

Metric Learning for Temporal Sequence Alignment

Damien Garreau, Rémi Lajugie, Sylvain Arlot, Francis Bach

▶ **To cite this version:**

Damien Garreau, Rémi Lajugie, Sylvain Arlot, Francis Bach. Metric Learning for Temporal Sequence Alignment. Advances in Neural Information Processing Systems 27 (NIPS 2014), Dec 2014, Montréal, Canada. hal-01062130

HAL Id: hal-01062130
https://inria.hal.science/hal-01062130

Submitted on 10 Sep 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Metric Learning for Temporal Sequence Alignment

Damien Garreau *
damien.garreau@ens.fr[1,2],
Rémi Lajugie *
remi.lajugie@inria.fr[1,2],
Sylvain Arlot
sylvain.arlot@ens.fr[1,2], and
Francis Bach
francis.bach@inria.fr[1,2]

[1]Département d'Informatique de l'Ecole Normale Supérieure
[2]SIERRA project team, Inria Paris Rocquencourt

September 10, 2014

Abstract

In this paper, we propose to learn a Mahalanobis distance to perform alignment of multivariate time series. The learning examples for this task are time series for which the true alignment is known. We cast the alignment problem as a structured prediction task, and propose realistic losses between alignments for which the optimization is tractable. We provide experiments on real data in the audio to audio context, where we show that the learning of a similarity measure leads to improvements in the performance of the alignment task. We also propose to use this metric learning framework to perform feature selection and, from basic audio features, build a combination of these with better performance for the alignment.

1 Introduction

The problem of aligning temporal sequences is ubiquitous in applications ranging from bioinformatics [6, 1, 23] to audio processing [5, 7]. The idea is to align two similar time series that have the same global structure but local temporal differences. Most alignments algorithms rely on similarity measures, and having a good metric is crucial, especially in the high-dimensional setting where some features of the signals can be

*Both authors contributed equally.

irrelevant to the alignment task. The goal of this paper is to show how to learn this similarity measure from annotated examples in order to improve the precision of the alignments.
For example, in the context of music information retrieval, alignment is used in two different cases: (1) audio-to-audio alignment and (2) audio-to-score alignment. In the first case, the goal is to match two audio interpretations of the same piece that are potentially different in rythm, whereas audio-to-score alignment focuses on matching an audio signal to a symbolic representation of the score. In the second case, there are some attempts to learn from annotated data a measure for performing the alignment. Joder et al. [13] propose to fit a generative model in that context, and Keshet et al. [14] learn this measure in a discriminative setting.
Similarly to Keshet et al. [14], we use a discriminative loss to learn the measure, but our work focuses on audio-to-audio alignment. In that context, the set of authorized alignments is much larger, and we explicitly cast the problem as a structured prediction task, that we solve using off-the-shelf stochastic optimization techniques [16] but with proper and significant adjustements, in particular in terms of losses.
The need for metric learning goes far beyond unsupervised partitioning problems. Weinberger and Saul [26] proposed a large-margin framework for learning a metric in nearest-neighbour algorithms based on sets of must-link/must-not-link constraints. Lajugie et al. [17] proposed to use a large margin framework to learn a Mahalanobis metric in the context of partitioning problems. Since structured SVM have been proposed by Tsochantaridis et al. [25], Taskar et al. [22], they have successfully been used to solve many learning problems, for instance to learn weights for graph matching [4] or a metric for ranking tasks [18]. They have also been used to learn graph structures using graph cuts [21].

Contributions. We make the following five contributions: (1) we cast the learning of a Mahalanobis metric in the context of alignment as a structured prediction problem, (2) we show that on real musical datasets this metric improves the performance of alignment algorithms using high-level features, (3) we propose to use the metric learning framework to learn combinations of basic audio features and get good alignment performances, (4) we show experimentally that the standard Hamming loss, although tractable computationnally does not permit to learn a relevant similarity measure in some real world settings, (5) we propose a new loss, closer to the true evaluation loss for alignments, leading to a tractable learning task, and derive an efficient Frank-Wolfe based algorithm to deal with this new loss.

2 Matricial formulation of alignment problems

2.1 Notations

In this paper, we consider the alignment problem between two multivariate time series sharing the same dimension p, but possibly of different lengths T_A and T_B, namely $A \in \mathbb{R}^{T_A \times p}$ and $B \in \mathbb{R}^{T_B \times p}$. We refer to the rows of A as $a_1, \dots, a_{T_A} \in \mathbb{R}^p$ and

those of B as $b_1, \dots, b_{T_B} \in \mathbb{R}^p$. From now on, we denote by X the pair of signals (A, B).
Let $C(X) \in \mathbb{R}^{T_A \times T_B}$ be an arbitrary pairwise *affinity matrix* associated to the pair X, that is, $C(X)_{i,j}$ encodes the affinity between a_i and b_j. Note our framework can be extended to the case where A and B are multivariate signals of different dimensions, as long as $C(X)$ is well-defined. The goal of the alignment task is to find two non-decreasing sequences of indices α and β of same length $u \geq \max(T_A, T_B)$ and to match each time index $\alpha(i)$ in time series A to the time index $\beta(i)$ in the time series B, in such a way that $\sum_{i=1}^{u} C(X)_{\alpha(i),\beta(i)}$ is maximal, and that (α, β) satisfies:

$$\begin{cases} \alpha(1) = \beta(1) = 1 & \text{(matching beginning)} \\ \alpha(u) = T_A,\ \beta(u) = T_B & \text{(matching ending)} \\ \forall i,\ (\alpha(i+1), \beta(i+1)) - (\alpha(i), \beta(i)) \in \{(1,0), (0,1), (1,1)\} & \text{(three type of moves)} \end{cases} \tag{1}$$

