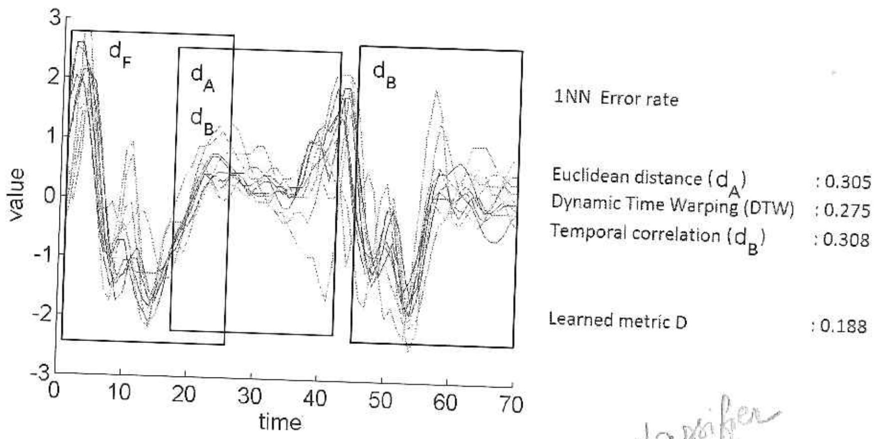
3.1 Motivations

kNN

This work focuses on defining a 'good' metric for classification of time series. The definition of a metric to compare samples is a fundamental issue in data analysis or machine learning. As seen in Chapter 2, temporal data may be compared based on one or several characteristics, called **modalities** (amplitude, behavior, frequency) and they might be subjected to delays. In some classification applications, the most discriminative characteristic between time series of different classes can be localized on a smaller part of the signal (scale). We believe that the definition of a temporal metric should consider at least these different aspects (modality, delay, scale) in order to improve the performance of a classifier. Fig. 3.1 illustrates a result obtained with our proposition. There is a significant improvement in classification performances by taking into account in the metric definition, several modalities (amplitude d_A, behavior d_B, frequential d_F) located at different scales (illustrated by black rectangles in the figure). The performance of the learned combined metric is compared with the ones of the standard metrics that take into account for each, only one modality on a global scale (involving all time series elements).



classifier

Figure 3.1: SonyAIBO dataset and error rate using a kNN ($k = 1$) with standard metrics (Euclidean distance, Dynamic Time Warping, temporal correlation) and a learned combined metric D. The figure shows the 4 major metrics involve in the combined metric D and their respective temporal scale (black rectangles).

Our aim is to take leverage from the metric learning framework [WS09b]; [BHS12] to learn a multi-modal and multi-scale temporal metric for time series nearest neighbors classification. Specifically, our objective is to learn from the data a linear or non linear function that combines several temporal modalities at several temporal scales, that satisfies metric properties (Section 2.2), and that generalizes the case of unimodal metrics at the global scale. Metric learning can be defined as learning, from the data and for a task, a pairwise function (*i.e.*, a similarity, dissimilarity or a distance) that brings closer samples that are expected to be similar, and pushes far away those expected to be dissimilar. Such similarity and dissimilarity expectations, is inherently task- and application-dependent, generally given *a priori* and fixed during the learning process. Metric learning has become an active area of research in the applying the k-NN classification:

$$D^2_{\mathbf{L}}(\mathbf{x}_i, \mathbf{x}_j) = D^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) \tag{3.2}$$

$$D^2_{\mathbf{L}}(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j)||^2_2 \tag{3.3}$$

Commonly, the squared distances can be expressed in terms of a square matrix:

$$D^2_{\mathbf{L}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)\mathbf{L}^T\mathbf{L}(\mathbf{x}_i - \mathbf{x}_j) \tag{3.4}$$

Let $\mathbf{M} = \mathbf{L}'\mathbf{L}$. It is proved that any matrix $\mathbf{M}$ formed as below from a real-valued matrix $\mathbf{L}$ is positive semidefinite (*i.e.*, no negative eigenvalues) [WS09a]. Using the matrix $\mathbf{M}$, squared distances can be expressed as:

$$D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)\mathbf{M}(\mathbf{x}_i - \mathbf{x}_j) \tag{3.5}$$

