Regressive Virtual Metric Learning

Michaël Perrot and Amaury Habrard
Université de Lyon, Université Jean Monnet de Saint-Etienne,
Laboratoire Hubert Curien, CNRS, UMR5516, F-42000, Saint-Etienne, France.
{michael.perrot,amaury.habrard}@univ-st-etienne.fr

Abstract

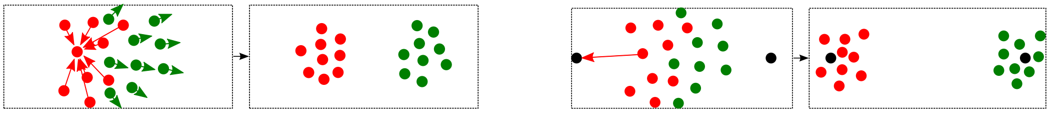
We are interested in supervised metric learning of Mahalanobis like distances. Existing approaches mainly focus on learning a new distance using similarity and dissimilarity constraints between examples. In this paper, instead of bringing closer examples of the same class and pushing far away examples of different classes we propose to move the examples with respect to virtual points. Hence, each example is brought closer to a a priori defined virtual point reducing the number of constraints to satisfy. We show that our approach admits a closed form solution which can be kernelized. We provide a theoretical analysis showing the consistency of the approach and establishing some links with other classical metric learning methods. Furthermore we propose an efficient solution to the difficult problem of selecting virtual points based in part on recent works in optimal transport. Lastly, we evaluate our approach on several state of the art datasets.

1 Introduction

The goal of a metric learning algorithm is to capture the idiosyncrasies in the data mainly by defining a new space of representation where some semantic constraints between examples are fulfilled. In the previous years the main focus of metric learning algorithms has been to learn Mahalanobis like distances of the form $d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x}-\mathbf{x}')^T\mathbf{M}(\mathbf{x}-\mathbf{x}')}$ where $\mathbf{M}$ is a positive semi-definite matrix (PSD) defining a set of parameters[1]. Using a Cholesky decomposition $\mathbf{M} = \mathbf{L}\mathbf{L}^T$, one can see that this is equivalent to learn a linear transformation from the input space.

Most of the existing approaches in metric learning use constraints of type must-link and cannot-link between learning examples [1, 2]. For example, in a supervised classification task, the goal is to bring closer examples of the same class and to push far away examples of different classes. The idea is that the learned metric should affect a high value to dissimilar examples and a low value to similar examples. Then, this new distance can be used in a classification algorithm like a nearest neighbor classifier. Note that in this case the set of constraints is quadratic in the number of examples which can be prohibitive when the number of examples increases. One heuristic is then to select only a subset of the constraints but selecting such a subset is not trivial. In this paper, we propose to consider a new kind of constraints where each example is associated with an a priori defined virtual point. It allows us to consider the metric learning problem as a simple regression where we try to minimize the differences between learning examples and virtual points. Fig. 1 illustrates the differences between our approach and a classical metric learning approach. It can be noticed that our algorithm only uses a linear number of constraints. However defining these constraints by hand can be tedious and difficult. To overcome this problem, we present two approaches to automatically define them. The first one is based on some recent advances in the field of Optimal Transport while the second one uses a class-based representation space.

[1] When $\mathbf{M} = \mathbf{I}$, the identity matrix, it corresponds to the Euclidean distance.


(a) Classical must-link cannot-link approach.

(b) Our virtual point-based regression formulation.

Figure 1: Arrows denote the constraints used by each approach for one particular example in a binary classification task. The classical metric learning approach in Fig. 1(a) uses $\mathcal{O}(n^2)$ constraints bringing closer examples of the same class and pushing far away examples of different classes. On the contrary, our approach presented in Fig. 1(b) moves the examples to the neighborhood of their corresponding virtual point, in black, using only $\mathcal{O}(n)$ constraints. (Best viewed in color)

Moreover, thanks to its regression-based formulation, our approach can be easily kernelized allowing us to deal efficiently with non linear transformations which is a nice advantage in comparison to some metric learning methods. We also provide a theoretical analysis showing the consistency of our approach and establishing some relationships with a classical metric learning formulation.

This paper is organized as follows. In Section 2 we identify several related works. Then in Section 3 we present our approach, provide some theoretical results and give two solutions to generate the virtual points. Section 4 is dedicated to an empirical evaluation of our method on several widely used datasets. Finally, we conclude in Section 5.

