model, shown in Figure 5, is a specific Input/Output Hidden Markov Model (IOHMM), as introduced by Bengio and Frasconi (1996). Usual IOHMMs have additional links connecting directly the input variables (level 1) to the outputs (level 3). We removed these links to decrease to number of parameters in the model, and thus being less prone to overfit the training data.

The probability distributions p_π, p_i, $p_{\bar{o}}$, and p_o are multinomials, as in Equation (7), and the model is learned by the standard EM algorithm. Marginalization must be carried out in this model both for learning (during the expectation step of the EM algorithm) and for evaluation. Exact marginalization with the standard Junction Tree Algorithm (Lauritzen 1996) is usually tractable in IOHMMs because of their limited complexity. Performance of the IOHMM in terms of melodic prediction accuracy given chords is presented in Section 5.

4.2. *Narmour Features*

In this section, we introduce melodic features that will prove to be useful for melodic prediction. The Implication-Realization (I-R) model has been developed by Narmour (1990, 1992) as a theory of musical expectation. This fairly complex musicological model was then simplified and implemented by Schellenberg (1997), who proposed a formal analysis of each sequence of three consecutive notes, according to five perceptual items: registral direction, intervallic difference, registral return, proximity, and closure, as described later in this section. The model returns five scores measuring expectancy according to these five criteria, and, according to Narmour's theory, high perceptual expectancy incurs high cumulative scores. This model was empirically shown to be relevant in information retrieval applications (Grachten et al. 2005).

In this paper, our goal is quite different. Instead of quantifying melodic expectancy, we design a probabilistic model of melodic sequences given chords. We propose to collectively use the Narmour principles as discrete features to characterize each sequence of three consecutive notes. In the remainder of this paper, we refer to these features as *Narmour features*. There is far fewer possible Narmour features (108 in our implementation) than possible groups of three notes (128^3 if we consider all MIDI notes). Given that observation, we expect that modeling sequences of Narmour features should be easier than modeling actual sequences of notes. We describe in Section 4.3 how we propose to generate actual melodies given sequences of Narmour features.

Our particular implementation of the Narmour features is mostly derived from Schellenberg (1997). We simply define the *interval* v_t between two notes u_t and u_{t-1} to be the difference $v_t = u_{t-1} - u_t$ between their MIDI note numbers. Interval has to be taken here in its musicological sense, which is not related to the usual mathematical definition: an interval is an integer that counts the number of semi-tones between two notes. Each Narmour principle can be computed for any sequence of three consecutive notes, corresponding to two intervals. In Narmour's theory, the first interval is referred to as the *Implication* while the second interval corresponds to the *Realization* of a melodic pattern of three notes. We define the sign function as

$$\mathrm{sgn}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases} .$$

The registral direction principle states that continuation in pitch direction is expected after small intervals and that large intervals imply a change of direction.

We define

$$\mathrm{rm}_t = \begin{cases} 0 & \text{if } |v_{t-1}| > 6 \text{ and } \mathrm{sgn}(v_{t-1}) = \mathrm{sgn}(v_t) \\ 1 & \text{if } |v_{t-1}| \leq 6 \\ 2 & \text{if } |v_{t-1}| > 6 \text{ and } \mathrm{sgn}(v_{t-1}) \neq \mathrm{sgn}(v_t) \end{cases}$$

to be the Narmour feature scoring the registral direction principle computed on arbitrary MIDI notes u_{t-2}, u_{t-1}, and u_t.

The intervallic difference principle says that small intervals imply similar-sized realized intervals and that large implicative intervals imply relatively smaller realized intervals. Formally,

$$\mathrm{id}_t = \begin{cases} 1 & \text{if } |v_{t-1}| < 6 \text{ and } \mathrm{sgn}(v_{t-1}) \neq \mathrm{sgn}(v_t) \text{ and } ||v_{t-1}| - |v_t|| < 3 \\ 1 & \text{if } |v_{t-1}| < 6 \text{ and } \mathrm{sgn}(v_{t-1}) = \mathrm{sgn}(v_t) \text{ and } ||v_{t-1}| - |v_t|| < 4 \\ 1 & \text{if } |v_{t-1}| > 6 \text{ and } |v_{t-1}| \geq |v_t| \\ 0 & \text{otherwise} \end{cases}$$

is the Narmour feature scoring the intervallic difference principle.

