LEARNING TO ALIGN POLYPHONIC MUSIC

Shai Shalev-Shwartz Joseph Keshet Yoram Singer
{shais,jkeshet,singer}@cs.huji.ac.il
School of Computer Science and Engineering,
The Hebrew University, Jerusalem, 91904, Israel

ABSTRACT

We describe an efficient learning algorithm for aligning a symbolic representation of a musical piece with its acoustic counterpart. Our method employs a supervised learning approach by using a training set of aligned symbolic and acoustic representations. The alignment function we devise is based on mapping the input acoustic-symbolic representation along with the target alignment into an abstract vector-space. Building on techniques used for learning support vector machines (SVM), our alignment function distills to a classifier in the abstract vector-space which separates correct alignments from incorrect ones. We describe a simple iterative algorithm for learning the alignment function and discuss its formal properties. We use our method for aligning MIDI and MP3 representations of polyphonic recordings of piano music. We also compare our discriminative approach to a generative method based on a generalization of hidden Markov models. In all of our experiments, the discriminative method outperforms the HMM-based method.

1. INTRODUCTION

There are numerous ways to represent musical recordings. Typically, a representation is either symbolic (e.g. a musical score or MIDI events) or a digitized audio form such as PCM. Symbolic representations entertain quite a few advantages which become handy in applications such as content-based retrieval. However, performances of musical pieces are typically recorded in one of the common forms for coding of audio signals. Score alignment is the task of associating each symbolic event with its actual time of occurrence in the observed audio signal.

There are several approaches to the alignment problem (see for instance [14, 16] and the references therein). Most of the previous work on alignment has focused on *generative* models and employed parameter estimation techniques in order to find a model that fits the data well. In this paper we propose an alternative approach for learning alignment functions that builds on recent work on *discriminative supervised* learning algorithms. The advantage of discriminative learning algorithms stems from the fact that the objective function used during the learning phase is tightly coupled with the decision task one needs to perform. In addition, there is both theoretical and empirical evidence that discriminative learning algorithms are likely to outperform generative models for the same task (cf. [4, 19]). To facilitate supervised learning, we need to have access to a training set of aligned data, consisting of symbolic representations along with the division of the performance into the actual start times of notes.

There are numerous applications where an accurate and fast alignment procedure may become handy. Soulez et al. [16] describe few applications of score alignment such as content-based retrieval and comparisons of different performances of the same musical piece. In addition, the ability to align between symbolic and acoustic representations is an essential first step toward a polyphonic note detection system (see also [18, 20, 9]). The goal of a polyphonic note detection system is to spot notes in an audio signal. This detection task is rather difficult if numerous notes are played simultaneously (e.g. in polyphonic pieces). There exist theoretical and empirical evidences that supervised learning is effective also for complex decision problems and is thus likely to be adequate for polyphonic note detection. However, supervised learning algorithms rely on the existence of labeled examples. Fortunately, the abundance of large acoustic and symbolic databases together with an efficient alignment procedure enables the construction of training set for the polyphonic note detection problem.

Related work Music to score alignment is an important research topic and has many applications. Most of the previous work has focused on monophonic signals. See for example [13, 5, 7] and the references therein. Several recent works [16, 14] deal with more complex polyphonic signals. In this paper, we suggest to automatically learn an alignment function from examples using a discriminative learning setting. Our learning algorithm builds upon recent advances in kernel machines and large margin classifiers for sequences [2, 1, 17] which in turn build on the pioneering work of Vapnik and colleagues [19, 4]. The specific form of the learning algorithm described in Sec. 3 stems from recent work on online algorithms [8, 3].

2. PROBLEM SETTING

In this section we formally describe the alignment problem. We denote scalars using lower case Latin letters (e.g. x), and vectors using bold face letters (e.g. $\mathbf{x}$). A sequence of elements is designated by a bar ($\bar{\mathbf{x}}$) and its length is denoted by $|\bar{\mathbf{x}}|$.