2 Related work

For up-to-date surveys on metric learning see [3] and [4]. In this section we focus on algorithms which are more closely related to our approach. First of all, one of the most famous approach in metric learning is LMNN [5] where the authors propose to learn a PSD matrix to improve the k-nearest-neighbours algorithm. In their work, instead of considering pairs of examples, they use triplets ($\mathbf{x}_i$, $\mathbf{x}_j$, $\mathbf{x}_k$) where $\mathbf{x}_j$ and $\mathbf{x}_k$ are in the neighborhood of $\mathbf{x}_i$ and such that $\mathbf{x}_i$ and $\mathbf{x}_j$ are of the same class and $\mathbf{x}_k$ is of a different class. The idea is then to bring closer $\mathbf{x}_i$ and $\mathbf{x}_j$ while pushing $\mathbf{x}_k$ far away. Hence, if the number of constraints seems to be cubic, the authors propose to only consider triplets of examples which are already close to each other. In contrast, the idea presented in [6] is to collapse all the examples of the same class in a single point and to push infinitely far away examples of different classes. The authors define a measure to estimate the probability of having an example $\mathbf{x}_j$ given an example $\mathbf{x}_i$ with respect to a learned PSD matrix $\mathbf{M}$. Then, they minimize, *w.r.t.* $\mathbf{M}$, the KL divergence between this measure and the best case where the probability is 1 if the two examples are of the same class and 0 otherwise. It can be seen as collapsing all the examples of the same class on an implicit virtual point. In this paper we use several explicit virtual points and we collapse the examples on these points with respect to their classes and their distances to them.

A recurring issue in Mahalanobis like metric learning is to fulfill the PSD constraint on the learned metric. Indeed, projecting a matrix on the PSD cone is not trivial and generally requires a costly eigenvalues decomposition. To address this problem, in ITML [1] the authors propose to use a LogDet divergence as the regularization term. The idea is to learn a matrix which is close to an a priori defined PSD matrix. The authors then show that if the divergence is finite, then the learned matrix is guaranteed to be PSD. Another approach, as proposed in [2], is to learn a matrix $\mathbf{L}$ such that $\mathbf{M} = \mathbf{L}\mathbf{L}^T$, *i.e.* instead of learning the metric the authors propose to learn the projection. The main drawback is the fact that most of the time the resulting optimization problem is not convex [3, 4, 7] and is thus harder to optimize. In this paper, we are also interested in learning $\mathbf{L}$ directly. However, because we are using constraints between examples and virtual points, we obtain a convex problem with a closed form solution allowing us to learn the metric in an efficient way.

The problem of learning a metric such that the induced space is not linearly dependent of the input space has been addressed in several works before. First, it is possible to directly learn an intrinsically non linear metric as in χ^2-LMNN [8] where the authors propose to learn a χ^2 distance rather than a Mahalanobis distance. This distance is particularly relevant for histograms comparisons. Note that this kind of approaches is close to the kernel learning problem which is beyond the scope of this work. Second, another solution used by local metric learning methods is to split the input space in several regions and to learn a metric in each region to introduce some non linearity, as in MM-LMNN [7]. Similarly, in GB-LMNN [8] the authors propose to locally refine the metric learned by LMNN by successively splitting the input space. A third kind of approach tries to project the learning examples in a new space which is non linearly dependent of the input space. It can be done in two ways, either by projecting a priori the learning examples in a new space with a KPCA [9] or by rewriting the optimization problem in a kernelized form [1]. The first approach allows one to include non linearity in most of the metric learning algorithms but imposes to select the interesting features beforehand. The second method can be difficult to use as rewriting the optimization problem is most of the times non trivial [4]. Indeed, if one wants to use the kernel trick it implies that the access to the learning examples should only be done through dot products which is difficult when working with pairs of examples as it is the case in metric learning. In this paper we show that using virtual points chosen in a given target space allows us to kernelize our approach easily and thus to work in a very high dimensional space without using an explicit projection thanks to the kernel trick.

Our method is based on a regression and can thus be linked, in its kernelized form, to several approaches in kernelized regression for structured output [10, 11, 12]. The idea behind these approaches is to minimize the difference between input examples and output examples using kernels, *i.e.* working in a high dimensional space. In our case, the learning examples can be seen as input examples and the virtual points as output examples. However, we only project the learning examples in a high dimensional space, the virtual points already belong to the output space. Hence, we do not have the pre-image problem [12]. Furthermore, our goal is not to predict a virtual point but to learn a metric between examples and thus, after the learning step, the virtual points are discarded.