For a given (α, β), we define the binary matrix $Y \in \{0,1\}^{T_A \times T_B}$ such that $Y_{\alpha(i),\beta(i)} = 1$ for every $i \in \{1, \dots, u\}$ and 0 otherwise. We denote by $\mathcal{Y}(X)$ the set of such matrices, which is uniquely determined by T_A and T_B. An example is given in Fig. 1. A vertical move in the Y matrix means that the signal B is waiting for A and an horizontal one that A is waiting for B. In this sense we can say that the time reference is "warped".
When $C(X)$ is known, the alignment task can be cast as the following linear program (LP) over the set $\mathcal{Y}(X)$:

$$\max_{Y \in \mathcal{Y}(X)} \operatorname{Tr}(C(X)^\top Y). \tag{2}$$

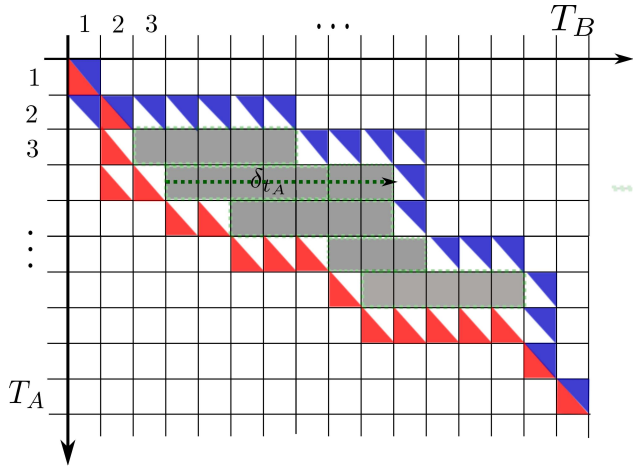
Our goal is to learn how to form the affinity matrix: once we have learned $C(X)$, the alignment is obtained from Eq. (2). The optimization problem in Eq. (2) will be referred to as the *decoding* of our model.

Dynamic time warping. Given the affinity matrix $C(X)$ associated with the pair of signals $X = (A, B)$, finding the alignment that solves the LP of Eq. (2) can be done efficiently in $O(T_A T_B)$ using a dynamic programming algorithm. It is often referred to as dynamic time warping [6, 19]. This algorithm is described in Alg. 1 of the supplementary material. Various additional constraints may be used in the dynamic time warping algorithm [19], which we could easily add.
The cardinality of the set $\mathcal{Y}(X)$ is huge: it corresponds to the number of paths on a rectangular grid from the southwest $(1,1)$ to the northeast corner (T_A, T_B) with only vertical, horizontal and diagonal moves allowed. This is the definition of the Delannoy numbers [2]. As noted in [24], when $t = T_A = T_B$ is big, one can show that $\#\mathcal{Y}_{t,s} \sim \frac{(3+2\sqrt{2})^t}{\sqrt{\pi t}\sqrt{3\sqrt{2}-4}}$.

2.2 The Mahalanobis metric

In many applications, for a pair $X = (A, B)$, the affinity matrix is computed by $C(A, B)_{i,j} = -\|a_{i,k} - b_{j,k}\|^2$. In this paper we propose to learn the metric to compare

Figure 1: Example of two valid alignments encoded by matrices Y^1 and Y^2. Red upper triangles show the (i, j) such that $Y^1_{i,j} = 1$, and the blue lower ones show the (i, j) such that $Y^2_{i,j} = 1$. The grey zone corresponds to the area loss δ_{abs} between Y^1 and Y^2, whereas the $\delta_{\max}$ loss corresponds to the maximum of the δ_t, $t \in \{1, \dots, T_1\}$.



a_i and b_j instead of using the plain Euclidean metric. That is, $C(X)$ is parametrized by a matrix $W \in \mathcal{W} \subset \mathbb{R}^{p \times p}$, where $\mathcal{W} \subset \mathbb{R}^{p \times p}$ is the set of semi-definite positive matrices, and we use the corresponding Mahalanobis metric to compute the pairwise affinity between a_i and b_j: $C(X; W)_{i,j} = -(a_i - b_j)^\top W (a_i - b_j)$.
Note that the decoding of Eq. (2) is the maximization of a linear function in the parameter W:

$$\max_{Y \in \mathcal{Y}(X)} \operatorname{Tr}(C(X; W)^\top Y) \Leftrightarrow \max_{Y \in \mathcal{Y}(X)} \operatorname{Tr}(W^\top \phi(X, Y)), \tag{3}$$

if we define the joint feature map $\phi(X, Y) = -\sum_{i,j} Y_{i,j}(a_i - b_j)(a_i - b_j)^\top \in \mathbb{R}^{p \times p}$.

3 Learning the metric

From now on, we assume that we are given n pairs of training instances[1] $(X^i, Y^i) = ((A^i, B^i), Y^i) \in \mathbb{R}^{T^i_A \times p} \times \mathbb{R}^{T^i_B \times p} \times \mathbb{R}^{T^i_A \times T^i_B}$, $i = 1, \dots, n$. Our goal is to find a matrix W such that the predicted alignments are close to the groundtruth on these examples, as well as on unseen examples. We first define a *loss* between alignments, in order to quantify this proximity between alignments.

[1]We will see that it is necessary to have fully labelled instances, which means that for each pair X^i we need an *exact* alignment Y^i between A^i and B^i. Partial alignment might be dealt with by alternating between metric learning and constrained alignment.

3.1 Losses between alignments

In our framework, the alignments are encoded by matrices in $\mathcal{Y}(X)$, thus we are interested in functions $\ell : \mathcal{Y}(X) \times \mathcal{Y}(X) \to \mathbb{R}_+$. Let us define the Frobenius norm by $\|M\|_F^2 = \sum_{i,j} M_{i,j}^2$.

