A Large Margin Algorithm for Speech-to-Phoneme and Music-to-Score Alignment

Joseph Keshet, Shai Shalev-Shwartz, Yoram Singer, and Dan Chazan

***Abstract*—We describe and analyze a discriminative algorithm for learning to align an audio signal with a given sequence of events that tag the signal. We demonstrate the applicability of our method for the tasks of speech-to-phoneme alignment ("forced alignment") and music-to-score alignment. In the first alignment task, the events that tag the speech signal are phonemes while in the music alignment task, the events are musical notes. Our goal is to learn an alignment function whose input is an audio signal along with its accompanying event sequence and its output is a timing sequence representing the actual start time of each event in the audio signal. Generalizing the notion of separation with a margin used in support vector machines for binary classification, we cast the learning task as the problem of finding a vector in an abstract inner-product space. To do so, we devise a mapping of the input signal and the event sequence along with any possible timing sequence into an abstract vector space. Each possible timing sequence therefore corresponds to an instance vector and the predicted timing sequence is the one whose projection onto the learned prediction vector is maximal. We set the prediction vector to be the solution of a minimization problem with a large set of constraints. Each constraint enforces a gap between the projection of the correct target timing sequence and the projection of an alternative, incorrect, timing sequence onto the vector. Though the number of constraints is very large, we describe a simple iterative algorithm for efficiently learning the vector and analyze the formal properties of the resulting learning algorithm. We report experimental results comparing the proposed algorithm to previous studies on speech-to-phoneme and music-to-score alignment, which use hidden Markov models. The results obtained in our experiments using the discriminative alignment algorithm are comparable to results of state-of-the-art systems.**

***Index Terms*—Forced alignment, large margin and kernel methods, music, speech processing, support vector machines (SVMs).**

I. Introduction

In THIS paper we describe a new approach for learning to align an audio signal with a given sequence of events associated with the signal. We focus on two applications of the above task: speech-to-phoneme alignment and music-to-score alignment. In speech-to-phoneme alignment ("forced alignment") tasks, the events are phonemes and the goal is to predict the start time of each phoneme in the spoken utterance. In music-to-score alignment, we are given a sequence of musical notes (extracted from a musical score) along with a recording of the musical piece and the goal is to predict the start time of each note in the recorded audio signal.

Most of the previous work on speech-to-phoneme and music-to-score alignment focused on a generative model of the audio signal using hidden Markov models (HMMs). See, for example, [1]–[5] and the references therein. Despite their popularity, HMM-based approaches have several drawbacks such as convergence of the EM procedure to local maxima and overfitting effects due to the large number of parameters. In this paper we propose an alternative approach for learning alignment functions that builds upon recent work on discriminative supervised learning. 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. [6] and [7]). One of the best known discriminative learning algorithms is the support vector machine (SVM), which has been successfully applied in speech processing applications [8]–[10]. The classical SVM algorithm is designed for simple decision tasks such as binary classification and regression. Hence, its exploitation in signal processing systems so far has also been restricted to simple decision tasks such as phoneme classification and music genre classification. The alignment problem is more involved, since we need to predict a sequence of event timings rather than a single number. The main challenge of this paper is to extend the notion of discriminative learning to the complex task of alignment.

Our proposed method is based on recent advances in kernel machines and large margin classifiers for sequences [11]–[13], which in turn build on the pioneering work of Vapnik and colleagues [6], [7]. The alignment function we devise is based on mapping the audio signal and the sequence of events along with the target event timing sequence into an abstract vector-space. Building on techniques used for learning SVMs, our alignment function distills to a classifier in this vector-space which is aimed at separating correct timing sequences from incorrect ones. We describe a simple iterative algorithm for learning the alignment function and discuss its formal properties. The specific form of the iterative algorithm stems from recent work on online algorithms [14] and our analysis is based on a recent framework for analyzing online algorithms [15].

This paper is organized as follows. In Section II, we formally introduce the general alignment problem and our two applications, namely, speech-to-phoneme alignment and music-to-

Manuscript received February 16, 2007; revised May 29, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Timothy J. Hazen.

J. Keshet and S. Shalev-Skwartz are with the School of Computer Science and Engineering, The Hebrew University of Jerusalem, Edmond Safra Campus, Givat Ram, Jerusalem 91904 Israel (e-mail: jkeshet@cs.huji.ac.il; shais@cs.huji.ac.il).

Y. Singer is with Google, Inc., Mountain View, CA 94043 USA (e-mail: singer@google.com).

D. Chazan is with the Electrical Engineering Department, The Technion, Haifa 32000, Israel (e-mail: dan_chazan@yahoo.com).

Digital Object Identifier 10.1109/TASL.2007.903928

score alignment. In Section III we describe a discriminative supervised learning approach for learning an alignment function from a training set of examples and specifically, in Section IV, we describe a large margin approach for the alignment problem. Our specific learning algorithm is described and analyzed in Section V. The evaluation of the alignment function and the learning algorithm are both based on an optimization problem for which we give an efficient dynamic programming procedure in Section VI. Next, in Sections VII and VIII, we describe the applicability of our method to speech-to-phoneme alignment and to music-to-score alignment. We present experimental results in which we compare our method to alternative state-of-the-art approaches. Finally, concluding remarks and future directions are discussed in Section IX.

II. The Alignment Problem

In the alignment problem, we are provided with a signal which is accompanied with a discrete sequence of symbols or events and the goal is to align each of the events in the tagging sequence with its corresponding position in the signal. In speech-to-phoneme alignment, the events designate the phoneme uttered in the signal. In music-to-score alignment, the events are the notes in the score accompanying the signal. The alignment problem is the task of finding the start time of each tagged event in the input signal.

We represent a signal as a sequence of acoustic feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t$ is a d-dimensional vector. For brevity, we denote the domain of the feature vectors by $\mathcal{X} \subset \mathbb{R}^d$. Naturally, the length of the acoustic signal varies from one signal to another and thus T is not fixed. We denote by $\mathcal{X}^*$ the set of all finite-length sequences over $\mathcal{X}$. The sequence of events is denoted by $\bar{e} = (e_1, \ldots, e_K)$, where $e_k \in E$ for all $1 \leq k \leq K$ and E is the domain of the events. We assume that E is a finite set and we denote by E^* the set of all finite-length sequences over E. In summary, each input is a pair $(\bar{\mathbf{x}}, \bar{e})$ where $\bar{\mathbf{x}}$ is a sequence representing the acoustic signal and $\bar{e}$ is a sequence of events that occur in the signal. The alignment of the signal $\bar{\mathbf{x}}$ with the events $\bar{e}$ is a sequence of start-times $\bar{y} = (y_1, \ldots, y_K)$, where $y_k \in \{1, \ldots, T\}$ is the start-time of the event e_k in the acoustic signal. Our goal is to learn an *alignment function*, denoted f, which takes as input the pair $(\bar{\mathbf{x}}, \bar{e})$ and returns an event timing sequence $\bar{y}$. That is, f is a function from $\mathcal{X}^* \times E^*$ to the set of finite-length sequences over the integers, $\mathbb{N}^*$.

In this paper, we focus on two applications of the above general setting: speech-to-phoneme alignment and music-to-score alignment. In both problems, the acoustic representation $\bar{\mathbf{x}}$ is produced by dividing the acoustic signal into frames of several milliseconds, and extracting a d dimensional feature vector from each frame. In the speech-to-phoneme alignment problem the feature vector extracted from each frame is the Mel-frequency cepstrum coefficients (MFCC) along with their first and second derivatives. The sequence of events is a sequence of phoneme symbols from E, where E is the set of 48 American English phoneme symbols as proposed by [16]. We assume that the acoustic signal is an utterance of the phoneme sequence $\bar{e} = (e_1, \ldots, e_K)$ and our goal is to find the start time of each phoneme in the utterance.