3 Contributions

The main idea behind our algorithm is to bring closer the learning examples to a set of virtual points. We present this idea in three subsections. First we assume that we have access to a set of n learning pairs ($\mathbf{x}$,$\mathbf{v}$) where $\mathbf{x}$ is a learning example and $\mathbf{v}$ is a virtual point associated to $\mathbf{x}$ and we present both the linear and kernelized formulations of our approach called RVML. It boils down to solve a regression in closed form, the main originality being the introduction of virtual points. In the second subsection, we show that it is possible to theoretically link our approach to a classical metric learning one based on [13]. In the last subsection, we propose two automatic methods to generate the virtual points and to associate them with the learning examples.

3.1 Regressive Virtual Metric Learning (RVML)

Given a probability distribution $\mathcal{D}$ defined over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y}$ is a finite label set, let $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be a set of examples drawn i.i.d. from $\mathcal{D}$. Let $f_\mathbf{v} : \mathcal{X} \times \mathcal{Y} \to \mathcal{V}$ where $\mathcal{V} \subseteq \mathbb{R}^{d'}$ be the function which associates each example to a virtual point. We consider the learning set $S_\mathbf{v} = \{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^n$ where $\mathbf{v}_i = f_\mathbf{v}(\mathbf{x}_i, y_i)$. For the sake of simplicity denote by $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$ and $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_n)^T$ the matrices containing respectively one example and the associated virtual point on each line. In this section, we consider that the function $f_\mathbf{v}$ is known. We come back to its definition in Section 3.3. Let $\|\cdot\|_\mathcal{F}$ be the Frobenius norm and $\|\cdot\|_2$ be the l_2 vector norm. Our goal is to learn a matrix $\mathbf{L}$ such that $\mathbf{M} = \mathbf{L}\mathbf{L}^T$ and for this purpose we consider the following optimisation problem:

$$\min_{\mathbf{L}} f(\mathbf{L}, \mathbf{X}, \mathbf{V}) = \min_{\mathbf{L}} \frac{1}{n} \|\mathbf{X}\mathbf{L} - \mathbf{V}\|_\mathcal{F}^2 + \lambda \|\mathbf{L}\|_\mathcal{F}^2. \tag{1}$$

The idea is to learn a new space of representation where each example is close to its associated virtual point. Note that $\mathbf{L}$ is a $d \times d'$ matrix and if $d' < d$ we also perform dimensionality reduction.

Theorem 1. *The optimal solution of Problem 1 can be found in closed form. Furthermore, we can derive two equivalent solutions:*

$$\mathbf{L} = \left(\mathbf{X}^T\mathbf{X} + \lambda n \mathbf{I}\right)^{-1} \mathbf{X}^T\mathbf{V} \tag{2}$$

$$\mathbf{L} = \mathbf{X}^T \left(\mathbf{X}\mathbf{X}^T + \lambda n \mathbf{I}\right)^{-1} \mathbf{V}. \tag{3}$$

Proof. The proof of this theorem can be found in the supplementary material. □

From Eq. 2 we deduce the matrix $\mathbf{M}$:

$$\mathbf{M} = \mathbf{L}\mathbf{L}^T = \left(\mathbf{X}^T\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{V}\mathbf{V}^T\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}. \tag{4}$$

Note that $\mathbf{M}$ is PSD by construction: $\mathbf{x}^T\mathbf{M}\mathbf{x} = \mathbf{x}^T\mathbf{L}\mathbf{L}^T\mathbf{x} = \|\mathbf{L}^T\mathbf{x}\|_2^2 \geq 0$.

So far, we have focused on the linear setting. We now present a kernelized version, showing that it is possible to learn a metric in a very high dimensional space without an explicit projection.

Let $\phi(\mathbf{x})$ be a projection function and $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}')$ be its associated kernel. For the sake of readability, let $K_\mathbf{X} = \phi(\mathbf{X})\phi(\mathbf{X})^T$ where $\phi(\mathbf{X}) = (\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n))^T$. Given the solution matrix $\mathbf{L}$ presented in Eq. 3, we have $\mathbf{M} = \mathbf{X}^T\left(\mathbf{X}\mathbf{X}^T + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}\mathbf{V}^T\left(\mathbf{X}\mathbf{X}^T + \lambda n\mathbf{I}\right)^{-1}\mathbf{X}$. Then, $\mathbf{M}_K$ the kernelized version of the matrix $\mathbf{M}$ is defined such that:

$$\mathbf{M}_K = \phi(\mathbf{X})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}\mathbf{V}^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\phi(\mathbf{X}).$$