Hamming loss. A simple loss between matrices is the Frobenius norm of their difference, which turns out to be the unnormalized Hamming loss [10] for $0/1$ valued matrices. For two matrices $Y_1, Y_2 \in \mathcal{Y}(X)$, it is defined as:

$$\begin{aligned} \ell_H(Y_1, Y_2) &= \|Y_1 - Y_2\|_F^2 \\ &= \operatorname{Tr}(Y_1^\top Y_1) + \operatorname{Tr}(Y_2^\top Y_2) - 2\operatorname{Tr}(Y_1^\top Y_2) \\ &= \operatorname{Tr}(Y_1 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) + \operatorname{Tr}(Y_2 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) - 2\operatorname{Tr}(Y_1^\top Y_2), \end{aligned} \tag{4}$$

where $\mathbf{1}_T$ is the vector of $\mathbb{R}^T$ with all coordinates equal to 1. The last line of Eq. (4) comes from the fact that the Y_i have $0-1$ values; that makes the Hamming loss linear in Y_1 and Y_2. This loss is often used in other structured prediction tasks [16]. In the audio-to-score setting, Keshet et al. [14] use a modified version of this loss, which is the average number of times the difference between the two alignments is greater than a fixed threshold.
This loss is easy to optimize since it is linear in our parametrization of the alignement problem, but not optimal for audio-to-audio alignment. Indeed, a major drawback of the Hamming loss is, for alignments of fixed length, it depends only on the number of "crossings" between alignment paths: one can easily find Y_1, Y_2, Y_3 such that $\ell_H(Y_2, Y_1) = \ell_H(Y_3, Y_1)$ but Y_2 is intuitively much closer to Y_1 than Y_3 (see Fig. 2). It is important to notice this is often the case as the length of the signals grows.

Area loss. A more natural loss can be computed as the mean distance beween the paths depicted by two matrices $Y^1, Y^2 \in \mathcal{Y}(X)$. This loss is represented by the grey zone on Fig. 1 and corresponds to the area between the paths of two matrices Y.
Formally, as in Fig. 1, for each $t \in \{1, \dots, T_B\}$ we define δ_t as the minimum between $|\min\{k, Y^1_{k,t} = 1\} - \max\{k, Y^2_{k,t} = 1\}|$ and $|\max\{k, Y^1_{k,t} = 1\} - \min\{k, Y^2_{k,t} = 1\}|$. Then the area loss is the mean of the δ_t. In the audio literature [15], this loss is sometimes called the "mean absolute deviation" loss and is noted $\delta_{\text{abs}}(Y^1, Y^2)$.
Unfortunately, in the general case of alignment problem δ_{abs} is not linear in the matrices Y. But in the context of alignment of sequences of two different nature, one of the signal is a reference and thus the index sequence α defined in Eq. (1) is increasing, e.g. for the audio to partition alignment problem [13]. This loss is then linear in each of its arguments. More precisely, if we introduce the matrices $L \in \mathbb{R}^{T_A \times T_A}$ which is lower triangular with ones, we can write the loss as

$$\begin{aligned} \ell_O &= \|L(Y_1 - Y_2)\|_F^2 \\ &= \operatorname{Tr}(L Y_1 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) + \operatorname{Tr}(L Y_2 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) - 2\operatorname{Tr}(L Y_1 Y_2^\top L^\top). \end{aligned} \tag{5}$$

We now prove that this loss corresponds to the area loss in this special case. Let Y be an alignment matrix and $L \in \mathbb{R}^{T_A \times T_A}$ be the matrix such that $L_{r,s} = 1$ if and only if

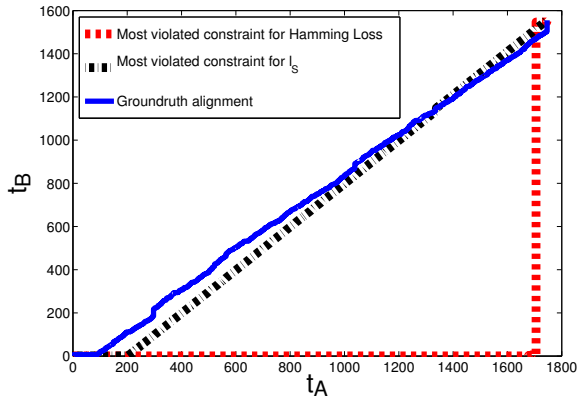


Figure 2: On the real world Bach chorales dataset, we have represented the most violated constrained at the end of learning, when the training loss is the Hamming one or the symmetrized area loss. Note also that, in terms of Hamming loss the most violated contraint for ℓ_S and the Hamming one are the same.

$r \geq s$. Then it is easy see that $(LY)_{i,j} = \sum_k L_{i,k} Y_{k,j} = \sum_{k=0}^{i} Y_{k,j}$. If Y does not have vertical moves, i.e. for each j there is an unique k_j such that $Y_{k,j} = 1$, we have that $(LY)_{i,j} = 1$ if and only if $i \geq k_j$. So $\sum_{i,j}(LY)_{i,j} = \#\{(i,j), i \geq k_j\}$, which is exactly the area under the curve determined by the path of Y.
In all our experiments, we use δ_{abs} for evaluation but not for training.