In the music-to-score alignment problem, each acoustic feature vector $\mathbf{x}_t$ in the sequence $\bar{\mathbf{x}}$ is produced by calculating the short time Fourier transform of the tth frame of the signal. E is a set of "note-on" events. Formally, each "note-on" event is a pair $e_k = (p_k, s_k)$. The first element of the pair, $p_k \in \mathcal{P} = \{0, 1, \ldots, 127\}$ is the note's pitch value (coded using the MIDI standard). The second element, s, is assumed to be a positive integer ($s_k \in \mathbb{N}$) as it measures the (theoretical) start time of the note according to the musical score. 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 in the music score alignment task is to find the actual start time of each note in the acoustic signal.

III. Discriminative Supervised Learning

In this section, we describe a discriminative supervised learning approach for learning an alignment function f from a training set of examples. Each example in the training set is composed of an acoustic signal, $\bar{\mathbf{x}}$, a sequence of events, $\bar{e}$, and the true event timing sequence, $\bar{y}$. Our goal is to find an alignment function, f, which performs well on the training set as well as on unseen examples. First, we define a quantitative assessment of alignment functions. Let $(\bar{\mathbf{x}}, \bar{e}, \bar{y})$ be an input example and let f be an alignment function. We denote by $\gamma(\bar{y}, f(\bar{x}, \bar{e}))$ the cost of predicting the timing sequence $f(\bar{\mathbf{x}}, \bar{e})$ where the true timing sequence is $\bar{y}$. Formally, $\gamma : \mathbb{N}^* \times \mathbb{N}^* \to \mathbb{R}$ is a function that gets two timing sequences (of the same length) and returns a scalar which is the cost of predicting the second timing sequence where the true timing sequence is the first. We assume that $\gamma(\bar{y}, \bar{y}') \geq 0$ for any two timing sequences $\bar{y}$, $\bar{y}'$ and that $\gamma(\bar{y}, \bar{y}) = 0$. An example for a cost function is

$$\gamma(\bar{y}, \bar{y}') = \frac{1}{|\bar{y}|} \left| \{ i : |y_i - y'_i| > \epsilon \} \right| . \tag{1}$$

In words, the above cost is the average number of times the absolute difference between the predicted timing sequence and the true timing sequence is greater than ϵ. Recall that our goal 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 the examples, $\mathcal{X}^* \times E^* \times \mathbb{N}^\star$. The goal of the learning process is to minimize the risk of using the alignment function, defined as the expected cost of f on the examples, where the expectation is taken with respect to the distribution Q

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

To do so, we assume that the examples of our training set are 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. The training set of examples is used as a restricted window through which we estimate the quality of alignment functions according to the distribution of unseen examples in the real world, Q. In the next sections, we show how to use the training set in order to find an alignment function, f, which achieves a small cost on the training set, and which achieves a small cost on unseen examples with high probability as well.

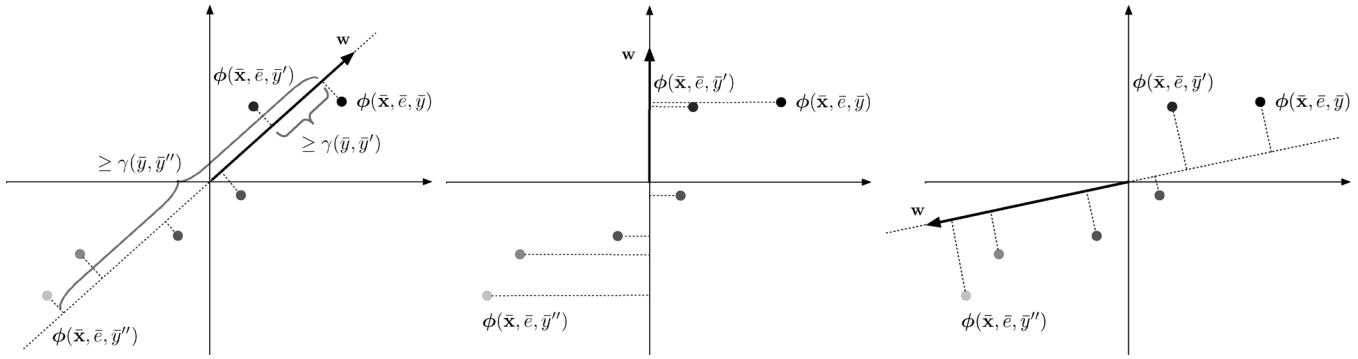

Fig. 1. Illustration of the constraints in (3). Left: a projection which attains large margin. Middle: a projection which attains a smaller margin. Right: an incorrect projection.

IV. A Large Margin Approach for Alignment

In this section, we describe a large margin approach for learning an alignment function. Recall that a supervised learning algorithm for alignment receives as input a training set $S = \{(\bar{\mathbf{x}}_1, \bar{e}_1, \bar{y}_1), \ldots, (\bar{\mathbf{x}}_m, \bar{e}_m, \bar{y}_m)\}$ and returns an alignment function f. 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 base alignment feature functions, $\{\phi_j\}_{j=1}^n$. Each base alignment feature is a function of the form $\phi_j : \mathcal{X}^* \times E^* \times \mathbb{N}^* \to \mathbb{R}$. That is, each base alignment feature gets the acoustic representation, $\bar{\mathbf{x}}$, and the sequence of events, $\bar{e}$, together with a candidate timing sequence, $\bar{y}$, and returns a scalar which, intuitively, represents the confidence in the suggested timing sequence $\bar{y}$. The construction of those base alignment features is task dependent. As an example, let us shortly describe a single base alignment feature for the speech-to-phoneme alignment task. This base alignment feature sums a cepstral distance between the frames $\mathbf{x}_{y_i+1}$ and $\mathbf{x}_{y_i-1}$ over $i = 1, 2, \ldots, |\bar{y}|$. For each i, if y_i is indeed the correct start time of phoneme i, we expect the distance between $\mathbf{x}_{y_i+1}$ and $\mathbf{x}_{y_i-1}$ to be large. On the other hand, if y_i does not reflect a true alignment point then the distance is likely to be small. Naturally, it is naive to assume that the above base alignment feature can be used alone for finding the correct timing sequence. However, as our experiments show, an appropriate combination of a few base alignment features enables us to accurately predict the correct timing sequence.

We denote by $\boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{e}, \bar{y})$ the vector in $\mathbb{R}^n$ whose jth element is $\phi_j(\bar{\mathbf{x}}, \bar{e}, \bar{y})$. The alignment functions we use are of the form

$$f(\bar{\mathbf{x}}, \bar{e}) = \arg\max_{\bar{y}} \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{e}, \bar{y}) \tag{2}$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of importance weights that we need to learn. In words, f returns a suggestion for a timing sequence by maximizing a weighted sum of the confidence scores returned by each base alignment function ϕ_j. Since f is parameterized by $\mathbf{w}$, we use the notation $f_{\mathbf{w}}$ for an alignment function f, which is defined as in (2). Note that the number of possible timing sequences, $\bar{y}$, is exponentially large. Nevertheless, as we show later, under mild conditions on the form of the base alignment functions $\{\phi_j\}$, the optimization problem in (2) can be efficiently calculated using a dynamic programming procedure.