The computation of the learned metric $D_{\mathbf{M}}$ can thus be seen as a two steps procedure: first, it computes a linear transformation of the samples $\mathbf{x}_i$ given by the transformation $\mathbf{L}$; second, it computes the Euclidean distance in the transformed space:

$$D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = D^2_{\mathbf{L}}(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) \tag{3.6}$$

Learning the linear transformation $\mathbf{L}$ is thus equivalent to learn the corresponding Mahalanobis metric D parametrized by $\mathbf{M}$. This equivalence leads to two different approaches to metric learning: we can either estimate the linear transformation $\mathbf{L}$, or estimate a positive semidefinite matrix $\mathbf{M}$. LMNN solution refers on the latter one.

Mathematically, the metric learning problem can be formalized as an optimization problem involving two terms for each sample $\mathbf{x}_i$: one term penalizes large distances between nearby inputs with the same label (pull), while the other term penalizes small distances between inputs with different labels (push). For all samples $\mathbf{x}_i$, this implies a minimization problem:

$$\begin{cases} \underset{\mathbf{M},\xi}{\operatorname{argmin}} \underbrace{\sum_{i,j\rightsquigarrow i} D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j)}_{pull} + C \underbrace{\sum_{i,j\rightsquigarrow i, l\not\rightsquigarrow i} \xi_{ijl}}_{push} \end{cases} \tag{3.7}$$

$$\begin{aligned} &\text{s.t. } \forall j \rightsquigarrow i, l \not\rightsquigarrow i, \\ &D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_l) - D^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) \geq 1 - \xi_{ijl} \\ &\xi_{ijl} \geq 0 \\ &\mathbf{M} \succeq 0 \end{aligned}$$

where ξ_{ijl} are slack variables, C is a trade-off between the push and pull term and $\mathbf{M} \succeq 0$ means that $\mathbf{M}$ is a positive semidefinite matrix. Generally, the parameter C is tuned via cross validation and grid search (Section 1.1.2). Similarly to Support Vector Machine (SVM) approach, slack variables ξ_{ijl} are introduced to relax the optimization problem.

Dire pourquoi on a rappelé cette méthode
qu'est-ce qu'on va en faire ?

In that space, the norm of a pairwise vector $||\mathbf{x}_{ij}||$ refers to the proximity between the time series $\mathbf{x}_i$ and $\mathbf{x}_j$. In particular, if $||\mathbf{x}_{ij}|| = 0$ then $\mathbf{x}_j$ is identical to $\mathbf{x}_i$ according to all metrics d_h.

3.3.2 Interpretation in the pairwise dissimilarity space

In this section, we give more detailed interpretations in the dissimilarity space. We recall that the norm of a pairwise vector is given by:

$$||\mathbf{x}_{ij}|| = \sqrt{\sum_{h=1}^{p} d_h^{2}(\mathbf{x}_i, \mathbf{x}_j)} \quad (3.10)$$

?

D -> distance / parts 0?

In the following, we denote the norm $||\mathbf{x}_{ij}||$ as an initial distance in the dissimilarity space and call it D_0. Any other initial metric could have been chosen. The norm of a pairwise vector $\mathbf{x}_{ij}$ can be interpreted as a proximity measure: the lower the norm of $\mathbf{x}_{ij}$ is, the closer are the time series $\mathbf{x}_i$ and $\mathbf{x}_j$. Two pairwise vectors $\mathbf{x}_{ij}$ and $\mathbf{x}_{kl}$ that are on a same line that passes through the origin $\mathbf{x}_{ii} = \mathbf{0}$ represent differences in the the same proportions between their respective modalities (Fig. 3.4).