The registral return principle states that the second tone of a realized interval is expected to be very similar to the original pitch (within 2 semi-tones). Thus, we define the following scoring function

$$\mathrm{rr}_t = \begin{cases} 1 & \text{if } |v_t + v_{t-1}| \leq 2 \\ 0 & \text{otherwise.} \end{cases}$$

Then, the closure principle states that either melody changes direction, or that large intervals are followed by a relatively smaller interval. This feature is scored by

$$\mathrm{cl}_t = \begin{cases} 2 & \text{if } \mathrm{sgn}(v_{t-1}) \neq \mathrm{sgn}(v_t) \text{ and } |v_{t-1}| - |v_t| > 2 \\ 1 & \text{if } \mathrm{sgn}(v_{t-1}) \neq \mathrm{sgn}(v_t) \text{ and } |v_{t-1}| - |v_t| < 3 \\ 1 & \text{if } \mathrm{sgn}(v_{t-1}) = \mathrm{sgn}(v_t) \text{ and } |v_{t-1}| - |v_t| > 3 \\ 0 & \text{otherwise.} \end{cases}$$

Finally, the proximity principle favors small realized intervals. We define

$$\mathrm{pr}_t = \begin{cases} 0 & \text{if } |v_t| \geq 6 \\ 1 & \text{if } 3 \leq |v_t| \leq 5 \\ 2 & \text{if } 0 \leq |v_t| \leq 2 \end{cases} .$$

We define this feature with less possible states than in (Schellenberg 1997) in order to limit the dimensionality of the Narmour representation. Besides, the actual numerical values for each of the Narmour features do not correspond to those of Schellenberg (1997), where the goal was to quantify numerically the subjective melodic expectation. In the context of this paper, these values only correspond to discrete ordered values summarizing triplets of notes.

From these definitions, the Narmour features for the note triplet (u_{t-2}, u_{t-1}, u_t) are defined as

$$\gamma_t = (\mathrm{rm}_t, \mathrm{id}_t, \mathrm{rr}_t, \mathrm{cl}_t, \mathrm{pr}_t) \ .$$

Such features have 108 possible different discrete states.

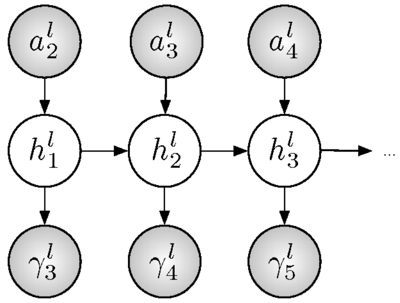

Figure 6. Variant of an IOHMM model for Narmour features given note lengths. The variables in level 1 are always observed and correspond to *previous* note lengths. Variables in level 2 are hidden, while variables in level 3 correspond to Narmour features. All variables in grey are observed during training.

As an example, the sequence of MIDI notes $(u_1, u_2, u_3, u_4) = (71, 74, 72, 84)$ would lead to the Narmour features $\gamma_3 = (1, 1, 1, 1, 2)$ and $\gamma_4 = (1, 0, 0, 1, 0)$.

4.3. *Melodic Model*

In this section, we describe a probabilistic model for melodies given rhythms and chord progressions. While the IOHMM in Section 4.1 was directly modeling the choice of notes given chords (and implicitly rhythms), the model described here proceeds in two steps. We first model sequences of Narmour features given rhythm. Then, we model the actual choice of melodic notes, given sequences of Narmour features generated in the last step and chord progressions. These two steps correspond to steps 2. and 3. in Figure 4.