The squared Mahalanobis distance can be written as $d^2_\mathbf{M}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{M}\mathbf{x} + \mathbf{x}'^T\mathbf{M}\mathbf{x}' - 2\mathbf{x}^T\mathbf{M}\mathbf{x}'$. Thus we can obtain $d^2_{\mathbf{M}_K}(\phi(\mathbf{x}), \phi(\mathbf{x}')) = \phi(\mathbf{x})^T\mathbf{M}_K\phi(\mathbf{x}) + \phi(\mathbf{x}')^T\mathbf{M}_K\phi(\mathbf{x}') - 2\phi(\mathbf{x})^T\mathbf{M}_K\phi(\mathbf{x}')$ the kernelized version by considering that:

$$\begin{aligned}\phi(\mathbf{x})^T\mathbf{M}_K\phi(\mathbf{x}) &= \phi(\mathbf{x})^T\phi(\mathbf{X})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}\mathbf{V}^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\phi(\mathbf{X})\phi(\mathbf{x})\\ &= K_\mathbf{X}(\mathbf{x})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}\mathbf{V}^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}K_\mathbf{X}(\mathbf{x})\end{aligned}$$

where $K_\mathbf{X}(\mathbf{x}) = (K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n))^T$ is the similarity vector to the examples *w.r.t.* K.

Note that it is also possible to obtain a kernelized version of $\mathbf{L}$: $\mathbf{L}_K = \phi(\mathbf{X})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}$.

This result is close to a previous one already derived in [11] in a structured output setting. The main difference is the fact that we do not use a kernel on the output (the virtual points here). Hence, it is possible to compute the projection of an example $\mathbf{x}$ of dimension d in a new space of dimension d':

$$\phi(\mathbf{x})\mathbf{L}_K = \phi(\mathbf{x})^T\phi(\mathbf{X})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V} = K_\mathbf{X}(\mathbf{x})^T\left(K_\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{V}.$$

Recall that in this work we are interested in learning a distance between examples and not in the prediction of the virtual points which only serve as a way to bring closer similar examples and push far away dissimilar examples.

From a complexity standpoint, we can see that, assuming the kernel function as easy to calculate, the main bottleneck when computing the solution in closed form is the inversion of a $n \times n$ matrix.

3.2 Theoretical Analysis

In this section, we propose to theoretically show the interest of our approach by proving (i) that it is consistent and (ii) that it is possible to link it to a more classical metric learning formulation.

3.2.1 Consistency

Let $l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) = \|\mathbf{x}^T\mathbf{L} - \mathbf{v}^T\|_2^2$ be our loss and let $\mathcal{D}_\mathbf{v}$ be the probability distribution over $\mathcal{X} \times \mathcal{V}$ such that $p_{\mathcal{D}_\mathbf{v}}(\mathbf{x}, \mathbf{v}) = p_\mathcal{D}(\mathbf{x}, y | \mathbf{v} = f_\mathbf{v}(\mathbf{x}, y))$. Showing the consistency boils down to bound with high probability the true risk, denoted by $R(\mathbf{L})$, by the empirical risk, denoted by $\hat{R}(\mathbf{L})$ such that:

$$R(\mathbf{L}) = \mathbb{E}_{(\mathbf{x},\mathbf{v})\sim\mathcal{D}_\mathbf{v}} l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) \text{ and } \hat{R}(\mathbf{L}) = \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) = \frac{1}{n}\|\mathbf{X}\mathbf{L} - \mathbf{V}\|_\mathcal{F}^2.$$

The empirical risk corresponds to the error of the learned matrix $\mathbf{L}$ on the learning set $S_\mathbf{v}$. The true risk is the error of $\mathbf{L}$ on the unknown distribution $\mathcal{D}_\mathbf{v}$. The consistency property ensures that with a sufficient number of examples a low empirical risk implies a low true risk with high probability. To show that our approach is consistent, we use the uniform stability framework [14].

Theorem 2. *Let $\|\mathbf{v}\|_2 \leq C_\mathbf{v}$ for any $\mathbf{v} \in \mathcal{V}$ and $\|\mathbf{x}\|_2 \leq C_\mathbf{x}$ for any $\mathbf{x} \in \mathcal{X}$. With probability $1 - \delta$, for any matrix $\mathbf{L}$ optimal solution of Problem 1, we have:*

$$R(\mathbf{L}) \leq \hat{R}(\mathbf{L}) + \frac{8C_\mathbf{v}^2C_\mathbf{x}^2}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2 + \left(\left(\frac{16C_\mathbf{x}^2}{\lambda} + 1\right)C_\mathbf{v}^2\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2\right)\sqrt{\frac{\ln\frac{1}{\delta}}{2n}}.$$

Proof. The proof of this theorem can be found in the supplementary material. □

We obtain a rate of convergence in $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ which is standard with this kind of bounds.