In the alignment problem, we are given a digitized audio signal of a musical piece along with a symbolic representation of the same musical piece. Our goal is to generate an alignment between the signal and the symbolic representation. The audio signal is first divided into fixed length frames (we use 20ms in our experiments) and a d dimensional feature vector is extracted from each frame of the audio signal. For brevity we denote the domain of the feature vectors by $\mathcal{X} \subset \mathbb{R}^d$. The feature representation of an audio signal is therefore a sequence of feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathcal{X}$ for all $1 \le t \le T$. A symbolic representation of a musical piece is formally defined as a sequence of events which represent a standard way to perform the musical piece. There exist numerous symbolic representations. For simplicity and concreteness we focus on events of type "note-on". Formally, each "note-on" event is a pair (p, s). The first element of the pair, $p \in \mathcal{P} = \{0, 1, \ldots, 127\}$ is the note's pitch value (coded using the MIDI standard). The second element, s is assumed to be a positive integer ($s \in \mathbb{N}$) as it measures the start time of the note in a predefined discrete units (we use 20ms in our experiments). Therefore, a symbolic representation of a musical piece consists of a sequence of pitch values $\bar{p} = (p_1, \ldots, p_k)$ and a corresponding sequence of start-times $\bar{s} = (s_1, \ldots, s_k)$. Note that the number of notes clearly varies from one musical piece to another and thus k is not fixed. We denote by $\mathcal{P}^\star$ (and similarly $\mathbb{N}^\star$ and $\mathcal{X}^\star$) the set of all finite-length sequences over $\mathcal{P}$. In summary, an alignment instance is a triplet $(\bar{\mathbf{x}}, \bar{p}, \bar{s})$ where $\bar{\mathbf{x}}$ is an acoustic representation of the musical piece and $(\bar{p}, \bar{s})$ is a symbolic representation of the piece. The domain of alignment instances is denoted by $\mathcal{Z} = \mathcal{X}^\star \times (\mathcal{P} \times \mathbb{N})^\star$. An alignment between the acoustic and symbolic representations of a musical piece is formally defined as a sequence of *actual* start-times $\bar{y} = (y_1, \ldots, y_k)$ where $y_i \in \mathbb{N}$ is the observed start-time of note i in the acoustic signal.

Clearly, there are different ways to perform the same musical score. Therefore, the actual (or observed) start times of the notes in the perceived audio signal are very likely to be different from the symbolic start-times. Our goal is to learn an alignment function that predicts the observed start-times from the audio signal and the symbolic representation, $f : \mathcal{Z} \to \mathbb{N}^\star$. To facilitate an efficient algorithm we confine ourselves to a restricted class of alignment functions. Specifically, we assume the existence of a predefined set of alignment features, $\{\phi_j\}_{j=1}^n$. Each alignment feature is a function of the form $\phi_j : \mathcal{Z} \times \mathbb{N}^\star \to \mathbb{R}$. That is, each alignment feature gets acoustic and symbolic representations of a musical piece $\mathbf{z} = (\bar{\mathbf{x}}, \bar{p}, \bar{s})$, together with a candidate alignment $\bar{y}$, and returns a scalar which, intuitively, represents the confidence in the suggested alignment $\bar{y}$. We denote by $\boldsymbol{\phi}(\mathbf{z}, \bar{y})$ the vector in $\mathbb{R}^n$ whose jth element is $\phi_j(\mathbf{z}, \bar{y})$. The alignment functions we use are of the form

$$f(\mathbf{z}) = \operatorname*{argmax}_{\bar{y}} \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}) \ , \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of importance weight that we need to learn. In words, f returns a suggestion for an alignment sequence by maximizing a weighted sum of the scores returned by each feature function ϕ_j. Note that the number of possible alignment sequences is exponentially large. Nevertheless, as we show below, under mild conditions on the form of the feature functions ϕ_j, the optimization in Eq. (1) can be efficiently calculated using a dynamic programming procedure.

As mentioned above, we would like to learn the function f from examples. Each example containing an alignment is composed of an acoustic and a symbolic representation of a musical piece $\mathbf{z} = (\bar{\mathbf{x}}, \bar{p}, \bar{s}) \in \mathcal{Z}$ together with the true alignment between them, $\bar{y}$. Let $\bar{y}' = f(\mathbf{z})$ be the alignment suggested by f. We denote by $\gamma(\bar{y}, \bar{y}')$ the cost of predicting the alignment $\bar{y}'$ where the true alignment is $\bar{y}$. Formally, $\gamma : \mathbb{N}^\star \times \mathbb{N}^\star \to \mathbb{R}$ is a function that gets two alignments and returns a scalar which is the cost to predict the second input alignment where the true alignment is the first. We assume that $\gamma(\bar{y}, \bar{y}') \ge 0$ and that $\gamma(\bar{y}, \bar{y}) = 0$. An example for a cost function is,

$$\gamma(\bar{y}, \bar{y}') = \frac{1}{|\bar{y}|} \sum_{i=1}^{|\bar{y}|} |y_i - y'_i| \ .$$