Approximation of the area loss: the symmetrized area loss. In many real world applications [15], the best loss to assess the quality of an alignment is the area loss. As shown by our experiments, if the Hamming loss is sufficient in some simple situations and allows to learn a metric that leads to good alignment performance in terms of area loss, on more challenging datasets it does not work at all (see Sec. 5). This is due to the fact that two alignments that are very close in terms of area loss can suffer a big Hamming loss. In Fig. 2, we provide examples where the Hamming loss is not sufficient to assess performance. Thus it is natural to extend the formulation of Eq. (5) to matrices in $\mathcal{Y}(X)$. We first start by symmetrizing the formulation of Eq. (5) to overcome problems of overpenalization of vertical vs. horizontal moves. Let $L_1 \in \mathbb{R}^{T_B \times T_B}$ be the matrix such that $L_{r,s} = 1$ if and only if $r \geq s$. We define, for any binary matrices Y^1 and Y^2,

$$\begin{aligned}\ell_S(Y_1, Y_2) &= \frac{1}{2}\big(\|L(Y_1 - Y_2)\|_F + \|(Y_1 - Y_2)L_1)\|_F^2\big) \\ &= \frac{1}{2}\Big[\operatorname{Tr}(Y_1^\top L^\top L Y_1) + \operatorname{Tr}(LY_2 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) - 2\operatorname{Tr}(Y_2^\top L^\top L Y_1) \\ &\quad + \operatorname{Tr}(Y_1 L_1 L_1^\top Y) + \operatorname{Tr}(Y_2^\top \mathbf{1}_{T_A} \mathbf{1}_{T_B} L_1 L_1^\top Y_2) - 2\operatorname{Tr}(Y_2 L_1 L_1^\top Y_1^\top)\Big].\end{aligned} \tag{6}$$

We propose to use the following trick to obtain a *concave* loss over $\overline{\mathcal{Y}}(X)$, the convex hull of $\mathcal{Y}(X)$. Let us introduce $D = \lambda_{\max}(L^\top L) I_{T_A \times T_A}$ and $D_1 = \lambda_{\max}(L_1 L_1^\top) I_{T_B \times T_B}$ with $\lambda_{\max}(U)$ the largest eigenvalue of U. For any binary matrices Y_1, Y_2, we have that:

$$\begin{aligned}\ell_S(Y_1, Y_2) = \frac{1}{2}\big[&\operatorname{Tr}(Y_1^\top (L^\top L - D) Y_1) + \operatorname{Tr}(D Y_1 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) \\ &+ \operatorname{Tr}(L Y_2 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) - 2 \operatorname{Tr}(Y_2^\top (L^\top L - D) Y_1) \\ &+ \operatorname{Tr}(Y_1 (L_1 L_1^\top - D) Y) + \operatorname{Tr}(Y_1 D_1 \mathbf{1}_{T_B} \mathbf{1}_{T_A}^\top) \\ &\operatorname{Tr}(Y_2^\top L_1 L_1^\top Y_2) - 2 \operatorname{Tr}(Y_2 L_1 L_1^\top Y_1^\top) \big]\end{aligned}$$

and we get a *concave* function over $\overline{\mathcal{Y}}(X)$ that coincides with ℓ_S on $\mathcal{Y}(X)$.

3.2 Empirical loss minimization

Recall that we are given n alignment examples $(X^i, Y^i)_{1 \leq i \leq n}$. For a fixed loss ℓ, our goal is now to solve the following minimization problem in W:

$$\min_{W \in \mathcal{W}} \big\{ \frac{1}{n} \sum_{i=1}^{n} \ell\big(Y^i, \operatorname*{argmax}_{Y \in \mathcal{Y}_{T_A^i, T_B^i}} \operatorname{Tr}(C(X^i; W)^\top Y)\big) + \lambda \Omega(W) \big\}, \tag{7}$$

where $\Omega = \frac{\lambda}{2} \|W\|_F^2$ is a convex regularizer preventing from overfitting, with $\lambda \geq 0$.

4 Large margin approach

In this section we describe a large margin approach to solve a surrogate to the problem in Eq. (7), which is untractable. As shown by Eq. (3), the decoding task is the maximum of a linear function in the parameter W and aims at predicting an output over a large and discrete space (the space of potential alignments with respect to the constraints in Eq. (1)). Learning W thus falls in the structured prediction framework [25, 22]. We define the hinge-loss, a convex surrogate to ℓ, by

$$L(X, Y; W) = \max_{Y' \in \mathcal{Y}(X)} \Big\{ \ell(Y, Y') - \operatorname{Tr}(W^\top [\phi(X, Y) - \phi(X, Y')]) \Big\}. \tag{8}$$

The evaluation of L is usually referred to as the "loss-augmented decoding", see [25]. Among the aforementionned losses, the Hamming loss ℓ_H is the only one leading directly to a tractable loss-augmented decoding problem and thus that falls directly into the structured prediction framework. Indeed, plugging Eq. (4) into (8) leads to a loss-augmented decoding that is a LP over the set $\mathcal{Y}(X)$ and that can therefore be solved using the dynamic time warping algorithm. If we define $\widehat{Y}^i$ as the argmax in Eq. (8) when $(X, Y) = (X^i, Y^i)$, then elementary computations show that

$$\widehat{Y}^i = \operatorname*{argmin}_{Y \in \mathcal{Y}_{T_A, T_B}} \operatorname{Tr}((U^\top - 2 Y^{i\top} - C(X^i; W)^\top) Y),$$

where $U = \mathbf{1}_{T_B}\mathbf{1}_{T_B}^\top \in \mathbb{R}^{T_A \times T_B}$.
We now aim at solving the following problem, sometimes called the *margin-rescaled problem*:

$$\min_{W \in \mathcal{W}} \frac{\lambda}{2}\|W\|_F^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{Y \in \mathcal{Y}_{T_A^i, T_B^i}} \left\{ \ell(Y, Y^i) - \mathrm{Tr}(W^\top \left[\phi(X^i, Y^i) - \phi(X^i, Y)\right]) \right\}. \tag{9}$$

Hamming loss case. From Eq. (3), one can notice that our joint feature map is linear in Y. Thus, if we take a loss that is linear in the first argument of ℓ, for instance the Hamming loss, the loss-augmented decoding is the maximization of a linear function over the spaces $\mathcal{Y}(X)$ that we can do efficiently using dynamic programming algorithms (see Sec. 2.1 and supplementary material).
That way, plugging the Hamming loss (Eq. (4)) in Eq. (9) leads to a convex structured prediction problem. This problem can be solved using standard techniques such that cutting plane methods [12], stochastic gradient descent [20], or block-coordinate Frank-Wolfe in the dual [16]. Note that we adapted the standard unconstrained optimization methods to our setting, where $W \succeq 0$.