Je ne comprends pas bien l'intérêt de cette partie

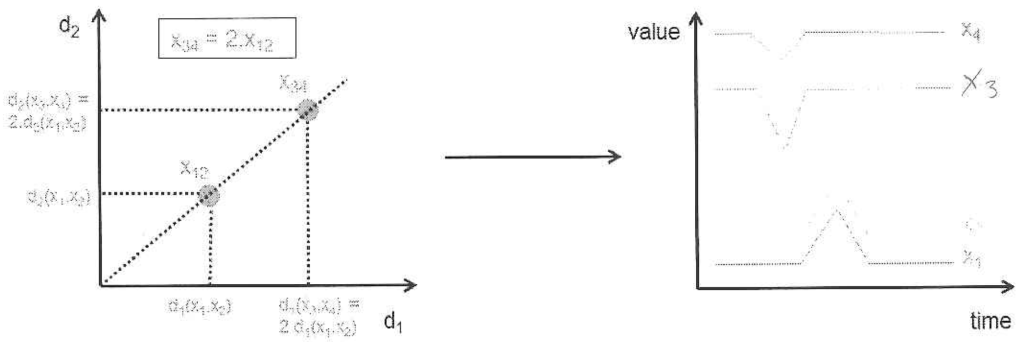


Figure 3.4: Example of interpretation of two pairwise vectors $\mathbf{x}_{12}$ and $\mathbf{x}_{34}$ on a same line passing through the origin in the pairwise dissimilarity space.

The Euclidean distance $\sqrt{\sum_{h=1}^{p}(d_h(\mathbf{x}_i, \mathbf{x}_j) - d_h(\mathbf{x}_k, \mathbf{x}_l))^2}$ between two pairwise vectors $\mathbf{x}_{ij}$ and $\mathbf{x}_{kl}$ represents the similarity between the differences among the same modalities, in the same proportions. Note that if the Euclidean distance is close to 0 ($\mathbf{x}_{ij}$ and $\mathbf{x}_{kl}$ are close in the dissimilarity space), it doesn't mean that the time series $\mathbf{x}_i$, $\mathbf{x}_j$, $\mathbf{x}_k$ and $\mathbf{x}_l$ are similar. Fig 3.5 shows an example of two pairwise vectors $\mathbf{x}_{ij}$ and $\mathbf{x}_{kl}$ close together in the pairwise space. However, in the temporal space, the time series $\mathbf{x}_1$ and $\mathbf{x}_3$ are not similar for example. It means that $\mathbf{x}_i$ is as similar to $\mathbf{x}_j$ as $\mathbf{x}_k$ is to $\mathbf{x}_l$, *i.e.*, the distance D_0 between $\mathbf{x}_i$ and $\mathbf{x}_j$ is nearly the same than the distance D_0 between $\mathbf{x}_k$ and $\mathbf{x}_l$.

Pourquoi on passe en euclidien?

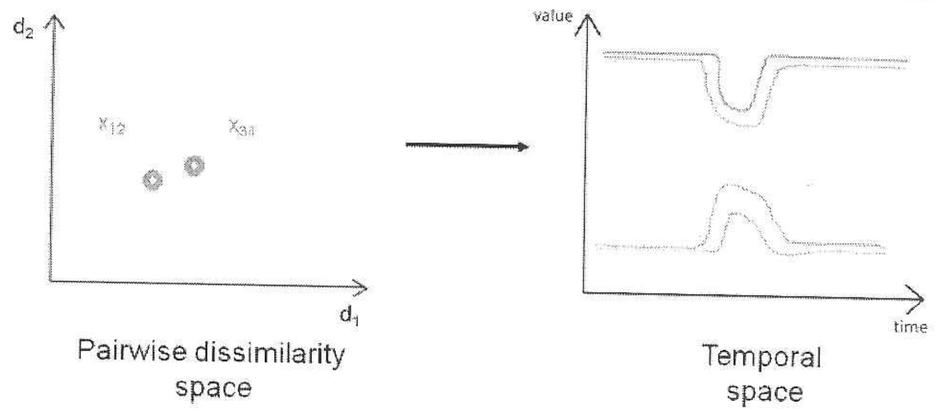


Figure 3.5: Example of two pairwise vectors $\mathbf{x}_{12}$ and $\mathbf{x}_{34}$ close in the pairwise dissimilarity space. However, the time series $\mathbf{x}_1$ and $\mathbf{x}_3$ are not similar in the temporal space.

temporal localisation.