We now describe a large margin approach for learning the weight vector $\mathbf{w}$, which defines an alignment function as in (2), from a training set $S = \{(\bar{\mathbf{x}}_1, \bar{e}_1, \bar{y}_1), \ldots, (\bar{\mathbf{x}}_m, \bar{e}_m, \bar{y}_m)\}$ of examples. Similar to the SVM algorithm for binary classification, our approach for choosing the weight vector $\mathbf{w}$ is based on the idea of large-margin separation. However, in our case, timing sequences are not merely correct or incorrect. Instead, the cost function $\gamma(\bar{y}, \bar{y}')$ is used for assessing the quality of sequences. Therefore, we do not aim at separating correct timing sequences from incorrect ones but rather try to rank the sequences according to their quality. Theoretically, our approach can be described as a two-step procedure: first, we construct a vector $\boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}')$ in the vector space $\mathbb{R}^n$ based on each instance $(\bar{\mathbf{x}}_i, \bar{e}_i)$ in the training set S and each possible timing sequence $\bar{y}'$. Second, we find a vector $\mathbf{w} \in \mathbb{R}^n$, such that the projection of vectors onto $\mathbf{w}$ ranks the vectors constructed in the first step above according to their quality. In Fig. 1, we illustrate three possible timing sequences for the same input $(\bar{\mathbf{x}}, \bar{e})$ and their projection onto $\mathbf{w}$. Ideally, for each instance $(\bar{\mathbf{x}}_i, \bar{e}_i)$ and for each possible suggested timing sequence $\bar{y}'$, we would like the following constraint to hold:

$$\mathbf{w} \cdot \phi(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i) - \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}') \geq \gamma(\bar{y}_i, \bar{y}'). \tag{3}$$

That is, $\mathbf{w}$ should rank the correct timing sequence $\bar{y}_i$ above any other possible timing sequence $\bar{y}'$ by at least $\gamma(\bar{y}_i, \bar{y}')$. We refer to the difference $\mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i) - \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}')$ as the *margin* of $\mathbf{w}$ with respect to the sequence $\bar{y}'$. Note that if the prediction of $\mathbf{w}$ is incorrect, then the margin is negative. The constraints in (3) imply that the margin of $\mathbf{w}$ with respect to any possible timing sequence $\bar{y}'$ should be at least the cost of predicting $\bar{y}'$ instead of the true timing sequence $\bar{y}_i$. An illustration of a vector $\mathbf{w}$ with sufficient margin (i.e., satisfies the constraints in (3)) is given on the left side of Fig. 1. The plot on the middle of Fig. 1 illustrates a vector $\mathbf{w}$, which ranks the different timing sequences correctly, but without the required margin. The plot on the right side of Fig. 1 illustrates a vector $\mathbf{w}$ which does not rank the different timing sequences correctly. Naturally, if $\mathbf{w}$ ranks the different possible timing sequences correctly, the margin requirements given in (3) can be satisfied by simply multiplying $\mathbf{w}$ by a large scalar. The SVM algorithm solves this problem by minimizing $1/2\,||\mathbf{w}||^2$ subject to the constraints given in (3).

In practice, it might be the case that the constraints given in (3) can not be satisfied. To overcome this obstacle, we follow the

soft SVM approach and define the following hinge-loss function for alignment

$$\ell(\mathbf{w};(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)) = \max_{\bar{y}'}\left[\gamma(\bar{y}_i,\bar{y}') - \mathbf{w} \cdot (\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}'))\right]_+ \quad (4)$$

where $[a] = \max\{0, a\}$. The hinge loss measures the maximal violation of any of the constraints given in (3). The soft SVM approach for alignment is to choose the vector $\mathbf{w}^\star$, which minimizes the following optimization problem:

$$\mathbf{w}^\star = \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \ell(\mathbf{w};(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)) \quad (5)$$

where the parameter C serves as a complexity-accuracy trade-off parameter (see [7]). It is easy to verify that the optimization problem in (5) is equivalent to the following quadratic optimization problem:

$$\min_{\mathbf{w},\boldsymbol{\xi}\geq 0} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i \quad \text{s.t.}$$
$$\forall i, \bar{y}', \mathbf{w}\cdot(\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}')) \geq \gamma(\bar{y}_i,\bar{y}') - \xi_i \quad (6)$$

where each ξ_i is a non-negative slack variable that indicates the loss of the ith example.

Solving the quadratic optimization problem given in (6) is complicated since the number of constraints is exponentially large. Several authors suggested specific algorithms for manipulating the exponential number of constraints [11], [13]. However, these methods are problematic when the size of the dataset is very large since several passes over the data are required. In the next section, we propose an alternative method, which visits each example only once.

V. An Iterative Algorithm

In this section, we describe an iterative algorithm for learning an alignment function, parameterized by $\mathbf{w}$. Our iterative algorithm first constructs a sequence of weight vectors $\mathbf{w}_1,\ldots,\mathbf{w}_m,\mathbf{w}_{m+1}$. The first weight vector is set to be the zero vector, $\mathbf{w}_1 = \mathbf{0}$. On iteration i of the algorithm, we utilize the ith example of the training set along with the previous weight vector $\mathbf{w}_i$, for defining the next weight vector $\mathbf{w}_{i+1}$ as follows. Let $\bar{y}_i'$ be the timing sequence, which corresponds to the highest violated margin constraint of the ith example according to $\mathbf{w}_i$, that is

$$\bar{y}_i' = \arg\max_{\bar{y}} \gamma(\bar{y},\bar{y}_i) - \mathbf{w}\cdot(\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y})). \quad (7)$$

In Section VI, we provide an algorithm that efficiently calculates the above optimization problem using dynamic programming. We set the next weight vector $\mathbf{w}_{i+1}$ to be the minimizer of the following optimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^n,\xi\geq 0} \quad \frac{1}{2}\|\mathbf{w}-\mathbf{w}_i\|^2 + C\xi \quad \text{s.t.}$$
$$\mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i') \geq \gamma(\bar{y}_i,\bar{y}_i') - \xi. \quad (8)$$

INPUT: training set $S = \{(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)\}_{i=1}^m$; validation set S_v ; parameter C

INITIALIZE: $\mathbf{w}_1 = \mathbf{0}$

FOR $i = 1,\ldots,m$

Set: $\bar{y}_i' = \operatorname{argmax}_{\bar{y}} \gamma(\bar{y}_i,\bar{y}) - \mathbf{w}\cdot(\boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}))$

Set: $\Delta\boldsymbol{\phi}_i = \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i')$

Set: $\ell_i = \max\{\gamma(\bar{y}_i,\bar{y}_i') - \mathbf{w}\cdot\Delta\boldsymbol{\phi}_i, 0\}$

Update: $\mathbf{w}_{i+1} = \mathbf{w}_i + \min\{\ell_i/\|\Delta\boldsymbol{\phi}_i\|^2, C\}\,\Delta\boldsymbol{\phi}_i$

OUTPUT: The weight vector which achieves the lowest average cost on the validation set S_v

Fig. 2. Alignment algorithm.