Optimization using the symmetrized area loss. In this section we propose to show that it is possible to deal The symmetrized area loss is concave in its first argument, thus the problem of Eq. 9 using it is in a min/max form and thus deriving a dual is straightforward. Details can be found in the supplementary material. If we plug the symmetrized area loss ℓ_S (SAL) defined in Eq. (6) into our problem (9), we can show that the dual of (9) has the following form:

$$\begin{aligned} \min_{(Z^1,\dots,Z^n) \in \overline{\mathcal{Y}}} \quad & \tfrac{1}{2\lambda n^2}\| \textstyle\sum_{i=1}^n - \sum_{j,k}(Y_i - Z^i)_{j,k}(a_j - b_k)(a_j - b_k)^T\|_F^2 \\ & -\tfrac{1}{n}\textstyle\sum_{i=1}^n \ell_S(Z, Z^i), \end{aligned} \tag{10}$$

if we denote by $\overline{\mathcal{Y}}(X^i)$ the convex hull of the sets $\mathcal{Y}(X^i)$, and by $\overline{\mathcal{Y}}$ the cartesian product over all the training examples i of such sets. Note that we recover a similar result as [16]. Since the SAL loss is concave, the aforrementionned problem is convex. The problem (10) is a quadratic program over the compact set $\overline{\mathcal{Z}}$. Thus we can use a Frank-Wolfe [8] algorithm. Note that it is similar to the one proposed by Lacoste-Julien et al. [16] but with supplementary term due to the concavity of the loss.

5 Experiments

We applied our method to the task of learning a good similarity measure for aligning audio signals. In this field researchers have spent a lot of efforts in designing well-suited and meaningful features [13, 5]. But the problem of combining these features for aligning temporal sequences is still challenging.

5.1 Dataset of Kirchhoff and Lerch [15]

Dataset description. First, we applied our method on the dataset of Kirchhoff and Lerch [15]. In this dataset, pairs of aligned examples (A^i, B^i) are artificially created by stretching an original audio signal. That way the groundtruth alignment Y^i is known and thus the data falls into our setting A more precise description of the dataset can be found in [15].
The $N = 60$ pairs are stretched along two different tempo curves. Each signal is made of $30\mathrm{s}$ of music that are divided in frames of $46\mathrm{ms}$ with a hopsize of $23\mathrm{ms}$, thus leading to a typical length of the signals of $T \approx 1300$ in our setting. We keep $p = 11$ features simple to implement and that are known to perform well for alignment tasks [15]. Those were: five MFCC [9] (labeled M1,...,M5 in Fig. 3), the spectral flatness (SF), the spectral centroid (SC), the spectral spread (SS), the maximum of the envelope (Max), and the power level of each frame (Pow), see [15] for more details on the computation of the features. We normalize each feature by subtracting the median value and dividing by the standard deviation to the median, as audio data are subject to outliers.

Experiments. We conducted the following experiment: for each individual feature, we perform alignment using dynamic time warping algorithm and evaluate the performance of this single feature in terms of losses typically used to asses performance in this setting [15]. In Fig. 3, we report the results of these experiments.
Then, we plug these data into our method, using the Hamming loss to learn a linear positive combination of these features. The result is reported in Fig 3. Thus, combining these features on this dataset yields to better performances than only considering a single feature.

5.2 Chorales dataset

Dataset. The Bach 10 dataset[2] consists in ten J. S. Bach's Chorales (small quadriphonic pieces). For each Chorale, a MIDI reference file corresponding to the "score", or basically a representation of the partition. The alignments between the MIDI files and the audio file are given, thus we have converted these MIDI files into audio following what is classically done for alignment (see e.g, [11]). That way we fall into the audio-to-audio framework in which our technique apply. Each piece of music is approximately $25\mathrm{s}$ long, leading to similar signal length.

Experiments. We use the same features as in Sec. 5.1. As depicted in Fig. 4, the optimization with Hamming loss performs poorly on this dataset. In fact, the best individual feature performance is far better than the performance of the learned W. Thus metric learning with the "practical" Hamming loss performs much worse than the best single feature.
Then, we conducted the same learning experiment with the symetrized area loss ℓ_S. The resulting learned parameter is far better than the one learned using the Hamming

[2] `http://music.cs.northwestern.edu/data/Bach10.html`.

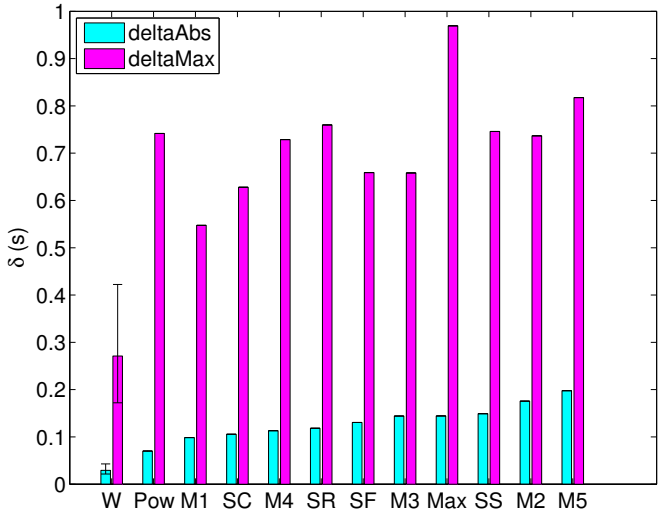


Figure 3: Comparison of performance between individual features and the learned metric. The left bars are mean δ_{abs} error in frame on the dataset and the right are $\delta_{\max}$. Error bars for the performance of the learned metric were determined with the best and the worst performance on 5 different experiments.

loss. We get a performance that is similar to the one of the best feature. Note that these features were handcrafted and reaching their performance on this hard task with only a few training instances is already challenging.
In Fig. 2, we have depicted the result, for a learned parameter W, of the loss augmented decoding performed either using the area. As it is known for structured SVM, this represents the most violated constraint [25]. We can see that the most violated constraint for the Hamming loss leads to an alignment which is totally unrelated to the groundtruth alignment whereas the one for the symmetrized area loss is far closer and much more discriminative.