3.3.3 Multi-scale description for time series

on localized temporal segments

Bien définir ce qu'on appelle multi-scale

scale → temporal localisation

The multi-modal representation in the dissimilarity space can be enriched for time series by measuring each unimodal metric d_h at different scales. Note that the distance measures (amplitude-based d_A, frequential-based d_F, behavior-based d_B) in Eqs. 2.1, 2.4 and 2.6 implies systematically the total time series elements x_{it} and thus, restricts the distance measures to capture local temporal differences. In this work, we provide a multi-scale framework for time series comparison using a hierarchical structure. Many methods exist in the literature such as the sliding window [Keo+03] or the dichotomy [DCA11]. We detail here the latter one.

segmentation localisée de la tempo

A multi-scale description can be obtained by repeatedly segmenting a time series expressed at a given temporal scale to induce its description at a more local level. Many approaches have been proposed assuming fixed either the number of the segments or their lengths [Fu11]. In this work, we consider a binary segmentation at each level. Let $I = [a; b]$ be a temporal interval of size $(b - a)$. The interval I is decomposed into two equal overlapped intervals I_L (left interval) and I_R (right interval). A parameter μ allows to overlap the two intervals I_L and I_R, covering discriminating subsequences in the central region of I (around $\frac{b+a}{2}$): $I = [a; b]; I_L = [a; a + \mu(b - a)]; I_R = [b - \mu(b - a); b]$. For $\mu = 0.6$, the overlap covers 10% of the size of the interval I. Then, the process is repeated on the intervals I_L and I_R. We obtain a set of intervals I_s illustrated in Fig. 3.6.

A multi-scale dissimilarity description between two time series is obtained by computing the usual time series metrics (d_A, d_B, d_F) on each of the resulting segments I_s. Note that for two time series $\mathbf{x}_i$ and $\mathbf{x}_j$, the comparison between $\mathbf{x}_i$ and $\mathbf{x}_j$ is done on the same interval I_s. For a multi-scale amplitude-based comparison based on binary segmentation, the set of involved amplitude-based measures $d_A^{I_s}$ is $\{d_A^{I_1}, d_A^{I_2}, \ldots\}$ where $d_A^{I_s}$ is defined as:

$$d_A^{I_s}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{t \in I_s} (x_{it} - x_{jt})^2} \tag{3.11}$$

The local behaviors- and frequential- based measures $d_B^{I_s}$ and $d_F^{I_s}$ are obtained similarly.

Our proposition is inspired from the LMNN framework where the optimization problem involves a pull term that penalizes large distances between sample of same labels ($Pull_i$). It can be interpreted as a regularization term on $Pull_i$. In LMNN, the push term penalizes small distances between samples of different labels ($Push_i$). It can be interpreted as a loss term on $Push_i$. To ensure a safety margin between similar and dissimilar samples, (in LMNN) a constraint is added: $D^2(\mathbf{x}_i, \mathbf{x}_l) - D^2(\mathbf{x}_i, \mathbf{x}_j) \geq 1 - \xi_{ijl}$.

pourquoi au carré ? D(xie) ?

Similarly, we formalize the M^{2}TML problem as an optimization problem involving both a **regularization term** on D and the pull set $Pull_i$, denoted $R_{Pull}(D)$, and a **loss term** on ξ and the push set $Push_i$, denoted $L_{Push}(\xi)$. **A set of constraints** is added to control the push term in order to have a large margin between $Pull_i$ and $Push_i$:

$$\begin{aligned} &\underset{D,\xi}{\operatorname{argmin}} \{R_{Pull}(D) + L_{Push}(\xi)\} \\ &\text{s.t. } \forall i, j \in Pull_i, l \in Push_i, \\ &\quad D(\mathbf{x}_{il}) - D(\mathbf{x}_{ij}) \geq 1 - \xi_{ijl} \\ &\quad \xi_{ijl} \geq 0 \end{aligned} \tag{3.12}$$

carré ? expliquer

Among the possibilities for the regularization term, we decide to choose to minimize the sum of the distances of the pull pairs. Among the possibilities for the loss term, we decide to choose to minimize the sum of the slack variables on the push pairs:

$$R_{Pull}(D) = \sum_{\substack{i \\ j \in Pull_i}} D(\mathbf{x}_{ij}) \tag{3.13}$$

$$L_{Push}(\xi) = \sum_{\substack{i \\ j \in Pull_i \\ l \in Push_i}} \xi_{ijl} \tag{3.14}$$