In words, the above cost is the average of the absolute difference between the predicted alignment and the true alignment. In our experiments, we used a variant of the above cost function and replaced the summands $|y_i - y'_i|$ with $\max\{0, |y_i - y'_i| - \varepsilon\}$, where ε is a predefined small constant. The advantage of this cost is that no loss is incurred due to the ith note if y_i and y'_i are within a distance of ε of each other. The goal of the learning process is to find an alignment function f that attains small cost on unseen examples. Formally, let Q be any (unknown) distribution over the domain of alignment examples, $\mathcal{Z} \times \mathbb{N}^\star$. The goal of the learning process is to minimize the risk of using the alignment function, defined as the expected error of f on alignment examples, where the expectation is taken with respect to the distribution Q,

$$\text{risk}(f) = \mathbb{E}_{(\mathbf{z}, \bar{y}) \sim Q} \left[\gamma(\bar{y}, f(\mathbf{z})) \right] \ .$$

To do so, we assume that we have a training set of alignment examples each of which is identically and independently distributed (i.i.d.) according to the distribution Q. (Note that we only observe the training examples but we do not know the distribution Q.) In the next section we show how to use the training set in order to find an alignment function f which achieves small cost on the training set and that with high probability, achieves small average cost on unseen examples as well.

The paper is organized as follows. In Sec. 3 we describe an efficient algorithm that learns an alignment function f from examples. The learning algorithm assumes that f is as described in Eq. (1). A specific set of acoustic features and feature functions is discussed in Sec. 4. In Sec. 5 we describe a dynamic programming procedure that efficiently calculates f. In Sec. 6 we describe an alternative method for alignment which is based on a generative model. In Sec. 7 we report experiments on alignment of polyphonic piano musical pieces and compare our method to the generative method. Finally, some future directions are discussed in Sec. 8.

3. DISCRIMINATIVE LEARNING ALGORITHM

Recall that a supervised learning algorithm for alignment receives as input a training set $S = \{(\mathbf{z}_1, \bar{y}_1), \ldots, (\mathbf{z}_m, \bar{y}_m)\}$ and returns a weight vector $\mathbf{w}$ defining an alignment function f given in Eq. (1). In the following we present an iterative algorithm for learning the weight vector $\mathbf{w}$. We denote by $\mathbf{w}_t$ the weight vector after iteration t of the algorithm. We start with the zero vector $\mathbf{w}_0 = \mathbf{0}$. On iteration t of the algorithm, we first receive a triplet $\mathbf{z} = (\bar{\mathbf{x}}, \bar{p}, \bar{s})$ representing the acoustic and symbolic representations of one of the musical pieces from our training set. Next, we use the current weight vector $\mathbf{w}_t$ for predicting the alignment between $\bar{\mathbf{x}}$ and $(\bar{p}, \bar{s})$ as in Eq. (1). Let $\bar{y}'_t$ be the predicted alignment. We then receive the true alignment $\bar{y}$ from the training set and suffer cost $\gamma(\bar{y}, \bar{y}'_t)$. If the cost is zero we continue to the next iteration and keep $\mathbf{w}_t$ intact, hence $\mathbf{w}_{t+1} = \mathbf{w}_t$. Otherwise, we update the weight vector to be

$$\mathbf{w}_{t+1} \;=\; \mathbf{w}_t + \frac{\sqrt{\gamma(\bar{y}, \bar{y}'_t)} - \mathbf{w}_t \cdot \mathbf{a}_t}{\|\mathbf{a}_t\|^2}\, \mathbf{a}_t \;\;, \qquad (2)$$

where $\mathbf{a}_t = \boldsymbol{\phi}(\mathbf{z}, \bar{y}) - \boldsymbol{\phi}(\mathbf{z}, \bar{y}'_t)$. In words, we add to $\mathbf{w}_t$ a vector which is a scaled version of the difference between the alignment feature vector resulting from the true alignment $\boldsymbol{\phi}(\mathbf{z}, \bar{y})$ and the one obtained by the alignment function $\boldsymbol{\phi}(\mathbf{z}, \bar{y}'_t)$. It is simple to show that $\mathbf{w}_{t+1}$ is the minimizer of the following projection problem

$$\min_{\mathbf{w}} \|\mathbf{w} - \mathbf{w}_t\|^2 \qquad \text{s.t.} \qquad (3)$$
$$\mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}) \;\;\geq\;\; \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}'_t) + \sqrt{\gamma(\bar{y}, \bar{y}'_t)}$$

Therefore, after updating $\mathbf{w}$, the score of the true alignment $\bar{y}$ is larger than the score of the predicted alignment $\bar{y}'_t$ by at least $\sqrt{\gamma(\bar{y}, \bar{y}'_t)}$. Moreover, among all weight vectors $\mathbf{w}$ that satisfy the inequality in Eq. (3), $\mathbf{w}_{t+1}$ is closest to the vector $\mathbf{w}_t$. After each update of $\mathbf{w}$, we find the largest alignment error on the training set

$$\epsilon = \max\{\gamma(\bar{y}, f(\mathbf{z})) : (\mathbf{z}, \bar{y}) \in S\} \;\; .$$

If ϵ is lower than a termination parameter, denoted ϵ_0, we stop and return the last value of $\mathbf{w}$. A pseudo-code of the learning algorithm is given in Fig. 1.