5.3 Feature selection

Last, we conducted experiments over the same datasets. Starting from low level features, namely the 13 leading MFCCs coefficients and their first two derivatives, we learn a linear combination of these that achieves good alignment performance in terms of the area loss.Note that very little musical prior knowledge is put into these. Moreover we either improve on the best handcrafted feature on the dataset of [15] or perform similarly. On both datasets, the performance of learned combination of handcrafted features performed similarly to the combination of these 39 MFCCs coefficients

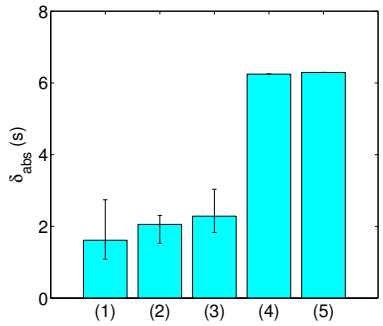


Figure 4: Performance of our algorithms on the Chorales dataset. From left to right: (1) Best single feature, (2) Best learned combination of features using the symmetrized area loss ℓ_S, (3) Best combination of MFCC and derivatives learned with ℓ_S, (4) Best combination of MFCCs and derivatives learned with Hamming loss, (5) Best combination of features of [15] using Hamming loss.

6 Conclusion

In this paper, we have presented a structured prediction framework for learning the metric in temporal alignment problems. We were able to combine hand-crafted features, as well as building automatically new state-of-the-art features from basic low-level information with very little expert knowledge.
Technically, this is made possible by considering a loss beyond the usual Hamming loss which is typically used because it is "practical" within a structured prediction framework (linear in the output representation).
The present work may be extended in several ways, the main one being to consider cases where only partial information about the alignments is available. This is often the case in music [5] or bioinformatics applications. Note a simple alternating optimization between metric learning and constrained alignment provide a simple first solution, which could probably be improved upon.

Acknowledgements

We acknowledge the support of the GARGANTUA project (Mastodons program of CNRS), the grant SIERRA-23999 from the European Research Council and a PhD fellowship from the EADS Foundation.

References

[1] J. Aach and G. M. Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508, 2001.

[2] C. Banderier and S. Schwer. Why Delannoy numbers? *Journal of statistical planning and inference*, 135(1):40–54, 2005.

[3] D. Bertsekas. *Nonlinear programming*. Athena Scientific Belmont, MA, 1999.

[4] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola. Learning graph matching. *IEEE Trans. on PAMI*, 31(6):1048–1058, 2009.

[5] A. Cont, D. Schwarz, N. Schnell, C. Raphael, et al. Evaluation of real-time audio-to-score alignment. In *Proc. ISMIR*, 2007.

[6] M. Cuturi, J.-P. Vert, O. Birkenes, and T. Matsui. A kernel for time series based on global alignments. In *Proc. ICASSP*, volume 2, pages II–413. IEEE, 2007.

[7] S. Dixon and G. Widmer. Match: A music alignment tool chest. In *Proc. ISMIR*, pages 492–497, 2005.

[8] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[9] B. Gold, N. Morgan, and D. Ellis. *Speech and audio signal processing: processing and perception of speech and music*. John Wiley & Sons, 2011.

[10] R. Hamming. Error detecting and error correcting codes. *Bell system technical journal*, 29(2), 1950.

[11] N. Hu, R. B. Dannenberg, and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. *Computer Science Department*, page 521, 2003.

[12] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.

[13] C. Joder, S. Essid, and G. Richard. Learning optimal features for polyphonic audio-to-score alignment. *IEEE Trans. on Audio, Speech, and Language Processing*, 21(10):2118–2128, 2013.

[14] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan. A large margin algorithm for speech-to-phoneme and music-to-score alignment. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(8):2373–2382, 2007.

[15] H. Kirchhoff and A. Lerch. Evaluation of features for audio-to-audio alignment. *Journal of New Music Research*, 40(1):27–41, 2011.

[16] S. Lacoste-Julien, M. Jaggi, M. Schmidt, P. Pletscher, et al. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *Proc. ICML*, 2013.

[17] R. Lajugie, F. Bach, and S. Arlot. Large-margin metric learning for constrained partitioning problems. In *Proc. ICML*, 2014.

[18] B. McFee and G. R. Lanckriet. Metric learning to rank. In *Proc. ICML*, pages 775–782, 2010.

[19] M. Müller. *Information retrieval for music and motion*. Springer, 2007.

[20] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30, 2011.

[21] M. Szummer, P. Kohli, and D. Hoiem. Learning crfs using graph cuts. In *Proc. CVPR*. 2008.

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

[23] J. D. Thompson, F. Plewniak, and O. Poch. Balibase: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics*, 15 (1):87–88, 1999.

[24] A. Torres, A. Cabada, and J. J. Nieto. An exact formula for the number of alignments between two dna sequences. *Mitochondrial DNA*, 14(6):427–430, 2003.

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun, and Y. Singer. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(9):1453–1484, 2005.

[26] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

A Derivation of the BCFW-like algorithm for the quadratic loss.

A.1 Relaxing the set for loss augmented inference.