The M^{2}TML problem for large margin k-NN classification can be written as the following optimization problem:

$$\begin{aligned} &\underset{D,\xi}{\operatorname{argmin}} \left\{ \underbrace{\sum_{\substack{i \\ j \in Pull_i}} D(\mathbf{x}_{ij})}_{pull} + C \underbrace{\sum_{\substack{i \\ j \in Pull_i \\ l \in Push_i}} \xi_{ijl}}_{push} \right\} \\ &\text{s.t. } \forall i = 1, \ldots, n, \forall j \in Pull_i, l \in Push_i, \\ &\quad D(\mathbf{x}_{il}) - D(\mathbf{x}_{ij}) \geq 1 - \xi_{ijl} \\ &\quad \xi_{ijl} \geq 0 \end{aligned} \tag{3.15}$$

where ξ_{ijl} are the slack variables and C, the trade-off between the pull (regularization) and push (loss) costs. In the next section, we detail different strategies to define the $Pull_i$ and $Push_i$ sets.

3.4.2 Push and pull set definition

To build the pairwise training set, we associate for each $\mathbf{x}_i$, two sets, $Pull_i$ and $Push_i$, where the two sets are chosen according to one of the following strategies, illustrated in Fig 3.7. Recall that the norm $D_0(\mathbf{x}_{ij}) = ||\mathbf{x}_{ij}||_2$ is set as our initial distance D_0.

1. k-**NN vs impostors**: for a given $\mathbf{x}_i$, the sets of pairs to pull and to push corresponds respectively to: → définition de weinberger

$$\forall i \in 1,\dots,n, \quad Pull_i = \{\mathbf{x}_{ij} \ / \ y_j = y_i,\ D_0(\mathbf{x}_{ij}) \text{ is among the } k\text{-lowest distance}\} \tag{3.16}$$

$$Push_i = \{\mathbf{x}_{il} \ / \ y_l \neq y_i,\ D_0(\mathbf{x}_{il}) \leq \max_{\mathbf{x}_{ij} \in Pull_i} D_0(\mathbf{x}_{ij})\} \tag{3.17}$$

2. k-**NN vs all**: for a given $\mathbf{x}_i$, the sets of pairs to pull and to push corresponds respectively to:

$$\forall i \in 1,\dots,n, \quad Pull_i = \{\mathbf{x}_{ij} \ / \ y_j = y_i,\ D_0(\mathbf{x}_{ij}) \text{ is among the } k\text{-lowest distance}\} \tag{3.18}$$

$$Push_i = \{\mathbf{x}_{il} \ / \ y_l \neq y_i\} \tag{3.19}$$

pourquoi ? pour être sûr qu'un x_{il} pas imposteur ne le devienne

3. m-**NN**$^+$ **vs** m-**NN**$^-$: for a given $\mathbf{x}_i$, the pull and push sets are defined respectively as the set of the m-nearest neighbors of the same class ($y_j = y_i$), and the m-nearest neighbor of $\mathbf{x}_i$ of a different class ($y_j \neq y_i$). More precisely, our proposition states: $m = \alpha.k$ with $\alpha \geq 1$. Other propositions for m are possible:

pourquoi ? éviter qu'un x_{il} ne devienne un imposteur mais en équilibrant les push et les pull

$$\forall i \in 1,\dots,n, \quad Pull_i = \{\mathbf{x}_{ij} \ / \ y_j = y_i,\ D_0(\mathbf{x}_{ij}) \text{ is among the } m\text{-lowest distance}\} \tag{3.20}$$

$$Push_i = \{\mathbf{x}_{il} \ / \text{ s.t. } y_l \neq y_i,\ D_0(\mathbf{x}_{il}) \text{ is among the } m\text{-lowest distance}\} \tag{3.21}$$

On aurait dû noter $Pull = \bigcup Pull_i$ $Push = \bigcup Push_i$

In the following, we denote m-NN$^+ = \bigcup_i Pull_i$ and m-NN$^- = \bigcup_i Push_i$

mais ne change rien

Finally, let discuss about the similarities and differences between LMNN (Weinberger & Saul [WS09a]) and our M^{2}TML proposition. In LMNN, the sets $Pull_i$ and $Push_i$ are defined according the k-**NN vs impostors** strategy (Eqs. 3.16 & 3.17) and may be unbalanced. The sets are defined and fixed during the optimization process according to the initial metric D_0. In M^{2}TML the sets $Pull_i$ and $Push_i$ are defined according the m-**NN**$^+$ **vs** m-**NN**$^-$ strategy (Eqs. 3.20 & 3.21) and are balanced. The sets are defined and fixed during the optimization process according to the initial metric D_0, but the m-neighborhood is larger than the k-neighborhood. By considering a neighborhood larger than the k-neighborhood, we believe that the generalization properties of the learned metric D will be improved.