Input: A training set $S = \{(\mathbf{z}_1, \bar{y}_1), \ldots, (\mathbf{z}_m, \bar{y}_m)\}$; accuracy parameter ϵ_0

Initialize: Set $\mathbf{w} = \mathbf{0}$;
$(\mathbf{z}, \bar{y}) = \arg\max\{\gamma(\bar{y}, f(\mathbf{z})) : (\mathbf{z}, \bar{y}) \in S\}$;
$\epsilon = \gamma(\bar{y}, f(\mathbf{z}))$

While $\epsilon \geq \epsilon_0$ **do**:

Predict: $\bar{y}' = f(\mathbf{z}) = \operatorname*{argmax}_{\bar{y}} \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y})$

Pay Cost: $\gamma(\bar{y}, \bar{y}')$

Set: $\mathbf{a} = \boldsymbol{\phi}(\mathbf{z}_i, \bar{y}_i) - \boldsymbol{\phi}(\mathbf{z}_i, \bar{y}')$

Update: $\mathbf{w} \leftarrow \mathbf{w} + \dfrac{\sqrt{\gamma(\bar{y}_i, \bar{y}')} - \mathbf{w} \cdot \mathbf{a}}{\|\mathbf{a}\|^2}\, \mathbf{a}$

Choose next example:
$(\mathbf{z}, \bar{y}) = \arg\max\{\gamma(\bar{y}, f(\mathbf{z})) : (\mathbf{z}, \bar{y}) \in S\}$;
$\epsilon = \gamma(\bar{y}, f(\mathbf{z}))$

Output: Final weight vector $\mathbf{w}$

Figure 1. The alignment learning algorithm.

The following theorem bounds the number of iterations performed by the above learning algorithm. Our analysis assumes that there exists a weight vector $\mathbf{w}^\star \in \mathbb{R}^n$ such that the following inequality holds for all examples in the training set $(\mathbf{z}, \bar{y}) \in S$ and for all $\bar{y}'$

$$\mathbf{w}^\star \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}) \geq \mathbf{w}^\star \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}') + \sqrt{\gamma(\bar{y}, \bar{y}')} \;\; . \qquad (4)$$

Note that if we use $\mathbf{w}^\star$ in Eq. (1) then $\gamma(\bar{y}, f(\mathbf{z})) = 0$ for all the examples in the training set. A modification of the algorithm to the case where such vector does not exist can be derived using a similar technique to the one described in [3].

Theorem 1. *Let $S = \{(\mathbf{z}_1, \bar{y}_1), \ldots, (\mathbf{z}_m, \bar{y}_m)\}$ be a set of training examples. Assume that there exists a weight vector $\mathbf{w}^\star \in \mathbb{R}^n$ such that Eq. (4) holds for all $(\mathbf{z}_t, \bar{y}_t)$ and $\bar{y}'$. In addition, assume that for all t and for all $\bar{y}'$ we have $\|\boldsymbol{\phi}(\mathbf{z}_t, \bar{y}')\| \leq 1/2$. Let f be the alignment function obtained by running the algorithm from Fig. 1 on S with accuracy parameter ϵ_o. Then the total number of iterations of the algorithm is at most $\|\mathbf{w}^\star\|^2/\epsilon_0$.*

The proof of the theorem is provided in a long version of the paper [15]. Thm. 1 states that the number of iterations of the algorithm does not depend on the number of examples. Therefore, only a small part of the examples in the training set actually effects the resulting alignment function. Intuitively, we can view the examples which do not effect the resulting alignment function as a validation set. By construction, the error of the alignment function on this validation set is small and thus it is very likely that the true risk of the alignment function (on unseen data) is also small. The following theorem formalizes this intuition.

Theorem 2. *Let $S = \{(\mathbf{z}_1, \bar{y}_1), \ldots, (\mathbf{z}_m, \bar{y}_m)\}$ be a training set of examples identically and independently distributed according to an unknown distribution Q. Assume that the assumptions of Thm. 1 hold. In addition, assume that $\gamma(\bar{y}, \bar{y}') \leq L$ for all pairs $(\bar{y}, \bar{y}')$ and let k be* *the smallest integer such that* $k \geq \|\mathbf{w}^\star\|^2/\epsilon_0$. *Let* f *be the alignment function obtained by running the algorithm from Fig. 1 on* S. *Then for any* $0 \leq \delta \leq 1$ *the following bound holds with a probability of at least* $1-\delta$

$$\mathit{risk}(f) \;\leq\; \epsilon_0 + L\sqrt{\frac{k\ln(em/k)+\ln(1/\delta)}{2(m-k)}} \;.$$

The proof of this theorem is also provided in a long version of the paper [15]. In summary, Thm. 2 states that if we present the learning algorithm with a large number of examples, the true risk of the resulting alignment function is likely to be small.