4.3.1. IOHMM for Narmour Features

An IOHMM like the one presented in Section 4.1 can be used to model sequences of Narmour features given rhythms. We first compress the rhythm dataset in a form that is synchronized with Narmour features: we define $\mathbf{a}^l = (a_2^l, \ldots, a_{g_l-1}^l)$ to be the l-th sequence of note lengths in the rhythm dataset $\mathcal{X}$, ignoring the first and last note lengths a_1^l and $a_{g_l}^l$. When considering the rhythm representation defined in Section 2.3, each a_i^l is equal to one plus the number of symbols 2 following the i-th symbol 1 in the corresponding rhythm sequence $\mathbf{x}^l$. For instance, the rhythm sequence $\mathbf{x}^l = (1, 1, 2, 2, 3, 1, 2, 1)$ produces the note length sequence $\mathbf{a}^l = (3, 2)$. We denote by $\boldsymbol{\gamma}^l = (\gamma_3^l, \ldots, \gamma_{g_l}^l)$ the sequence of Narmour features associated to the l-th melody. This sequence starts with index 3 because each Narmour feature spans three notes.

The joint probability of each sequence of Narmour features $\boldsymbol{\gamma}^l$, its associated sequence of note lengths $\boldsymbol{a}^l$, and hidden states $\mathbf{h}^l$ can be modeled by

$$p_{\text{NIOHMM}}(\mathbf{a}^l, \boldsymbol{\gamma}^l, \mathbf{h}^l) = p_i(a_2^l) p_\pi(h_1^l | a_2^l) p_o(\gamma_3^l | h_1^l) \prod_{t=4}^{g_l} p_i(a_{t-1}^l) p_{\bar{o}}(h_{t-2}^l | h_{t-3}^l, a_{t-1}^l) p_o(\gamma_t^l | h_{t-2}^l) \ . \tag{13}$$

This model is shown in Figure 6. As in Equation (12), the probability distributions p_π, p_i, $p_{\bar{o}}$, and p_o are multinomials, and the model is learned by the standard EM algorithm.

As can be seen in Equation (13), we arbitrarily chose to condition the Narmour features on the *previous* note length. This is due to the empirical observation that greater intervals tend to occur after long notes while smaller intervals tend to occur after short notes. Other models of Narmour features given *current* length, a longer past context, or even no note length at all could be considered. We leave this exploration for future work.

4.3.2. Notes Model

We are now about to reach the end of our quest for a complete generative model of melodies given chord progressions and rhythms. The only piece of the puzzle that remains to be defined is a model for MIDI notes given Narmour features, chord progressions, and rhythms.

As stated in Section 1, each chord is made of three or more notes. Usually, the lowest note of the chord is called the *root* of the chord. This note is so important in the chord that it gives its name to the chord. For instance, the root of the chord CMaj7b5 is the pitch class C. Here we decompose the chord representation defined in Section 4.1 into two parts: $\nu_i^l = (\eta_i^l, \tau_i^l)$, where η_i^l is the structure of the chord and τ_i^l is the root pitch class. Chord structures are just the chord definitions aside of the name of the root (e.g. "m7b5" is the chord structure in the chord "Bm7b5"). Each different chord structure is mapped to a specific state of the variables η_i^l. The sequences $\boldsymbol{\eta}^l = (\eta_1^l, \ldots, \eta_{g_l}^l)$ and $\boldsymbol{\tau}^l = (\tau_1^l, \ldots, \tau_{g_l}^l)$ are respectively the *chord structure* and the *root* progressions of the l-th song in the dataset.

Let $\tilde{u}_t^l$ be an arbitrary MIDI note played at time t. We define

$$\phi(\tilde{u}_t^l, \tau_t^l) = ((\tilde{u}_t^l \mod 12) - \tau_t^l) \mod 12$$

to be the representation of the pitch class associated to the MIDI note $\tilde{u}_t^l$, *relative* to the root of the current chord. For instance, let $\tilde{u}_t^l = 65$ (note F) be played over the D minor chord. In that case, we have $\tau_t^l = 2$, meaning that the pitch class of the root of the chord is D. Hence, $\phi(65, 2) = 3$ for that particular example, meaning that the current melody note pitch class is 3 semi-tones higher than the root of the current chord.

It is easy to estimate $p(\eta_t^l | \tilde{u}_t^l, \tau_t^l)$ with a multinomial distribution conditioned on the values of $\phi(\tilde{u}_t^l, \tau_t^l)$. This distribution can be estimated by maximum likelihood over a training set. Hence, we learn a simple distribution of the chord structures η for each possible pitch classes of the melodies relative to the roots of the corresponding chords. For instance, this distribution could learn the fact that we often observe a minor seventh chord when playing a minor third over the tonic in the melody.