OK

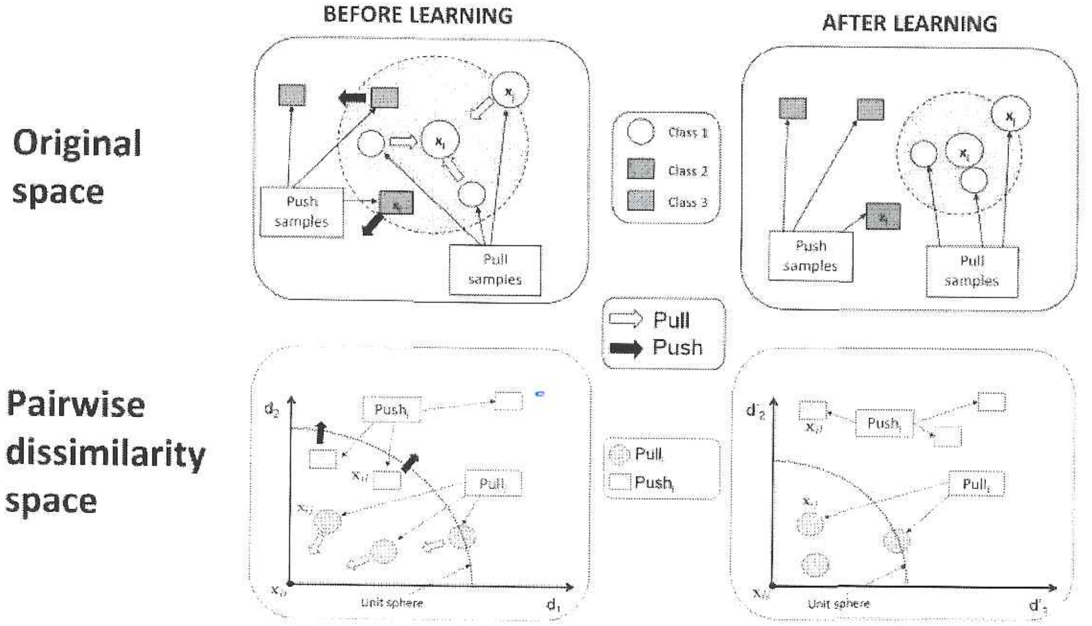


Figure 3.8: Metric learning problem in the original space (top) and the pairwise dissimilarity space (bottom) for a $k = 3$ neighborhood of $\mathbf{x}_i$. Before learning (left), push samples $\mathbf{x}_l$ invade the targets perimeter $\mathbf{x}_j$. In the dissimilarity pairwise space, this is equivalent to have push pairwise vectors $\mathbf{x}_{il}$ with an initial distance D_0 lower than the distance of pull pairwise vectors $\mathbf{x}_{ij}$. The aim of M²TML is to learn a metric D to push $\mathbf{x}_{il}$ (black arrow) and pull $\mathbf{x}_{ij}$ from the origin (white arrow).

- If $D(\mathbf{x}_{il}) < D(\mathbf{x}_{ij})$, then the pairs $\mathbf{x}_{il}$ is an imposter pair that invades the neighborhood of the target pairs $\mathbf{x}_{ij}$. The slack variable $\xi_{ijl} > 1$ will be penalized in the objective function.

- If $\cancel{D(\mathbf{x}_{il}) >}\ D(\mathbf{x}_{ij})$ ~~but~~ $\leq$ $D(\mathbf{x}_{il}) \leq D(\mathbf{x}_{ij}) + 1$, the pair $\mathbf{x}_{il}$ is within the safety margin of the target pairs $\mathbf{x}_{ij}$. The slack variable $\xi_{ijl} \in [0; 1]$ will have a small penalization effect in the objective function.

- If $D(\mathbf{x}_{il}) > D(\mathbf{x}_{ij}) + 1$, $\xi_{ijl} = 0$ and the slack variable has no effect in the objective function.