4. FEATURES

In this section we describe the alignment feature functions $\{\phi_j\}_{j=1}^n$. In our experiments we used $n = 10$ alignment features as follows.

The first 9 alignment features take the following form,

$$\phi_j(\bar{\mathbf{x}}, \bar{p}, \bar{s}, \bar{y}) = \sum_{i=1}^{|\bar{p}|} \hat{\phi}_j(\mathbf{x}_{y_i}, p_i)\;, \quad 1 \leq j \leq 9 \qquad (5)$$

where each $\hat{\phi}_j : \mathcal{X} \times \mathcal{P} \to \mathbb{R}$ $(1 \leq j \leq 9)$ is a set of local templates for an alignment function. Intuitively, $\hat{\phi}_j$ is the confidence that a pitch value p_i starts at time index y_i of the signal.

We now describe the specific form of each of the above local templates, starting with $\hat{\phi}_1$. Given the tth acoustic feature vector $\mathbf{x}_t$ and a pitch value $p \in \mathcal{P}$, the local template $\hat{\phi}_1(\mathbf{x}_t, p)$ is the energy of the acoustic signal at the frequency corresponding to the pitch p. Formally, let F_p denote a band-pass filter with a center frequency at the first harmony of the pitch p and cut-off frequencies of $1/4$ tone below and above p. Concretely, the lower cut-off frequency of F_p is $440 \cdot 2^{\frac{p-57-0.5}{12}} Hz$ and the upper cut-off frequency is $440 \cdot 2^{\frac{p-57+0.5}{12}} Hz$, where $p \in \mathcal{P} = \{0, 1, \ldots, 127\}$ is the pitch value (coded using the MIDI standard) and $440 \cdot 2^{\frac{p-57}{12}}$ is the frequency value in Hz associated with the codeword p. Similarly, $\hat{\phi}_2(\mathbf{x}_t, p)$ and $\hat{\phi}_3(\mathbf{x}_t, p)$ are the output energies of band-pass filters centered at the second and third pitch harmonics, respectively. All the filters were implemented using the fast Fourier transform.

The above 3 local templates $\{\hat{\phi}_j\}_{j=1}^3$ measure energy values for each time t. Since we are interested in identifying notes onset times, it is reasonable to compare energy values at time t with energy values at time $t-1$. However, the (discrete) first order derivative of the energy is highly sensitive to noise. Instead, we calculate the derivatives of a fitted second-order polynomial of each of the above local features. (This method is also a common technique in speech processing systems [11].) Therefore, the next 6 local templates $\{\hat{\phi}_j\}_{j=4}^9$ measure the first and second derivatives of the first 3 local templates.

While the first 9 alignment features measure confidence of alignment based on spectral properties of the signal, the last alignment feature captures the similarity between $\bar{s}$ and $\bar{y}$. Formally, let

$$\mu_i = \frac{y_i - y_{i-1}}{s_i - s_{i-1}} \qquad (6)$$

be the ratio between the ith interval according to the observation to the interval of the corresponding symbolic representation. We also refer to μ_i as the relative tempo. The sequence of relative tempo values is presumably constant in time, since $\bar{s}$ and $\bar{y}$ represent two performances of the *same* musical piece. However, in practice the tempo ratios often differ from performance to performance and within a given performance. The local template $\hat{\phi}_{10}$ measures the local change in the tempo,

$$\hat{\phi}_{10}(\mu_i, \mu_{i-1}) = (\mu_i - \mu_{i-1})^2 \;\;,$$

and ϕ_{10} is simply the cumulative sum of the changes in the tempo,

$$\phi_{10}(\bar{\mathbf{x}}, \bar{p}, \bar{s}, \bar{y}) = \sum_{i=2}^{|\bar{s}|} \hat{\phi}_{10}(\mu_i, \mu_{i-1}) \;. \qquad (7)$$

The relative tempo of Eq. (6) is ill-defined whenever $s_i - s_{i-1}$ is zero (or relatively small). Since we deal with polyphonic musical pieces, very short intervals between notes are rather relevant. Therefore, we define the tempo μ_i as in Eq. (6) but confine ourselves to indices i for which $s_i - s_{i-1}$ is greater than a predefined value τ (in our experiments we set $\tau = 60$ ms). Finally, we denote by $\hat{\boldsymbol{\phi}}(\mathbf{x}_t, p, \mu, \mu')$ the vector in $\mathbb{R}^{10}$ of local templates, whose jth element is $\hat{\phi}_j(\mathbf{x}_t, p)$ if $1 \leq j \leq 9$ and whose 10th element is $\hat{\phi}_{10}(\mu, \mu')$.