Let us start from the global structured objective equation of the paper. Recall that we dispose of the training examples $((X^1, Y^1), \ldots, (X^n, Y^n))$. In order to make the derivation easier, and following Lacoste-Julien et al. [16], we denote the difference between the feature map associated to any $Y \in \mathcal{Y}(X^i)$ and the one associated to the true training example label Y_i by: $\mathrm{Tr}(W\phi(X^i, Y^i)) = \mathrm{Tr}(W \sum_{j,k}(Y^i_{j,k} - Y_{j,k})(a^i_j - b^i_k)(a^i_j - b^i_k)^T) = \langle W, \psi^i(Y)\rangle$. The objective of structured prediction is thus:

$$\min_{W \in \mathcal{W}} \frac{\lambda}{2}\|W\|_2^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{Y \in \mathcal{Y}(X^i)} \left\{\ell_i(Y, Y^i) - \langle W, \psi^i(Y)\rangle\right\}. \tag{11}$$

The term $\max_{Y \in \mathcal{Y}(X^i)} \left\{\ell_i(Y, Y^i) - \langle W, \psi^i(Y)\rangle\right\}$ corresponds to the structural hinge loss for our problem. Let us introduce $\overline{\mathcal{Y}}(X^i)$ the convex hull of the sets $\mathcal{Y}(X^i)$. We will also use $\overline{\mathcal{Y}} = \overline{\mathcal{Y}}(X^1) \times \ldots \times \overline{\mathcal{Y}}(X^n)$. From now on, we will perform the loss augmented decoding on this relaxed set. This problem has potentially *non integral*

solutions. We call the maximization of the hinge loss over $\overline{\mathcal{Y}}$ the *loss augmented inference*. Now we can write a new optimization objective:

$$\min_{W \in \mathcal{W}} \frac{\lambda}{2}\|W\|_2^2 + \max_{(Z_1,\dots,Z_n)\in\overline{\mathcal{Y}}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \left[\ell_i(Z_i, Y^i) - \langle W, \psi^i(Z_i)\rangle\right] \right\}. \tag{12}$$

Note that since our joint feature map $\phi(X^i, Y)$ is linear in Y, if ℓ is linear as well (for instance if ℓ is the Hamming loss), this problem is strictly equivalent to (11) since in that case, the loss-augmented inference is a LP over $\overline{\mathcal{Y}}(X^i)$, which has necessary a solution in $\mathcal{Y}(X^i)$ (see, e.g, [Prop. B.21] of [3].
In general, in order to be convex and thus tractable, the aforrementioned problem requires a loss which is concave over the convex sets $\overline{\mathcal{Y}}(X^i)$.

A.2 Dual of the structured SVM

Since Prob. (11) is in saddle point form, we get the dual by switching the max and the min:

$$\max_{(Z_1,\dots,Z_n)\in\overline{\mathcal{Y}}(X^1)\times\dots\times\overline{\mathcal{Y}}(X^n)} \min_{W\in\mathcal{W}} \frac{\lambda}{2}\|W\|_2^2 + \{\frac{1}{n}\sum_{i=1}^{n}\left[\ell_i(Y, Y^i) - \langle W, \psi^i(Z_i)\rangle\right]\}. \tag{13}$$

From the above equation, we deduce the following relation linking primal variable W and dual variables $(Z_1, \dots, Z_n) \in \mathcal{Y}(X^1) \times \dots \times \mathcal{Y}(X^n)$:
In the specific case when $\mathcal{W}$ is unconstrained and simply equals to $\mathbb{R}^{p\times p}$, this reduces to:

$$W = \frac{1}{\lambda}\sum_{i=1}^{n}\psi_i(Z_i). \tag{14}$$

If $\mathcal{W}$ is the set of symmetric semidefinite positive matrices we get:

$$W = \frac{1}{\lambda}\sum_{i=1}^{n}(\psi_i(Z_i))_+, \tag{15}$$

with $(\psi_i(Z_i))_+$ the projection of $(\psi_i(Z_i))$ over $\mathcal{W}$.
Eventually if we consider $\mathcal{W}$ the set of diagonal matrices, and denote by Diag the operator associating to a matrix the matrix composed of its diagonal:

$$W = \frac{1}{\lambda}\sum_{i=1}^{n}\mathrm{Diag}(\psi_{\mathrm{i}}(\mathrm{Z_i})). \tag{16}$$

These relations are also known as the "representer theorems".
For what follows we consider the case of $\mathcal{W} = \mathbb{R}^{p\times p}$ but dealing with the other cases is similar.
In that case the dual can be written simply as:

$$\max_{(Z_1,\dots,Z_n)\in\overline{\mathcal{Y}}(X^1)\times\dots\times\overline{\mathcal{Y}}(X^n)} -\frac{1}{2\lambda n^2}\|\sum_{i=1}^{n}\psi_i(Z_i)\|_F^2 + \frac{1}{n}\sum_{i=1}^{n}\ell(Y_i, Z_i). \tag{17}$$

We recover a result similar to the ones of Lacoste-Julien et al. [16].