Let $\tilde{\gamma}_t^l(u_{t-2}^l, u_{t-1}^l, \tilde{u}_t^l)$ be the extracted Narmour feature when notes u_{t-2}^l and u_{t-1}^l are followed by the arbitrary note $\tilde{u}_t^l$. Also, let $\tilde{\kappa}_t^l$ be an arbitrary random variable such that

$$p(\tilde{\kappa}_t^l = 1 | \tilde{u}_t^l, u_{t-2}^l, u_{t-1}^l, \gamma_t^l) = \begin{cases} 1 & \text{if } \gamma_t^l = \tilde{\gamma}_t^l(u_{t-2}^l, u_{t-1}^l, \tilde{u}_t^l) \\ 0 & \text{otherwise.} \end{cases}$$

In words, $\tilde{\kappa}_t^l$ is equal to 1 if and only if the Narmour feature produced when playing arbitrary note $\tilde{u}_t^l$ is equal to the *given* Narmour feature γ_t^l, given the two previous notes.

We define a factorization of the joint probability of the variables $\tilde{u}_t^l, u_{t-1}^l, u_{t-2}^l, \eta_t^l, \tau_t^l, \boldsymbol{\gamma}_t^l$, and $\tilde{\kappa}_t^l$ with

$$\begin{aligned} &p(\tilde{u}_t^l, u_{t-1}^l, u_{t-2}^l, \eta_t^l, \tau_t^l, \boldsymbol{\gamma}_t^l, \tilde{\kappa}_t^l) = \\ &\quad p(u_{t-1}^l)p(u_{t-2}^l)p(\boldsymbol{\gamma}_t^l)p(\tilde{\kappa}_t^l | \tilde{u}_t^l, u_{t-2}^l, u_{t-1}^l, \gamma_t^l)p(\tilde{u}_t^l)p(\tau_t^l)p(\eta_t^l | \tilde{u}_t^l, \tau_t^l) \end{aligned} \tag{14}$$

at each time t. This factorization is is shown by the graphical model in Figure 7.

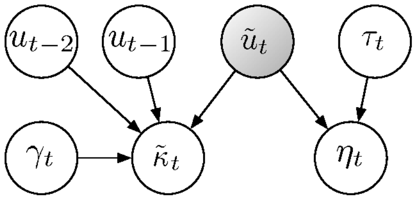

Figure 7. Graphical model representation of the factorization of the joint probability defined in Eq. (14).

We want to estimate the probability of playing any arbitrary MIDI note $\tilde{u}_t^l$ at time t in the l-th song of the dataset given the two previous observed notes u_{t-2}^l and u_{t-1}^l, the current Narmour feature $\boldsymbol{\gamma}_t^l$, and the current chord $\boldsymbol{\nu}_t^l = (\eta_t^l, \tau_t^l)$. Given the factorization in Equation (14), we have that

$$p_{MEL}(\tilde{u}_t^l | u_{t-1}^l, u_{t-2}^l, \eta_t^l, \tau_t^l, \gamma_t^l, \tilde{\kappa}_t^l = 1) = \frac{p(\tilde{\kappa}_t^l = 1 | \tilde{u}_t^l, u_{t-2}^l, u_{t-1}^l, \gamma_t^l) p(\tilde{u}_t^l) p(\eta_t^l | \tilde{u}_t^l, \tau_t^l)}{\sum_{\tilde{u}_t^l} p(\tilde{\kappa}_t^l = 1 | \tilde{u}_t^l, u_{t-2}^l, u_{t-1}^l, \gamma_t^l) p(\tilde{u}_t^l) p(\eta_t^l | \tilde{u}_t^l, \tau_t^l)} \quad (15)$$

where $p(\tilde{u}_t^l)$ is the prior probability of observing $\tilde{u}_t^l$. The distribution $p(\tilde{u}_t^l)$ is a multinomial that can be simply estimated by maximum likelihood on the training set.