This optimization problem can be thought of as a relaxed version of the SVM optimization problem with two major differences. First, we replace the exponential number of constraints from (6) with a single constraint. This constraint is based on the timing sequence $\bar{y}_i'$ defined in (7). Second, we replaced the term $\|\mathbf{w}\|^2$ in the objective function of the SVM with the term $\|\mathbf{w}-\mathbf{w}_i\|^2$. Intuitively, we would like to minimize the loss of $\mathbf{w}$ on the current example, i.e., the slack variable ξ, while remaining as close as possible to our previous weight vector $\mathbf{w}_i$. It can be shown (see [14]) that the solution to the above optimization problem is

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \min\left\{\frac{\ell(\mathbf{w}_i;(\mathbf{x}_i,\bar{e}_i,\bar{y}_i))}{\|\Delta\boldsymbol{\phi}_i\|^2}, C\right\}\cdot\Delta\boldsymbol{\phi}_i$$

where $\Delta\boldsymbol{\phi}_i = \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)$.

The above iterative procedure gives us a sequence of $m+1$ weight vectors, $\mathbf{w}_1,\ldots,\mathbf{w}_{m+1}$. In the sequel we prove that the average performance of this sequence of vectors is comparable to the performance of the SVM solution. Formally, let $\mathbf{w}^\star$ be the optimum of the SVM problem given in (6). Then, we show in the sequel that setting $C = 1/\sqrt{m}$ gives

$$\frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{w}_i;(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)) \leq \frac{1}{m}\sum_{i=1}^{m}\ell(\mathbf{w}^\star;(\bar{\mathbf{x}}_i,\bar{e}_i,\bar{y}_i)) + \frac{1}{\sqrt{m}}\left(\|\mathbf{w}^\star\|^2 + \frac{1}{2}\right). \quad (9)$$

That is, the average loss of our iterative procedure is upper bounded by the average loss of the SVM solution plus a factor that decays to zero. However, while each prediction of our iterative procedure is calculated using a different weight vector, our learning algorithm is required to output a *single* weight vector, which defines the output alignment function. To overcome this obstacle, we calculate the average cost of each of the weight vector $\mathbf{w}_1,\ldots,\mathbf{w}_{m+1}$ on a validation set, denoted S_v, and choose the one achieving the lowest average cost. We show in the sequel that with high probability, the weight vector which achieves the lowest cost on the validation set also generalizes well. A pseudocode of our algorithm is given in Fig. 2.

A. Analysis

We now analyze our alignment algorithm from Fig. 2. Our first theorem shows that the average loss of our alignment algorithm is comparable to the average loss of the SVM solution for the alignment problem defined in (6).

Theorem 1: Let $S = \{(\bar{\mathbf{x}}_1, \bar{e}_1, \bar{y}_1), \ldots, (\bar{\mathbf{x}}_m, \bar{e}_m, \bar{y}_m)\}$ be a set of training examples and assume that for all i and $\bar{y}'$ we have that $\|\boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}')\| \leq 1/2$. Let $\mathbf{w}^\star$ be the optimum of the SVM problem given in (6). Let $\mathbf{w}_1, \ldots, \mathbf{w}_m$ be the sequence of weight vectors obtained by the algorithm in Fig. 2 given the training set S. Then

$$\frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) \leq \frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) + \frac{1}{Cm}\|\mathbf{w}^\star\|^2 + \frac{1}{2}C. \quad (10)$$

In particular, if $C = 1/\sqrt{m}$, then

$$\frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) \leq \frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) + \frac{1}{\sqrt{m}}\left(\|\mathbf{w}^\star\|^2 + \frac{1}{2}\right). \quad (11)$$

The proof of this theorem is based on [15, Theor. 2] and is given in the Appendix.

The next theorem tells us that the output alignment function of our algorithm is likely to have good generalization properties.

Theorem 2: Under the same conditions of Theorem 1. Assume that the training set S and the validation set S_v are both sampled i.i.d. from a distribution Q. Denote by m_v the size of the validation set. Assume in addition that $\gamma(\bar{y}, \bar{y}') \leq 1$ for all $\bar{y}$ and $\bar{y}'$. Let $\mathbf{w}$ be the output weight vector of the algorithm in Fig. 2 and let $f_{\mathbf{w}}$ be the corresponding alignment function. Then, with probability of at least $1 - \delta$, we have that

$$\text{risk}(f_{\mathbf{w}}) \leq \frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) + \frac{\|\mathbf{w}^\star\|^2 + \frac{1}{2} + \sqrt{2\ln(2/\delta)}}{\sqrt{m}} + \frac{\sqrt{2\ln(2m/\delta)}}{\sqrt{m_v}}. \quad (12)$$

The proof of this theorem is based on [17, Prop. 1] and is also given in the Appendix.

As mentioned before, the learning algorithm we present in this paper share similarities with the SVM method for structured output prediction [11], [13]. Yet, the weight vector resulted by our method is not identical to the one obtained by directly solving the SVM optimization problem. We would like to note in passing that our generalization bound from Theorem 2 is comparable to generalization bounds derived for the SVM method (see, for example, [11]). The major advantage of our method over directly solving the SVM problem is its simplicity and efficiency.

To conclude this section, we would like to emphasize the role of the function γ in our model. Recall that the risk of an alignment function is defined to be the expected value of $\gamma(\bar{y}, f(\bar{\mathbf{x}}, \bar{e}))$. The constraints we imposed in (6) were constructed so that the risk will be small. Naturally, the specific choice of the function γ is problem dependent. One possible choice is to simply set $\gamma(\bar{y}, \bar{y}')$ to be 1 if $\bar{y} \neq \bar{y}'$ and 0 otherwise. This choice might lead to poor results in the alignment setting since it is likely that any function f will not find the exact correct alignment. For example, the above definition for γ will give the worst possible risk (1) to an alignment function that predicts correctly 99% of the alignment points, while such an alignment function is usually considered to perform very well.

VI. Efficient Evaluation of the Alignment Function

So far, we have put aside the problem of evaluation time of the function f given in (2). Recall that calculating f requires solving the following optimization problem:

$$f(\bar{\mathbf{x}}, \bar{e}) = \arg\max_{\bar{y}} \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{e}, \bar{y}).$$

Similarly, we need to find an efficient way for solving the maximization problem given in (7). A direct search for the maximizer is not feasible since the number of possible timing sequences, $\bar{y}$, is exponential in the number of events. Fortunately, as we show below, by imposing a few mild conditions on the structure of the alignment feature functions and on the cost function, γ, both problems can be solved in polynomial time.

We start with the problem of calculating the prediction given in (2). For simplicity, we assume that each base feature function, ϕ_j, can be decomposed as follows. Let ψ_j be any function from $\mathcal{X}^* \times E^* \times \mathbb{N}^3$ into the reals, which can be computed in a constant time. That is, ψ_j receives as input the signal, $\bar{\mathbf{x}}$, the sequence of events, $\bar{e}$, and three time points. Additionally, we use the convention $y_0 = 0$ and $y_{|\bar{e}|+1} = T + 1$. Using the above notation, we assume that each ϕ_j can be decomposed to be

$$\phi_j(\bar{\mathbf{x}}, \bar{e}, \bar{y}) = \sum_{i=1}^{|\bar{y}|} \psi_j(\bar{\mathbf{x}}, \bar{e}, y_{i-1}, y_i, y_{i+1}). \quad (13)$$

The base alignment functions we derive in later sections for speech-to-phoneme and music-to-score alignment can be decomposed as in (13).