A.3 A Frank-Wolfe algorithm for solving Prob. (17)

Now, we can derivate a Frank-Wolfe algorithm for solving the dual problem of 17. As noted in the paper, we are able to maximize or minimize any linear form over the sets $\mathcal{Y}(X^i)$, thus we are able to solve LPs over the convex hulls $\overline{\mathcal{Y}}(X^i)$ of such sets.
Plugging back the specific form of our joint feature map directly into Eq. (17) we get that $\psi_i(Z^i) = -\sum_{j,k}(Y_i - Z^i)_{j,k}(a_j - b_k)(a_j - b_k)^T$ and thus we can write the dual problem as:

$$\min_{\substack{(Z^1,\dots,Z^n)\in \\ \overline{\mathcal{Y}}(X^1)\times\dots\times\overline{\mathcal{Y}}(X^n)}} \frac{1}{2\lambda n^2}\|\sum_{i=1}^{n} -\sum_{j,k}(Y_i - Z^i)_{j,k}(a_j - b_k)(a_j - b_k)^T\|_F^2 - \frac{1}{n}\sum_{i=1}^{n}\ell(Y^i, Z^i) \quad (18)$$

Now, as in the paper, let us introduce $L \in \mathbb{R}^{T_A\times T_A}$ and $L_1 \in \mathbb{R}^{T_B\times T_B}$. If U_i is the matrix of ones of the same size as Z^i, we consider the following loss:

$$\begin{aligned}\ell(Y^i, Z^i) &= \frac{1}{2}\big[\operatorname{Tr}(Z^{iT}(L^TL - D)Z^i) + \operatorname{Tr}(DZ^iU^i) + \operatorname{Tr}(Y^{iT}L^TL^i) - 2\operatorname{Tr}(Z^{iT}L^TLY^i) \\ &+ \operatorname{Tr}(Z^i(L_1^TL_1 - D_1)Z^i) + \operatorname{Tr}(D_1Z^iU^i) + \operatorname{Tr}(Y^iL_1^TL_1^i) - 2\operatorname{Tr}(ZL_1^TL_1Y^i)\big] \quad (19)\end{aligned}$$

This loss is sound for alignments problems since, when Y_i and Z^i are in $\mathcal{Y}$, this is simply the ℓ_S loss $\|LY_i - LZ^i\|_F^2 + \|Y_iL_1 - Z^iL_1\|_F^2$.
Thus we get the following overall dual objective:

$$\begin{aligned}\min_{(Z^1,\dots,Z^n)\in\overline{\mathcal{Y}}} \quad & \frac{1}{2\lambda n^2}\|\sum_{i=1}^{n} -\sum_{j,k}(Y_i - Z^i)_{j,k}(a_j - b_k)(a_j - b_k)^T\|_F^2 \\ - \quad & \frac{1}{n}\big(\sum_{i=1}^{n}[\operatorname{Tr}(Z^{iT}(L^TL - D)Z^i) + \operatorname{Tr}(Z^{iT}DU^i) + \\ & \operatorname{Tr}(Y^TL^TL^i) - 2\operatorname{Tr}(Z^{iT}L^TLY^i) + \operatorname{Tr}(Z^i(L_1^TL_1 - D_1)Z^{iT}) \\ + \quad & \operatorname{Tr}(U_iD_1Z^i) + \operatorname{Tr}(YL_1^TL_1^i) - 2\operatorname{Tr}(ZL_1^TL_1Y^{iT})]\big). \quad (20)\end{aligned}$$

We recall that D is a diagonal matrix such that $A^\top A - D \preceq 0$ and thus our objective is convex. Our dynamic programming algorithm (DTW) is able to maximize any linear function over the sets. Thus we can use a Frank-Wolfe [8] technique. At iteration t, this algorithm iteratively computes a linearization of the function at the current point $(Z^1,\dots Z^n)_k$, computes a linearization of the function, optimize it, get a new point $(Z^1,\dots Z^n)_k^*$and then make a convex combination using a stepsize γ.
Note that we have directly a stochastic version of such an algorithm. As noted in Lacoste-Julien et al. [16] instead of computing a gradient for each block of variable Z^i, we simply need to choose randomly one block at each timestep and make an update on these variables.
The linearization simply consists in computing the matrix gradient for each of the matrix variables Z^i which turns out to be:

$$\begin{aligned}\nabla_{Z^i}(g) = \quad & \tfrac{1}{n}\big[C - \tfrac{1}{2}\big(2(L^TL - D)Z^i + DU_i - 2L^TLY^i \\ + \quad & 2Z^i(L_1^TL_1 - D) + U_iD - 2Y^iL^TL\big)\big] \quad (21)\end{aligned}$$

where C is simply the affinity matrix of dynamic time warping.

B The dynamic time warping algorithm

Let us give the pseudocode of the dynamic time warping that maximize the LP (2) of the article.In opposition to Müller [19], we give a version of the algorithm for the affinity matrix C. Intuitively, the cost matrix is the opposite of a cost matrix, thus we aim to maximize the cumulated affinity instead of minimizing the cumulated cost. This algorithm is $O(T_A T_B)$, making it very costly to compute for large time series.

Cumulated affinity matrix:
$T, S \leftarrow \text{size}(C)$, $D \leftarrow \text{zeros}(T_A + 1, T_B + 1)$
for $i = 1$ **to** T_A **do**
  $D(i, 0) \leftarrow -\infty$
end for
for $j = 1$ **to** T_B **do**
  $D(0, j) \leftarrow -\infty$
end for
for $i = 1$ **to** T_A **do**
  for $j = 1$ **to** T_B **do**
    $D(i, j) \leftarrow C(i, j) + \max(D(i-1, j), D(i, j-1), D(i-1, j-1))$
  end for
end for
Backtracking:
$Y \leftarrow \text{zeros}(T_A, T_B)$, $i \leftarrow T_A$, $j \leftarrow T_B$
while $i > 1$ **or** $j > 1$ **do**
  $Y(i, j) \leftarrow 1$
  if $i == 1$ **then**
    $j \leftarrow j - 1$
  else if $j == 1$ **then**
    $i \leftarrow i - 1$
  else
    $m \leftarrow \max(D(i-1, j), D(i, j-1), D(i-1, j-1))$
    if $D(i-1, j) == m$ **then**
      $i \leftarrow i - 1$
    else if $D(i, j-1) == m$ **then**
      $j \leftarrow j - 1$
    else
      $i \leftarrow i - 1$, $j \leftarrow j - 1$
    end if
  end if
end while
return Y

Figure 5: The dynamic time-warping algorithm that solves the LP (2), for a given similarity matrix C.