Hence, a simple strategy to find the most likely MIDI note $\tilde{u}_t^l$ given u_{t-1}^l, $u_{t-2}^l, \eta_t^l, \tau_t^l$, and $\boldsymbol{\gamma}^l$ is to solve

$$\arg \max_{\{\tilde{u}_t^l | \tilde{\kappa}_t^l = 1, u_{t-1}^l, u_{t-2}^l, \boldsymbol{\gamma}^l\}} p(\tilde{u}_t^l) p(\eta_t^l | \tilde{u}_t^l, \tau_t^l) \ ,$$

since the denominator in the right-hand side of Equation (15) is the same for all values of $\tilde{u}_t^l$. In other words, we search for the most likely melodic note (with respect to the current chord) among all the possible notes given the current Narmour constraint and the current chord. Despite the fact that this model only predicts one note at a time, it is able to take into account longer term melodic shapes through the constraints imposed by the sequences of Narmour features.

Melodic prediction *without* observing Narmour features can be done with this model in two steps. We first generate the most likely sequence of Narmour features given rhythms with the IOHMM model described in Section 4.3.1. Then, we can use the melodic prediction model described in the current section to predict MIDI notes given chord progressions. Such a model is shown in Section 5 to have much better prediction accuracy than using a simpler IOHMM model alone.

Unsupervised probabilistic models can be sampled to generate genuine chord progressions (Paiement et al. 2005). The melodic model described here is able to generate realistic melodies given these chord progressions and beginning of melodies. This system can be used as a tool to ease music composition. Audio files generated by sampling the different models presented in this paper are available at `http://www.idiap.ch/~paiement/connection`. Even for the non musician, it should be obvious that the sequences generated by sampling the melodic model introduced in this section are much more realistic than sequences generated by sampling the IOHMM model described in Section 4.1. Both models generate notes that are coherent with the current chord. However, the sequences generated by the IOHMM model do not have any coherent structure. On the other hand, melodies generated by the melodic model presented here tend to follow the same melodic

Table 4. Local accuracy (the higher the better) for prediction models on the jazz standards database, for various prediction starting points s.

s	IOHMM	Narmour
32	2.0%	11.9%
64	1.7%	12.1%
96	2.2%	14.9%

shapes than the songs in the training sets. These melodic shapes are constrained by the conditioning sequences of Narmour features used as inputs.

5. Melodic Prediction Experiments

To compare the melodic model described in the previous section with the IOHMM model of Section 4.1, we propose a slightly different evaluation criterion than the prediction accuracy defined in Section 3. This alternate criterion was chosen in this case for its computational simplicity.

The goal of the proposed models is to predict or generate melodies *given* chord progressions and rhythms. Let $\mathbf{u}^j = (u_1^j, \dots, u_{g_j}^j)$ be a test sequence of MIDI notes and $\hat{u}_i^j$ to be the output of the evaluated prediction model on the i-th position when given $(u_1^j, \dots, u_{i-1}^j)$ and the associated rhythm sequence $\mathbf{x}^j$. Assume that the dataset is divided into K folds $T_1, \dots, T_K$ (each containing different sequences), and that the k-th fold T_k contains n_k test sequences. Again, let s be the first rhythm position from which the evaluated model try to guess what would be the next notes in each test songs. When using cross-validation, we define the "local accuracy" *LocAcc* of an evaluated model to be

$$LocAcc = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{n_k} \sum_{j \in T_k} \frac{1}{g_j - \zeta_s^j + 1} \sum_{i=\zeta_s^j}^{g_j} \tilde{\varepsilon}_i^j \tag{16}$$

where $\tilde{\varepsilon}_i^j = 1$ if $\hat{u}_i^j = u_i^j$, and 0 otherwise, and ζ_s^j is the smallest note index in the j-th song whose onset position is greater than position s. Hence, prediction models have access to *all* the previously observed notes when trying to guess what would be the next note. This rate of success is averaged on the last $g_j - \zeta_s^j + 1$ notes of each song.

We tried 2 to 20 possible hidden states in the reported experiments for the baseline IOHMM model of Section 4.1 and the "Narmour" IOHMM of Section 4.3.1. Both models try to predict out-of-sample melody notes, given chord progressions and complete test rhythm sequences $\mathbf{x}^j$. The same chord representations are used as input for both models. We used 5-fold cross-validation to compute prediction accuracies. We report results for the choices of parameters that provided the highest accuracies for each model. The IOHMM model of notes given chords is a stronger contender than would be a simpler HMM trained on melodies, because the prediction given by the IOHMM takes advantage of the current input.