We now describe an efficient algorithm for calculating the best timing sequence assuming that ϕ_j can be decomposed as in (13). Similar algorithms can be constructed for any base feature functions that can be described as a dynamic Bayesian network ([11], [18]). Given $i \in \{1, \ldots, |\bar{e}|\}$ and two time indices $t, t' \in \{1, \ldots, T\}$, denote by $D(i, t, t')$ the score for the prefix of the events sequence $1, \ldots, i$, assuming that their actual start times are $y_1, \ldots, y_i$, where $y_i = t'$ and assuming that $y_{i+1} = t$. This variable can be computed efficiently in a similar fashion to the forward variables calculated by the Viterbi procedure in HMMs (see for instance [19]). The pseudo code for computing $D(i, t, t')$ recursively is shown in Fig. 3. 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 $\mathcal{O}(|\bar{e}|\ |\bar{\mathbf{x}}|^3)$. However, in practice, we can use the assumption that the maximal length of an event is bounded, $t - t' \leq L$. This assumption reduces the complexity of the algorithm to be $\mathcal{O}(|\bar{e}|\ |\bar{\mathbf{x}}| L^2)$.

INPUT: audio signal $\bar{\mathbf{x}}$, sequence of events $\bar{e}$;
weight vector $\mathbf{w}$; maximal length of an event L

INITIALIZE: $\forall(1 \leq t \leq L),\ D(0,t,0) = 0$

RECURSION:

For $i = 1,\ldots,|\bar{e}|$

For $t = 1,\ldots,|\bar{\mathbf{x}}|$

For $t' = t-L,\ldots,t-1$

$$D(i,t,t') = \max_{t'-L \leq t'' < t'} D(i-1,t',t'') + \mathbf{w} \cdot \boldsymbol{\psi}(\bar{\mathbf{x}},\bar{e},t'',t',t)$$

TERMINATION: $D^\star = \max_{t'} D(|\bar{e}|,T,t')$

Fig. 3. Efficient procedure for evaluating the alignment function given in (2).

Solving the maximization problem given in (7) can be performed in a similar manner as we now briefly describe. Assume that $\gamma(\bar{y},\bar{y}')$ can be decomposed as follows:

$$\gamma(\bar{y},\bar{y}') = \sum_{i=1}^{|\bar{y}|} \hat{\gamma}\,(y_i,y_i')$$

where $\hat{\gamma}$ is any computable function. For example, for the definition of γ given in (1), we can set $\hat{\gamma}(y_i,y_i')$ to be zero if $|y_i - y_i'| \leq \epsilon$ and otherwise $\hat{\gamma}(y_i,y_i') = 1/|\bar{y}|$. A dynamic programming procedure for calculating (7) can be obtained from Fig. 3 by replacing the recursion definition of $D(i,t,t')$ to

$$D(i,t,t') = \max_{t'-L \leq t'' < t'} D(i-1,t',t'') + \hat{\gamma}(y_{i+1},t) + \mathbf{w} \cdot \boldsymbol{\psi}(\bar{\mathbf{x}},\bar{e},t'',t',t). \quad (14)$$

To conclude this section we discuss the global complexity of our proposed method. In the training phase, our algorithm performs m iterations, one iteration per each training example. At each iteration, the algorithm evaluates the alignment function once, updates the alignment function, if needed, and evaluates the new alignment function on a validation set of size m_v. Each evaluation of the alignment function takes an order of $\mathcal{O}(|\bar{e}|\ |\bar{\mathbf{x}}|L^2)$ operations. Therefore the total complexity of our method becomes $\mathcal{O}(mm_v|\bar{e}|\ |\bar{\mathbf{x}}|L^2)$. In practice, however, we can evaluate the updated alignment function only for the last 50 iterations or so, which reduces the global complexity of the algorithm to $\mathcal{O}(m|\bar{e}|\ |\bar{\mathbf{x}}|L^2)$. In all of our experiments, evaluating the alignment function only for the last 50 iterations was found empirically to give sufficient results. Finally, we compare the complexity of our method to the complexity of other algorithms which directly solve the SVM optimization problem given in (6). The algorithm given in [11] is based on the SMO algorithm for solving SVM problems. While there is no direct complexity analysis for this algorithm, in practice it usually required at least m^2 iterations which results in a total complexity of the order $\mathcal{O}(m^2|\bar{e}|\ |\bar{\mathbf{x}}|L^2)$. The complexity of the algorithm presented in [13] depends on the choice of several parameters. For reasonable choice of these parameters, the total complexity is also of the order $\mathcal{O}(m^2|\bar{e}|\ |\bar{\mathbf{x}}|L^2)$.

VII. Speech-to-Phoneme Alignment

In this section, we present the implementation details of our learning approach for the task of speech-to-phoneme alignment. Recall that our construction is based on a set of base alignment functions, $\{\phi_j\}_{j=1}^n$, which maps an acoustic-phonetic representation of a speech utterance as well as a suggested phoneme start time sequence into an abstract vector-space. All of our base alignment functions are decomposable as in (13) and therefore it suffices to describe the functions $\{\psi_j\}$. We start the section by introducing a specific set of base functions, which is highly adequate for the speech-to-phoneme alignment problem. Next, we report experimental results comparing our algorithm to alternative state-of-the-art approaches.

A. Base Alignment Functions

We utilize seven different base alignment functions $(n = 7)$. These base functions are used for defining our alignment function $f(\bar{\mathbf{x}},\bar{e})$ as in (2).

Our first four base functions aim at capturing transitions between phonemes. These base functions are based on the distance between frames of the acoustical signal at two sides of phoneme boundaries as suggested by a phoneme start time sequence $\bar{y}$. The distance measure we employ, denoted by d, is the Euclidean distance between feature vectors. Our underlying assumption is that if two frames, $\mathbf{x}_t$ and $\mathbf{x}_{t'}$, are derived from the same phoneme then the distance $d(\mathbf{x}_t,\mathbf{x}_{t'})$ should be smaller than if the two frames are derived from different phonemes. Formally, our first four base functions are defined as

$$\psi_j(\bar{\mathbf{x}},\bar{e},y_{i-1},y_i,y_{i+1}) = d\,(\mathbf{x}_{y_i-j},\mathbf{x}_{y_i+j}), \quad j \in \{1,2,3,4\}. \quad (15)$$

If $\bar{y}$ is the correct timing sequence then distances between frames across the phoneme change points are likely to be large. In contrast, an incorrect phoneme start time sequence is likely to compare frames from the same phoneme, often resulting small distances. Note that the first four base functions described above only use the start time of the ith phoneme and does not use the values of y_{i-1} and y_{i+1}.

The fifth base function we use is based on the framewise phoneme classifier described in [20]. Formally, for each phoneme event $e \in \mathcal{P}$ and frame $\mathbf{x} \in \mathcal{X}$, there is a confidence, denoted $g_e(\mathbf{x})$, that the phoneme e is pronounced in the frame $\mathbf{x}$. The resulting base function measures the cumulative confidence of the complete speech signal given the phoneme sequence and their start-times

$$\psi_5(\bar{\mathbf{x}},\bar{e},y_{i-1},y_i,y_{i+1}) = \sum_{t=y_i}^{y_{i+1}-1} g_{e_i}(\mathbf{x}_t). \quad (16)$$

The fifth base function use both the start time of the ith phoneme and the $(i+1)$th phoneme but ignores y_{i-1}.