In the following, we propose different regularizers for the pull term $R_{Pull}(D)$. First, we use a linear regularization. Secondly, we use a quadratic regularization that enables to extend the approach to learn non-linear function for D by using the "kernel" trick. Thirdly, we formulate the problem as a SVM problem to solve a large margin problem between $\cup Pull_i$ and $\cup Push_i$ sets, and then, we define the combined metric D based on the SVM solution. Finally, we sum up the retained solution (SVM-based solution) and give the main steps of the algorithm.

union

on a trouvé w par SVM → comment définir métrique D(x_i,test)

3.7.2 Solution for the linearly separable Pull and Push sets

Let $\mathbf{x}_{test}$ be a new sample, $\mathbf{x}_{i,test} \in \mathscr{E}$ gives the proximity between $\mathbf{x}_i$ and $\mathbf{x}_{test}$ based on the p multi-modal and multi-scale metrics d_h. We review in this section different interpretations in the dissimilarity space.

objectif : trouver le y test par k-NN.
Calcul de la proximité de x test avec chacun des x i

M^{2}TML metric definition

|D(x test,i)| x test i

Given a test pair $\mathbf{x}_{i,test}$, the norm of the pair allows to estimate the proximity between $\mathbf{x}_i$ and $\mathbf{x}_{test}$. In particular, for M^{2}TML, two quantities are used to define the dissimilarity measure: the projected norm and the distance to the margin.

l'hyperplan ⊥ à w sépare les push et les pull

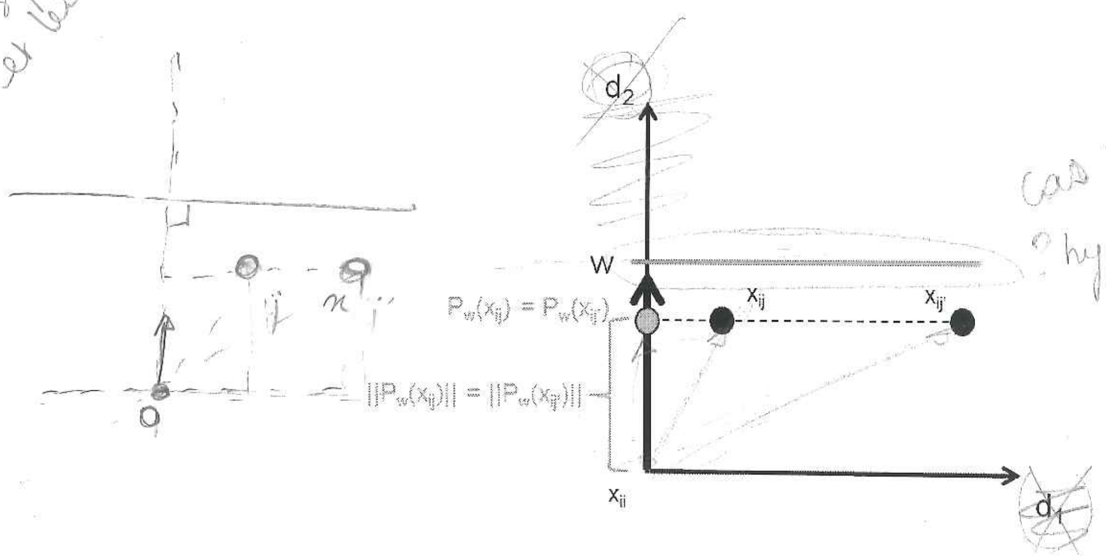
Let denote $\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test})$, the orthogonal projection of $\mathbf{x}_{i,test}$ on the axis of direction $\mathbf{w}$:

$$\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test}) = \frac{< \mathbf{w}, \mathbf{x}_{i,test} >}{||\mathbf{w}||^2}\mathbf{w} = \frac{\mathbf{w}^T\mathbf{x}_{i,test}}{||\mathbf{w}||^2}\mathbf{w} \tag{3.47}$$

The projected norm $||\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test})||$ of $\mathbf{x}_{i,test}$ on the direction $\mathbf{w}$ limits the comparison of $\mathbf{x}_i$ and $\mathbf{x}_{test}$ to the features separating pull and push sets (Fig. 3.9), it is defined as:

$$||\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test})|| = \frac{|\mathbf{w}^T\mathbf{x}_{i,test}|}{||\mathbf{w}||} \tag{3.48}$$

Fig 3.10 a)



cas particulier ? hyperplan

Figure 3.9: The projected vector $\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{ij})$ and $\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{ij'})$

Although the norm $||\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test})||$ satisfies positivity, it doesn't guarantee lower distances for pull pairs than for push pairs as illustrated in Fig 3.10.