Results in Table 4 for the jazz standards database show that generating Narmour features as an intermediate step greatly improves prediction accuracy. Since there are 128 different MIDI notes, a completely random predictor would have a local accuracy of 0.8%. Both models take into account chord progressions when trying to predict the next MIDI note. However, the Narmour model favors melodic shapes similar to the ones found in the training set. The Narmour model still provides

Table 5. Local accuracy (the higher the better) for prediction models on the hornpipes database, for various prediction starting points s.

s	IOHMM	Narmour
48	2.5%	4.6%
96	2.6%	4.9%
144	2.6%	5.0%

consistently better prediction accuracy on the hornpipes database, as can be seen in Table 5. However, prediction accuracies are lower on the hornpipes database than on the jazz database for the Narmour model. Note onsets (symbol 1) occur on most rhythm positions in this database. This means that rhythm sequences in this database have relatively low entropy. Hence, rhythm sequences are less informative when used as conditioning inputs to generate sequences of Narmour features. Another observation is that the chord structures in this database are almost always the same (i.e. simple triads). The melodic model of Section 4.3 is directly modeling the distribution $p(\eta_t^l|\tilde{u}_t^l, \tau_t^l)$ of chord structures given relative MIDI notes. This distribution was probably more helpful for melodic prediction in the jazz database than in the hornpipes database. Despite these two drawbacks, the melodic model of Section 4.3 has a prediction accuracy twice as good as what was obtained with the simpler IOHMM model in the hornpipes database.

Again, while the prediction accuracy is simple to compute and to apprehend, other performance criteria, such as ratings provided by a panel of experts, should be more appropriate to evaluate the relevance of music models. The fact that the Narmour model accurately predict "only" about 12% of the notes on out-of-sample sequences does not mean that it is not performing well when generating the other "wrong" notes. Many realistic melodies can be generated on the same chord progression in a given musical genre. Moreover, some mistakes are more harmful than others. For most applications, a model that would have very low prediction accuracy, but that would generate realistic melodies, would be preferable to a model with 50% prediction accuracy, but that would generate unrealistic notes the other half of the time. In principle, likelihood of out-of-sample sequences could have been an interesting alternative evaluation measure to the reported prediction accuracies that would somehow alleviate this problem. Indeed, out-of-sample likelihood measures the fit of a whole distribution to out-of-sample data, whereas prediction error rate concentrates on the fit of the modes of the distribution. However, out-of-sample likelihood has three major drawbacks that prevented us to use it as an evaluation measure in this paper. First, each factor in the likelihood can be potentially arbitrary close to 0 when a very unlikely event is encountered during testing. This means that a single unlikely event can have a dramatic impact on the evaluation of the model. Prediction error rate does not suffer from this problem because all errors have the same cost, which makes this evaluation method much more robust to outliers. Secondly, our objective was not to sample the proposed models, but to predict the most probable ending of a song given its beginning. Prediction error rate is an evaluation measure that is directly related to this particular application. Last but not least, exact computation of the likelihood is not tractable in our proposed rhythm model.

6. Conclusion

The main contribution of this paper is the design and evaluation of predictive models for rhythms and melodies. While a few models have already been proposed for music in general (Dubnov et al. 2003, Pachet 2003), we are not aware of proper quantitative comparisons between predictive models of music, as we do in Sections 3 and 5.

In Section 2, we considered rhythm modeling as a first step towards full melodic modeling. For doing so, we proposed a generative model for distance patterns in temporal data. The model is specifically well-suited to music data, which exhibits strong regularities in dyadic distance patterns between subsequences. Reported conditional prediction accuracies in Section 3 show that such regularities are present in music data and can be effectively captured by the proposed model. Moreover, learning distributions of distances between subsequences really helps for accurate rhythm prediction.