Our next base function scores timing sequences based on phoneme durations. Unlike the previous base functions, the sixth base function is oblivious to the speech signal itself. It merely examines the length of each phoneme, as suggested by

$\bar{y}$, compared to the typical length required to pronounce this phoneme. Formally

$$\psi_6(\bar{\mathbf{x}}, \bar{e}, y_{i-1}, y_i, y_{i+1}) = \log \mathcal{N}\left(y_{i+1} - y_i; \hat{\mu}_{e_i}, \hat{\sigma}_{e_i}\right) \quad (17)$$

where $\mathcal{N}$ is a Normal probability density function with mean $\hat{\mu}_e$ and standard deviation $\hat{\sigma}_e$. In our experiments, we estimated $\hat{\mu}_e$ and $\hat{\sigma}_e$ from the entire TIMIT training set, excluding SA1 and SA2 utterances.

Our last base function exploits assumptions on the speaking rate of a speaker. Intuitively, people usually speak in an almost steady rate and therefore a timing sequence in which speech rate is changed abruptly is probably incorrect. Formally, let $\hat{\mu}_e$ be the average length required to pronounce the eth phoneme. We denote by r_i the relative speech rate, $r_i = (y_{i+1} - y_i)/\hat{\mu}_e$. That is, r_i is the ratio between the actual length of phoneme e_i as suggested by $\bar{y}$ to its average length. The relative speech rate presumably changes slowly over time. In practice the speaking rate ratios often differ from speaker to speaker and within a given utterance. We measure the local change in the speaking rate as $(r_i - r_{i-1})^2$ and we define the base function ψ_7 as the local change in the speaking rate

$$\psi_7(\bar{\mathbf{x}}, \bar{e}, y_{i-1}, y_i, y_{i+1}) = (r_i - r_{i-1})^2. \quad (18)$$

Note that ψ_7 relies on all three start-times it receives as an input, y_{i-1}, y_i, y_{i+1}.

B. Experiments

To validate the effectiveness of the proposed approach, we performed experiments with the TIMIT corpus. We first divided the training portion of TIMIT (excluding the SA1 and SA2 utterances) into three disjoint parts containing 500, 100, and 3093 utterances, respectively. The first part of the training set was used for learning the functions g_{e_i} [(16)], which defines the base function ψ_5. Those functions were learned by the algorithm described in [20] using the $\mathrm{MFCC} + \Delta + \Delta\Delta$ acoustic features [21] and a Gaussian kernel ($\sigma = 6.24$ and $C = 5.0$). The second set of 100 utterances formed the validation set needed for our alignment algorithm as described in Section V. Finally, we ran our iterative alignment algorithm on the remaining utterances in the training set. The value of ϵ in the definition of γ was set to be 1 (i.e., 10 ms).

We evaluated the learned alignment functions on both the core test set and the entire test set of TIMIT. We compare our results to the results reported by Brugnara *et al.* [1] and the results obtained by Hosom [2]. The results are summarized in Table I. For each tolerance value $\tau \in \{10\ \mathrm{ms}, 20\ \mathrm{ms}, 30\ \mathrm{ms}, 40\ \mathrm{ms}\}$, we counted the number of predictions whose distance to the true boundary, $t = |y_i - y_i'|$, is less than τ. As can be seen in the table our discriminative large margin algorithm is comparable to the best results reported on TIMIT. Furthermore, we found out in our experiments that the same level of accuracy is obtained when using merely the first 50 utterances (rather than the entire 3093 utterances that are available for training).

Our alignment function is based on the weight vector $\mathbf{w}$ that determines the linear combination of base alignment functions. It is therefore interesting to observe the specific weights $\mathbf{w}$ gives to each of the base alignment functions after the training phase. Since the training algorithm depends on the order of examples in the training set, we ran the algorithm on several random permutations of the same training set and average the resulting weight vector. The resulting vector was found to be

$$\mathbf{w} = (0.17788, 0.0093, -2.82 \times 10^{-5}, 0.0087, 0.9622, 1.41 \times 10^{-5}, 0.15815). \quad (19)$$

We also calculated the standard deviation of each element of $\mathbf{w}$. The standard deviation was found to be almost 0 for all the elements of the weight vector, indicating that our training algorithm is rather stable and the resulting weight vector does not depend on the order of examples in the training set. It is also apparent that the weight of the fifth base alignment function is dominant. To remind the reader, the fifth base alignment function corresponds to the frame-wise phoneme classifier. The domination of this feature calls for a comparison between the performance of each single feature to the performance of our method that combined together all the features. In Table II, we report the performance of each of the single base alignment features. We again see that the fifth base alignment function is most effective. The accuracy of this single feature is inferior to the accuracy of our combined method roughly by 3%. When using an alternative single-base alignment function, we obtain rather poor results. The advantage of our method is that it combines the features together to obtain the best performing alignment function.

TABLE I
PERCENTAGE OF CORRECTLY POSITIONED PHONEME BOUNDARIES, GIVEN A PREDEFINED TOLERANCE ON THE TIMIT CORPUS

	$t \leq 10$ms	$t \leq 20$ms	$t \leq 30$ms	$t \leq 40$ms
TIMIT core test-set				
Discrim. Alignment	**79.7**	92.1	**96.2**	**98.1**
Brugnara *et al.* [1]	75.3	88.9	94.4	97.1
Hosom [2]		**92.57**		
TIMIT entire test-set				
Discrim. Alignment	**80.0**	**92.3**	**96.4**	**98.2**
Brugnara *et al.* [1]	74.6	88.8	94.1	96.8

TABLE II
PERCENTAGE OF CORRECTLY POSITIONED PHONEME BOUNDARIES FOR EACH ELEMENT OF THE VECTOR $\mathbf{w}$

	$t \leq 10$ms	$t \leq 20$ms	$t \leq 30$ms	$t \leq 40$ms
w_1	7.6	9.9	12.5	15.1
w_2	8.3	11.8	15.2	18.7
w_3	8.2	12.3	15.9	19.2
w_4	6.7	9.4	12.0	14.9
w_5	77.0	88.8	93.6	95.5
w_6	12.6	19.2	26.2	33.1
w_7	12.2	18.3	24.9	31.3
$\mathbf{w}$	79.7	92.1	96.2	98.1

VIII. MUSIC-TO-SCORE ALIGNMENT

In this section, we present the implementation details of our learning approach for the task of music-to-score alignment. We

start the section by introducing a specific set of base alignment functions which is highly adequate for the music-to-score alignment problem. Next, we report experimental results comparing our algorithm to an alternative generative method for score alignment.

A. Base Alignment Functions

We utilize ten different base alignment functions ($n = 10$). Recall that each note-on event in the music-to-score alignment problem is a pair $e = (p, s)$, where p is the pitch of the note and s is the (theoretical) start time of the note. Our first nine base alignment functions ignore the value of s and thus, for these features, ψ_j only depends on $\bar{\mathbf{x}}$, $\bar{p}$, and $\bar{y}$. Intuitively, the first nine base alignment functions measure 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 base functions, starting with ψ_1. The function $\psi_1(\bar{\mathbf{x}}, \bar{e}, y_{i-1}, y_i, y_{i+1})$ measures the energy of the acoustic signal at the frame $\mathbf{x}_{y_i}$ and the frequency corresponding to the pitch p_i. Formally, let F_{p_i} denotes a band-pass filter with a center frequency at the first harmony of the pitch p_i and cutoff frequencies of 1/4 tone below and above p_i. Concretely, the lower cutoff frequency of F_{p_i} is $440 \cdot 2^{(p_i - 57 - 0.5)/12}$ Hz and the upper cutoff frequency is $440 \cdot 2^{(p_i - 57 + 0.5)/12}$ Hz, where $p_i \in \mathcal{P} = \{0, 1, \ldots, 127\}$ is the pitch value (coded using the MIDI standard) and $440 \cdot 2^{(p_i - 57)/12}$ is the frequency value in Hz associated with the codeword p_i. Similarly, ψ_2 and ψ_3 are the output energies of bandpass filters centered at the second and third pitch harmonics, respectively. All the filters were implemented using the fast Fourier transform.