Note that the distance of the projection to the margin $\mathbf{w}^T\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test}) + b$ gives the membership of the projected vector $\mathbf{P}_{\mathbf{w}}(\mathbf{x}_{i,test})$ in the pull or push side. However, it can't used as a dissimilarity (non-positivity).

3.8 SVM-based solution and algorithm for M²TML

In this section, we review the main steps of the retained SVM solution. In particular, we detail two pre-processing steps needed to adapt the SVM framework to our metric learning problem that are the pairwise space normalization and the neighborhood scaling.

Pairwise space normalization. The scale between the p basic metrics d_h can be different. Thus, there is a need to scale the data within the pairwise space and ensure comparable ranges for the p basic metrics d_h. In our experiment, we use dissimilarity measures with values in $[0; +\infty[$. Therefore, we propose to Z-normalize their log distributions as explained in Section 2.5.2.

ok

Neighborhood scaling. In real datasets, local neighborhoods may have very different scales as illustrated in Fig. 3.12. To make the pull neighborhood spreads comparable, we propose for each $\mathbf{x}_i$ to scale each pairs $\mathbf{x}_{ij}$ such that the L_2 norm (radius) of the farthest m-th nearest neighbor is 1:

$$\mathbf{x}_{ij}^{norm} = \left[\frac{d_1(\mathbf{x}_i,\mathbf{x}_j)}{r_i}, \ldots, \frac{d_p(\mathbf{x}_i,\mathbf{x}_j)}{r_i}\right]^T \tag{3.52}$$

where r_i is the radius associated to $\mathbf{x}_i$ corresponding to the maximum norm of its m-th nearest neighbor of same class in $Pull_i$:

$$r_i = \max_{\mathbf{x}_{ij} \in Pull_i} D_0(\mathbf{x}_{ij}) \tag{3.53}$$

Il me semble qu'on l'a déjà dit, mais je ne retrouve pas où 3.7.1.

For simplification purpose, we denote $\mathbf{x}_{ij}$ as $\mathbf{x}_{ij}^{norm}$. Fig. 3.12 illustrates the effect of neighborhood scaling in the dissimilarity space.

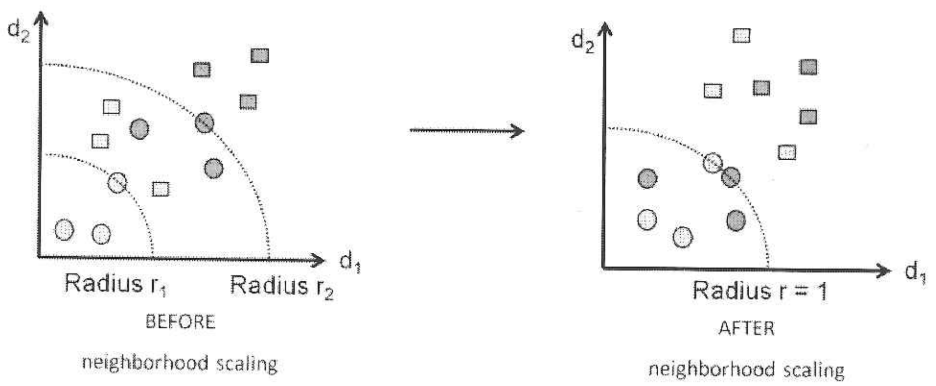


Figure 3.12: Effect of neighborhood scaling before (left) and after (right) on the neighborhood of two time series $\mathbf{x}_1$ (green) and $\mathbf{x}_2$ (red). Circle represent pairs $Pull_i$ and square represents pairs $Push_i$ for $m = 3$ neighbors. Before scaling, the problem is not linearly separable with a global SVM approach and the spread of each neighborhood are not comparable. After scaling, the target neighborhood becomes comparable and in this example, the problem becomes linearly separable.