Input: Acoustic-symbolic representation $z = (\bar{\mathbf{x}}, \bar{p}, \bar{s})$;
An alignment function defined by a weight vector $\mathbf{w}$
Initialize: $\forall (1 \leq t \leq |\bar{\mathbf{x}}|),\; D(0, t, 1) = 0$
Recursion:
For $i = 1, \ldots, |\bar{p}|$
For $\mathbf{t} = 0, \ldots, |\bar{\mathbf{x}}|$
For $\mu \in M$
If $(s_i - s_{i-1} > \tau)$
$D(i, t, \mu) = \max_{\mu' \in M} D(i{-}1, t{-}l, \mu') + \mathbf{w} \cdot \hat{\boldsymbol{\phi}}(\mathbf{x}_t, p_i, \mu, \mu')$,
where $l = \mu'(s_i - s_{i-1})$
Else [If $(s_i - s_{i-1} \leq \tau)$]
$D(i, t, \mu) = \max_{l \in L} D(i{-}1, t{-}l, \mu) + \mathbf{w} \cdot \hat{\boldsymbol{\phi}}(\mathbf{x}_t, p_i, \mu, \mu)$,
where $L = \{-\tau, -\tau+1, \ldots, \tau-1, \tau\}$
Termination: $D^\star = \max_{t,\mu} D(|\bar{p}|, t, \mu)$

Figure 2. The procedure for calculating the best alignment.

5. EFFICIENT CALCULATION OF THE ALIGNMENT FUNCTION

So far we have put aside the problem of evaluation time of the function f. Recall that calculating f requires solving the following optimization problem,

$$f(\mathbf{z}) = \operatorname*{argmax}_{\bar{y}} \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{z}, \bar{y}) \;.$$

A direct search for the maximizer is not feasible since the number of possible alignment sequences $\bar{y}$ is exponential in the length of the sequence. Fortunately, as we show below, by imposing a few mild conditions on the structure of the alignment feature functions, the best alignment sequence can be found in polynomial time.

For simplicity, we describe an efficient algorithm for calculating the best alignment using the feature functions ϕ_j described in Sec. 4. Similar algorithms can be constructed for any feature functions that can be described as a dynamic Bayesian network (c.f. [6, 17]).

We now turn to the description of a dynamic programming procedure for finding the best alignment given an alignment function defined by a weight vector $\mathbf{w}$. Let M be the set of potential tempo ratios of the form $(y_i - y_{i-1})/(s_i - s_{i-1})$. For a given ratio $\mu \in M$, denote by $D(i, t, \mu)$ the score for the prefix of the notes sequence $1, \ldots, i$ assuming that their actual start times are $y_1, \ldots, y_{i-1}$ and for the last note $y_i = t$ with $\mu = (y_i - y_{i-1})/(s_i - s_{i-1})$. This variable can be computed efficiently in a similar fashion to the forward variables calculated by the Viterbi procedure in HMMs (see for instance [12]). The pseudo code for computing $D(i, t, \mu)$ recursively is shown in Fig. 2. The best sequence of actual start times, $\bar{y}'$, is obtained from the algorithm by saving the intermediate values that maximize each expression in the recursion step. The complexity of the algorithm is $O(|\bar{p}|\ |\bar{\mathbf{x}}|\ |M|^2)$, where $|M|$ is the size of the set M. Note that $|M|$ is trivially upper bounded by $|\bar{\mathbf{x}}|^2$. However, in practice, we can discretize the set of tempo ratios and obtain a good approximation to the actual ratios. In our experiments we chose this set to be $M = \{2^{-1}, 2^{-0.5}, 1, 2^{0.5}, 2^1\}$.