The above three local templates $\{\psi_j\}_{j=1}^{3}$ measure energy values for each time y_i. Since we are interested in identifying notes onset times, it is reasonable to compare energy values at time y_i with energy values at time $y_i - 1$. 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 [19].) Therefore, the next six local templates, $\{\psi_j\}_{j=4}^{9}$, measure the first and second derivatives of the energy of the output signal of the three filters around the first three harmonics of p_i.

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

$$r_i = \frac{y_{i+1} - y_i}{s_{i+1} - s_i} \tag{20}$$

be the ratio between the ith interval, according to $\bar{y}$, to the interval according to $\bar{s}$. We also refer to r_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 ψ_{10} measures the local change in the tempo

$$\psi_{10}(\bar{\mathbf{x}}, \bar{e}, y_{i-1}, y_i, y_{i+1}) = (r_i - r_{i-1})^2.$$

The relative tempo of (20) is ill-defined whenever $s_{i+1} - s_i$ 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 r_i as in (20) but confine ourselves to indices i for which $s_{i+1} - s_i$ is greater than a predefined value τ (in our experiments we set $\tau = 60$ ms). Thus, if $s_{i+1} - s_i \leq \tau$ or $s_i - s_{i-1} \leq \tau$, then we set psi_{10} to be zero.

B. Experiments

We now describe experiments with our alignment algorithm for the task of score alignment of polyphonic piano musical pieces. Specifically, we compare our alignment algorithm to a generative method which is based on a generalized HMM (GHMM). The details of the GHMM approach can be found in [22]. This GHMM method is closely related to graphical model approaches for score alignment, first proposed by Raphael [23], [24]. We would like to note in passing that more advanced algorithms for real-time score alignment have been suggested (see, for example, [25] and the references therein). In our experiments we focus on the basic comparison between the discriminative and generative approaches for score alignment. Recall that our alignment algorithm uses a training set of 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 web-sites and used these MIDI files for creating the sequence of events $\bar{e}$. The complete dataset we used is available from www.cs.huji.ac.il/shais/alignment.

In the score alignment problem we report the average alignment error, that is, we set

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

Since this dataset is rather small, we ran our iterative algorithm given in Fig. 2 on the training set several times and choose the alignment function which minimizes the error on the training set. 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 error between the predicted and true start times is computed for each of the algorithms. The GHMM approach uses a Gaussian Mixture Model (GMM) for modeling 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 III. As seen from the table, our discriminative learning algorithm outperforms all the variants of the

TABLE III
SUMMARY OF THE LOO LOSS (IN MS) FOR DIFFERENT ALGORITHMS FOR MUSIC-TO-SCORE ALIGNMENT

	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**

GHMM method in all of the experiments. Moreover, in all but two of the experiments the error 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.

IX. CONCLUSION

We presented a discriminative algorithm for learning an alignment function from a training set of examples. The proposed approach is based on recent advances in large margin classifiers. The contribution of our algorithm is twofold. First, we showed how the tasks of speech-to-phoneme alignment and music-to-score alignment can be cast as large margin problems. Second, we presented a simple and effective algorithm for solving the induced large margin problem. Our learning algorithm is more efficient than similar algorithms for large margin sequence prediction, such as [11] and [13], and is thus more adequate for speech and audio applications, in which we typically have a large number of training examples. Our learning algorithm is simple to implement and entertains convergence guarantees. Moreover, we have shown both theoretical and empirical evidence demonstrating the generalization abilities of our method. Indeed, the experiments reported above suggest that the discriminative training requires fewer training examples than an HMM-based speech-to-phoneme alignment procedure. We are currently investigating generalizations of the framework to more demanding tasks in which a sequence constituents should also be predicted, as is the case in phoneme recognition and music transcription.

APPENDIX

Proof of Thm. 1

Our proof relies on [15, Theor. 2]. We first construct a sequence of binary classification examples, $(\Delta\boldsymbol{\phi}_1, +1), \ldots, (\Delta\boldsymbol{\phi}_m, +1)$. For all i and for all $\mathbf{w} \in \mathbb{R}^n$, define the following classification hinge-loss:

$$\ell_i^c(\mathbf{w}) = \max\left\{\gamma\left(\bar{y}_i, \bar{y}_i'\right) - \mathbf{w} \cdot \Delta\boldsymbol{\phi}_i, 0\right\}.$$

The implication in [15, Theor. 2] is that the following bound holds for all $\mathbf{w} \in \mathbb{R}^n$:

$$\sum_{i=1}^{m} \mu\left(\ell_i^c(\mathbf{w}_i)\right) \leq \frac{1}{C}\|\mathbf{w}\|^2 + \sum_{i=1}^{m} \ell_i^c(\mathbf{w}) \tag{21}$$

where

$$\mu(a) = \frac{1}{C}\left(\min\{a, C\}\left(a - \frac{1}{2}\min\{a, C\}\right)\right).$$

Let $\mathbf{w}^\star$ denote the optimum of the alignment problem given by (6). The bound of (21) holds for any $\mathbf{w}$ and in particular for the optimal solution $\mathbf{w}^\star$. Furthermore, the definition of ℓ_i^c implies that $\ell_i^c(\mathbf{w}^\star) \leq \ell(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i))$ and $\ell_i^c(\mathbf{w}_i) = \ell(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i))$ for all i. Using the latter two facts in (21) gives that

$$\sum_{i=1}^{m} \mu\left(\ell\left(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right)\right) \leq \frac{1}{C}\|\mathbf{w}^\star\|^2 + \sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right). \tag{22}$$

By definition, the function μ is bounded below by a linear function, that is, for any $a > 0$

$$\mu(a) \geq a - \frac{1}{2}C.$$

Using the lower bound with the argument $\ell(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i))$ and summing over i, we obtain

$$\sum_{i=1}^{m} \ell\left(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) - \frac{1}{2}Cm \leq \sum_{i=1}^{m} \mu\left(\ell\left(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right)\right).$$

Combining the above inequality with (22) and rearranging terms gives the bound stated in the theorem and concludes our proof.■

Proof of Thm 2

Denote by $f_1, \ldots, f_m$ the alignment prediction functions corresponding to the weight vectors $\mathbf{w}_1, \ldots, \mathbf{w}_m$ that are found by the alignment algorithm. Proposition 1 in [17] implies that with probability of at least $1 - \delta_1$ the following bound holds:

$$\frac{1}{m}\sum_{i=1}^{m} \text{risk}(f_i) \leq \frac{1}{m}\sum_{i=1}^{m} \gamma\left(\bar{y}_i, f_i(\bar{\mathbf{x}}_i, \bar{e}_i)\right) + \frac{\sqrt{2\ln(1/\delta_1)}}{\sqrt{m}}.$$

By definition, the hinge-loss $\ell(\mathbf{w}_i; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i))$ bounds from above the loss $\gamma(\bar{y}_i, f_i(\bar{\mathbf{x}}_i, \bar{e}_i))$. Combining this fact with Theorem 1, we obtain that

$$\frac{1}{m}\sum_{i=1}^{m} \text{risk}(f_i) \leq \frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) + \frac{\|\mathbf{w}^\star\|^2 + \frac{1}{2} + \sqrt{2\ln(1/\delta_1)}}{\sqrt{m}}. \tag{23}$$