Finally, with a reliable rhythm model available, we introduced in Section 4 a generative model of melodies *given* rhythms and chord progressions. For that purpose, we first described melodic features derived from Narmour (1990) that put useful constraints on melodies based on musicological substantiation. We then defined in Section 4.3 a probabilistic model of melodies that provides significantly higher prediction rates than a simpler, yet powerful, Markovian model. The combination of the rhythm model and the melodic model given chords leads to a predictive model of music that could be interesting in many applications. Furthermore, these models are able to generate realistic melodies given appropriate musical contexts.

Acknowledgments

This work was supported in part by the IST Program of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, funded in part by the Swiss Federal Office for Education and Science (OFES) and the Swiss NSF through the NCCR on IM2.

References

Allan, M., and Williams, C.K.I. (2004), "Harmonising Chorales by Probabilistic Inference," in *Advances in Neural Information Processing Systems*, Vol. 17.

Bengio, Y., and Frasconi, P. (1996), "Input/Output HMMs for Sequence Processing," *IEEE Transactions on Neural Networks*, 7(5), 1231–1249.

Bengio, Y., Simard, P., and Frasconi, P. (1994), "Learning Long-Term Dependencies with Gradient Descent is Difficult," *IEEE Transactions on Neural Networks*, 5(2), 157–166.

Bilmes, J., "A Gentle Tutorial on the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models," (1997).

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

Dixon, S. (2007), "Evaluation of the Audio Beat Tracking System BeatRoot," *Journal of New Music Research*, 36(1), 39–50.

Dubnov, S., Assayag, G., Lartillot, O., and Bejerano, G. (2003), "Using Machine-Learning Methods for Musical Style Modeling," *IEEE Computer*, 10(38).

Duda, R.O., Hart, P.E., and Stork, D.G., *Pattern Classification, Second Edition*, Wiley Interscience (2000).

Eck, D., and Schmidhuber, J. (2002), "Finding Temporal Structure in Music: Blues Improvisation with LSTM Recurrent Networks," in *Neural Networks for Signal Processing XII, Proc. 2002 IEEE Workshop*, ed. H. Bourlard, New York: IEEE, pp. 747–756.

Grachten, M., Arcos, J.L., and de Mantaras, R.L. (2005), "Melody Retrieval using the Implication/Realization Model," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*.

Handel, S., *Listening: An introduction to the perception of auditory events*, Cambridge, Mass.: MIT Press (1993).

Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983), "Optimization by Simulated Annealing," *Science, Number 4598, 13 May 1983*, 220, 4598, 671–680.

Lauritzen, S.L., *Graphical Models*, Oxford University Press (1996).

Lee, K., and Slaney, M. (2007), "A Unified System for Chord Transcription and Key Extraction Using Hidden Markov Models," in *Proceedings of International Conference on Music Information Retrieval (ISMIR)*.

Narmour, E., *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*, Chicago: Univeristy of Chicago Press (1990).

Narmour, E., *The Analysis and Cognition of Melodic Complexity: The Implication-Realization Model*, University of Chicago Press (1992).

Pachet, F. (2003), "The Continuator: Musical Interaction With Style," *Journal of New Music Research*, 32(3), 333–341.

Paiement, J.F., Eck, D., Bengio, S., and Barber, D. (2005), "A Graphical Model for Chord Progressions Embedded in a Psychoacoustic Space," in *Proceedings of the 22nd International Conference on Machine Learning*.

Paiement, J.F., Eck, D., and Bengio, S. (2006), "Probabilistic Melodic Harmonization," in *Proceedings of the 19th Canadian Conference on Artificial Intelligence*, Springer, pp. 218–229.

Rabiner, L.R. (1989), "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, 77(2), 257–285.

Rohde, D.L.T. (2002), "Methods for binary multidimensional scaling," *Neural Comput.*, 14(5), 1195–1232.

Scheirer, E. (1998), "Tempo and beat analysis of acoustic musical signals," *Journal of the Acoustical Society of America*, 103(1), 588–601.

Schellenberg, E. (1997), "Simplifying the Implication-Realization Model of Musical Expectancy," *Music Perception*, 14(3), 295–318.

Sher, C. (ed.) *The New Real Book*, Vol. 1-3, Sher Music Co. (1988).

Titterington, D.M., Smith, A.F.M., and Makov, U.E., *Statistical Analysis of Finite Mixture Distributions*, Wiley (1985).