6. GENERATIVE METHOD FOR ALIGNMENT

We compare our discriminative method for alignment to a generative method based on the Generalized Hidden Markov Model (GHMM) [10]. In the generative setting, we assume that the acoustic signal $\bar{\mathbf{x}}$ is generated from the symbolic representation $(\bar{p}, \bar{s})$ as follows. First, the actual start times sequence $\bar{y}$ is generated from $\bar{s}$. Then, the acoustic signal $\bar{\mathbf{x}}$ is generated from the pitch sequence $\bar{p}$ and the actual start time sequence $\bar{y}$. Therefore,

$$\begin{aligned}\Pr[\bar{\mathbf{x}}|\bar{p}, \bar{s}] &= \sum_{\bar{y}} \Pr[\bar{\mathbf{x}}, \bar{y}|\bar{p}, \bar{s}] \\ &= \sum_{\bar{y}} \Pr[\bar{y}|\bar{s}] \Pr[\bar{\mathbf{x}}|\bar{y}, \bar{p}] \quad .\end{aligned}$$

We now describe the parametric form we use for each of the terms in the above equation. As in [14], we model the probability of the actual start-times given the symbolic start time by $\Pr[\bar{y}|\bar{s}] = \prod_{i=1}^{|\bar{y}|} \Pr[\mu_i|\mu_{i-1}]$, where μ_i is as defined in Sec. 4. In our experiments, we estimated the probability $\Pr[\mu_i|\mu_{i-1}]$ from the training data. To model the probability $\Pr[\bar{\mathbf{x}}|\bar{y}, \bar{p}]$ we use two Gaussian Mixture Models (GMM). The first GMM approximates

	GHMM-1	GHMM-3	GHMM-5	GHMM-7	Discrim.
1	10.0	188.9	49.2	69.7	**8.9**
2	15.3	159.7	31.2	20.7	**9.1**
3	22.5	48.1	29.4	37.4	**17.1**
4	12.7	29.9	15.2	17.0	**10.0**
5	54.5	82.2	55.9	53.3	**41.8**
6	12.8	46.9	26.7	23.5	**14.0**
7	336.4	75.8	30.4	43.3	**9.9**
8	11.9	24.2	15.8	17.1	**11.4**
9	11473	11206	51.6	12927	**20.6**
10	16.3	60.4	16.5	20.4	**8.1**
11	22.6	39.8	27.5	19.2	**12.4**
12	13.4	14.5	13.8	28.1	**9.6**
mean	1000.1	998.1	30.3	1106.4	**14.4**
std	3159	3078.3	14.1	3564.1	**9.0**
median	15.8	54.2	28.5	25.8	**10.7**

Table 1. Summary of the LOO loss (in ms) for different algorithms for alignment.

the probability of $\mathbf{x}_t$ given that a pitch p starts at time t. We denote this probability function by $\Pr[\mathbf{x}_t|p]$. The second GMM approximates the probability of $\mathbf{x}_t$ given that a pitch p does *not* start at time t. This probability is denoted by $\Pr[\mathbf{x}_t|\neg p]$. For a given time t, let $\mathcal{P}_t = \{p \in \mathcal{P} : \exists i, y_i = t, p_i = p\}$ be the set of all pitches of notes that start on time t, and let $\overline{\mathcal{P}_t} = \mathcal{P} \setminus \mathcal{P}_t$ be the completion set. Using the above definitions the probability of the acoustic signal $\bar{\mathbf{x}}$ given the actual start time sequence $\bar{y}$ and the pitch sequence $\bar{p}$ can be written as

$$\Pr[\bar{\mathbf{x}}|\bar{y}, \bar{p}] = \prod_{t=1}^{|\bar{\mathbf{x}}|} \prod_{p:\mathcal{P}_t} \Pr[\mathbf{x}_t|p] \prod_{p:\overline{\mathcal{P}_t}} \Pr[\mathbf{x}_t|\neg p] \quad .$$

We estimated the parameters of the GMMs from the training set using the Expectation Maximization (EM) algorithm. The best alignment of a new example $(\bar{\mathbf{x}}, \bar{p}, \bar{s})$ from the test set is the alignment sequence $\bar{y}'$ that maximizes the likelihood of $\bar{\mathbf{x}}$ according to the model described above. A detailed description of the dynamic programming procedure for finding the best alignment is provided in a long version of this paper [15].

7. EXPERIMENTAL RESULTS

In this section we describe experiments with the algorithms described above for the task of alignment of polyphonic piano musical pieces. Specifically, we compare our discriminative and generative algorithms. Recall that our algorithms use a training set of alignment examples for deducing an alignment function. We downloaded 12 musical pieces from `http://www.piano-midi.de/mp3.php` where sound and MIDI were both recorded. Here the sound serves as the acoustical signal $\bar{\mathbf{x}}$ and the MIDI is the actual start times $\bar{y}$. We also downloaded other MIDI files of the same musical pieces from a variety of other websites and used these MIDI files for creating the sequences $\bar{p}$ and $\bar{s}$. The complete dataset we used is available from `http://www.cs.huji.ac.il/~shais/alignment`.