The left-hand side of the above inequality upper bounds $\text{risk}(f_b)$, where $b = \arg\min_i \text{risk}(f_i)$. Therefore, among the finite set of alignment functions, $F = \{f_1, \ldots, f_m\}$, there exists at least one alignment function (for instance the function f_b) whose true risk is bounded above by the right hand side of

(23). Recall that the output of our algorithm is the alignment function $f_{\mathbf{w}} \in F$, which minimizes the average cost over the validation set S_v. Applying Hoeffding inequality together with the union bound over F, we conclude that with probability of at least $1 - \delta_2$

$$\text{risk}(f_{\mathbf{w}}) \leq \text{risk}(f_b) + \sqrt{\frac{2\ln(m/\delta_2)}{m_v}}$$

where, to remind the reader, $m_v = |S_v|$. We have therefore shown that with probability of at least $1-\delta_1-\delta_2$, the following inequality holds:

$$\text{risk}(f_{\mathbf{w}}) \leq \frac{1}{m}\sum_{i=1}^{m} \ell\left(\mathbf{w}^\star; (\bar{\mathbf{x}}_i, \bar{e}_i, \bar{y}_i)\right) + \frac{||\mathbf{w}^\star||^2 + \frac{1}{2} + \sqrt{2\ln(1/\delta_1)}}{\sqrt{m}} + \frac{\sqrt{2\ln(m/\delta_2)}}{\sqrt{m_v}}.$$

Setting $\delta_1 = \delta_2 = \delta/2$ concludes our proof. ■

REFERENCES

[1] F. Brugnara, D. Falavigna, and M. Omologo, "Automatic segmentation and labeling of speech based on hidden Markov models," *Speech Commun.*, vol. 12, pp. 357–370, 1993.

[2] J.-P. Hosom, "Automatic phoneme alignment based on acoustic-phonetic modeling," in *Proc. 7th Int. Conf. Spoken Language Processing*, 2002, pp. 357–360.

[3] D. Toledano, L. Gomez, and L. Grande, "Automatic phoneme segmentation," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 6, pp. 617–625, Nov. 2003.

[4] S. Shalev-Shwartz, S. Dubnov, N. Friedman, and Y. Singer, "Robust temporal and spectral modeling for query by melody," in *Proc. 25th Annual Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Tampere, Finland, Aug. 2002.

[5] F. Soulez, X. Rodet, and D. Schwarz, "Improving polyphonic and poly-instrumental music to score alignment," in *Proc. Int. Symp. Music Information Retrieval*, 2003.

[6] V. N. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.

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

[8] J. Salomon, S. King, and M. Osborne, "Framewise phone classification using support vector machines," in *Proc. 7th Int. Conf. Spoken Language Processing*, 2002, pp. 2645–2648.

[9] J. Keshet, D. Chazan, and B.-Z. Bobrovsky, "Plosive spotting with margin classifiers," in *Proc. 7th European Conf. Speech Communication and Technology*, 2001, pp. 1637–1640.

[10] Z. Litichever and D. Chazan, "Classification of transition sounds with application to automatic speech recognition," in *EUROSPEECH*, Aalborg, Denmark, 2001.

[11] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *Advances in Neural Information Processing Systems 17*, 2003.

[12] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *Proc. Conf. Empirical Methods in Natural Language Processing*, Philadelphia, PA, 2002.

[13] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *Proc. 21st Int. Conf. Machine Learning*, Banff, AB, Canada, Jul. 2004.

[14] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, Mar. 2006.

[15] S. Shalev-Shwartz and Y. Singer, "Online learning meets optimization in the dual," in *Proc. 19th Annu. Conf. Computational Learning Theory*, Pittsburgh, PA, 2006.

[16] K.-F. Lee and H.-W. Hon, "Speaker independent phone recognition using hidden Markov models," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 11, pp. 1641–1648, Nov. 1989.

[17] N. Cesa-Bianchi, A. Conconi, and C. Gentile, "On the generalization ability of on-line learning algorithms," *IEEE Trans. Inf. Theory*, vol. 50, no. 9, pp. 2050–2057, Sep. 2004.

[18] T. Dean and K. Kanazawa, "A model for reasoning about persistence and causation," *Comput. Intell.*, vol. 5, no. 3, pp. 142–150, 1989.

[19] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[20] O. Dekel, J. Keshet, and Y. Singer, "Online algorithm for hierarchical phoneme classification," in *Workshop on Multimodal Interaction and Related Machine Learning Algorithms; Lecture Notes in Computer Science*, 2004, pp. 146–159, Springer-Verlag.

[21] ETSI Standard, ETSI ES 201 108 2000.

[22] S. Shalev-Shwartz, J. Keshet, and Y. Singer, "Learning to align polyphonic music," in *Proc. 5th Int. Conf. Music Information Retrieval*, Barcelona, Spain, Oct. 2004.

[23] C. Raphael, "Automatic segmentation of acoustic musical signals using hidden markov models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 4, pp. 360–370, Apr. 1999.

[24] C. Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores," in *Proc. 5th Int. Conf. Music Information Retrieval*, Barcelona, Spain, Oct. 2004.

[25] A. Cont, "Realtime audio to score alignment for polyphonic music instruments using sparse non-negative constraints and hierarchical HMMs," in *Proc. IEEE Int. Conf. Acoustics and Speech Signal Processing*, Toulouse, France, May 2006.

Joseph Keshet received the B.Sc. and M.Sc. degrees in electrical engineering in 1994 and 2002, respectively, from Tel-Aviv University, Tel-Aviv, Israel. He is currently pursuing the Ph.D. degree in computer science at The Hebrew University of Jerusalem, Jerusalem, Israel, under the supervision of Prof. Y. Singer.

From 1994 to 2002, he was with the Israeli Defense Forces (Intelligence Corps), where he was in charge of advanced research activities in the fields of speech coding. His research interests are in the areas of speech recognition, speech processing, machine learning algorithms, and statistical signal processing.

Shai Shalev-Shwartz received the B.Sc. degree in computer science from the Open University of Israel in 1999 and the M.Sc. degree in computer science from The Hebrew University, Jerusalem, Israel, in 2002, where he is currently pursuing the Ph.D. degree in computer science under the supervision of Prof. Y. Singer.

His research interests are in the areas of machine learning algorithms, online learning and games, optimization techniques, and signal processing applications.

Yoram Singer received the Ph.D. degree in computer science from The Hebrew University of Jerusalem, Jerusalem, Israel, in 1995.

He is a Senior Research Scientist at Google. From 1999 to 2007, he was an Associate Professor of Computer Science and Engineering at The Hebrew University of Jerusalem. From 1995 to 1999, he was a Member of Technical Staff at AT&T Research. His current research focuses on the design, analysis, and implementation of machine learning algorithms.

Dan Chazan received the B.S., M.S., and Ph.D. degrees from the University of California at Berkeley in 1961, 1963, and 1965, respectively.

Most of his professional career was spent as a Researcher at IBM, where he worked in varied fields including numerical analysis, operations research, numerical simulation, econometric modelling, and during the last 20 years, speech signal processing, where he mentors students for advanced degrees and carries out research in speech coding and speech synthesis and modelling. Currently, he is retired from IBM and is involved with projects at The Technion, Haifa, Israel, and personal research interests.