We used the leave-one-out (LOO) cross-validation procedure for evaluating the test results. In the LOO setup the algorithms are trained on all the training examples except one, which is used as a test set. The loss between the predicted and true start times is computed for each of the algorithms. We compared the results of the discriminative learning algorithm described in Sec. 3 to the results of the generative learning algorithm described in Sec. 6. Recall that the generative algorithm uses a GMM to model some of the probabilities. The number of Gaussians used by the GMM needs to be determined. We used the values of 1, 3, 5 and 7 as the number of Gaussians and we denote by GHMM-n the resulting generative model with n Gaussians. In addition, we used the EM algorithm to train the GMMs. The EM algorithm converges to a local maximum, rather than to the global maximum. A common solution to this problem is to use a random partition of the data to initialize the EM. In all our experiments with the GMM we used 15 random partitions of the data to initialize the EM and chose the one that leads to the highest likelihood. The LOO results for each of the 12 musical pieces are summarized in Table 1. As seen from the table, the discriminative learning algorithm outperforms all the variants of generative algorithms in all of the experiments. Moreover, in all but two of the experiments the loss of the discriminative algorithm is less than 20 ms, which is the length of an acoustic frame in our experiments, thus it is the best accuracy one can hope for this time resolution. It can be seen that the variance of the LOO loss obtained by the generative algorithms is rather high. This can be attributed to the fact that the EM algorithm converges to a local maximum which depends on initialization of the parameters. Therefore, we omitted the highest and lowest loss values obtained by each of the algorithms and re-calculated the average loss over the 12 experiments. The resulting mean values along with the range of the loss values are depicted in Fig. 3.

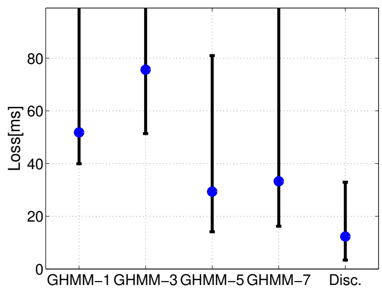

Figure 3. The average loss of all the LOO experiments excluding the best and worst results.

8. FUTURE WORK

We are currently pursuing a few extensions. First, we are now working on applying the methods described in this paper to other musical instruments. The main difficulty here is to obtain a training set of labeled examples. We are examining semi-supervised methods that might overcome the lack of supervision. Second, we plan to automatically generate large databases of aligned acoustic-symbolic representations of musical pieces. These datasets would serve as a necessary step towards the implementation of a polyphonic note detection system.

Acknowledgements Thanks to Ofer Dekel, Nir Kruase, and Moria Shalev for helpful comments on the manuscript. This work was supported by EU PASCAL Network Of Excellence.

9. REFERENCES

[1] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *ICML*, 2003.

[2] M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

[3] K. Crammer, O. Dekel, S. Shalev-Shwartz, and Y. Singer. Online passive aggressive algorithms. In *NIPS*, 2003.

[4] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

[5] R. Dannenberg. An on-line algorithm for real-time accompaniment. *ICMC*, 1984.

[6] T. Dean and K. Kanazawa. A model for reasoning about persistent and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[7] A. S. Durey and M. A. Clements. Melody spotting using hidden Markov models. In *ISMIR*, 2001.

[8] M. Herbster. Learning additive models online with fast evaluating kernels. In *COLT*, 2001.

[9] A. Klapuri, T. Virtanen, A. Eronen, and J. Seppanen. Automatic transcription of musical recordings. In *CRAC*, 2001.

[10] L.T. Niles and H.F. Silverman. Combining hidden Markov model and neural network classifiers. In *ICASSP*, 1990.

[11] L. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[12] L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3(1):4–16, Jan. 1986.

[13] C. Raphael. Automatic segmentation of acoustic musical signals using hidden Markov models. *IEEE trans. on Pattern Analysis and Machine Intelligence*, 21(4), April 1999.

[14] S. Shalev-Shwartz, S. Dubnov, N. Friedman, and Y. Singer. Robust temporal and spectral modeling for query by melody. In *SIGIR*, 2002.

[15] S. Shalev-Shwartz, J. Keshet, and Y. Singer. Learning to align polyphonic music. Long version. http://www.cs.huji.ac.il/∼shais/ShalevKeSi04long.ps .

[16] F. Soulez, X. Rodet, and D. Schwarz. Improving polyphonic and poly-instrumental music to score alignment. In *ISMIR*, 2003.

[17] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *NIPS*, 2003.

[18] R. Turetsky and D. Ellis. Ground-truth transcriptions of real music from force-aligned MIDI syntheses. In *ISMIR*, 2003.

[19] V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[20] P.J. Walmsley, S.J. Godsill, and P.J.W. Rayner. Polyphonic pitch tracking using joint Bayesian estimation of multiple frame parameters. In *Proc. Ieee Workshop on Applications of Signal Processing to Audio and Acoustics*, October 1999.