3.2.2 Link with a Classical Metric Learning Formulation

In this section we show that it is possible to bound the true risk of a classical metric learning approach with the empirical risk of our formulation. Most of the classical metric learning approaches make use of a notion of margin between similar and dissimilar examples. Hence, similar examples have to be close to each other, *i.e.* at a distance smaller than a margin γ_1, and dissimilar examples have to be far from each other, *i.e.* at a distance greater than a margin γ_{-1}. Let $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$ be two examples from $\mathcal{X} \times \mathcal{Y}$, using this notion of margin, we consider the following loss [13]:

$$l(\mathbf{L}, (\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)) = \left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ \tag{5}$$

where $y_{ij} = 1$ if $y_i = y_j$ and -1 otherwise, $[z]_+ = \max(0, z)$ is the hinge loss and $\gamma_{y_{ij}}$ is the desired margin between examples. As introduced before, we consider that $\gamma_{y_{ij}}$ takes a big value when the examples are dissimilar, *i.e.* when $y_{ij} = -1$, and a small value when the examples are similar, *i.e.* when $y_{ij} = 1$. In the following we show that, relating the notion of margin to the distances between virtual points, it is possible to bound the true risk associated with this loss by the empirical risk of our approach with respect to a constant.

Theorem 3. *Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{V} \subset \mathbb{R}^{d'}$ be a finite set of virtual points and $f_\mathbf{v}$ is defined as $f_\mathbf{v}(\mathbf{x}_i, y_i) = \mathbf{v}_i$, $\mathbf{v}_i \in \mathcal{V}$. Let $\|\mathbf{v}\|_2 \leq C_\mathbf{v}$ for any $\mathbf{v} \in \mathcal{V}$ and $\|\mathbf{x}\|_2 \leq C_\mathbf{x}$ for any $\mathbf{x} \in \mathcal{X}$. Let $\gamma_1 = 2\max_{\mathbf{x}_k,\mathbf{x}_l,y_{kl}=1} d^2(\mathbf{v}_k, \mathbf{v}_l)$ and $\gamma_{-1} = \frac{1}{2}\min_{\mathbf{x}_k,\mathbf{x}_l,y_{kl}=-1} d^2(\mathbf{v}_k, \mathbf{v}_l)$, we have:*

$$\begin{aligned}&\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}}\left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ \\ &\quad\leq 8\left(\hat{R}(\mathbf{L}) + \frac{8C_\mathbf{v}^2C_\mathbf{x}^2}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2 + \left(\left(\frac{16C_\mathbf{x}^2}{\lambda} + 1\right)C_\mathbf{v}^2\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2\right)\sqrt{\frac{\ln\frac{1}{\delta}}{2n})}\right).\end{aligned}$$

Proof. The proof of this theorem can be found in the supplementary material. □

In Theorem 3, we can notice that the margins are related to the distances between virtual points and correspond to the ideal margins, *i.e.* the margins that we would like to achieve after the learning step. Aside this remark, we can define $\hat{\gamma}_1$ and $\hat{\gamma}_{-1}$ the observed margins obtained after the learning step: All the similar examples are in a sphere centered in their corresponding virtual point and of diameter $\hat{\gamma}_1 = 2\max_{(\mathbf{x},\mathbf{v})}\left\|\mathbf{x}^T\mathbf{L} - \mathbf{v}^T\right\|_2$. Similarly, the distance between hyperspheres of dissimilar examples is $\hat{\gamma}_{-1} = \min_{\mathbf{v},\mathbf{v}',\mathbf{v}\neq\mathbf{v}'}\left\|\mathbf{v} - \mathbf{v}'\right\|_2 - \hat{\gamma}_1$. As a consequence, even if we do not use cannot-link constraints our algorithm is able to push reasonably far away dissimilar examples.

In the next subsection we present two different methods to select the virtual points.

3.3 Virtual Points Selection

Previously, we assumed to have access to the function $f_\mathbf{v} : \mathcal{X} \times \mathcal{Y} \to \mathcal{V}$. In this subsection, we present two methods for generating automatically the set of virtual points and the mapping $f_\mathbf{v}$.

3.3.1 Using Optimal Transport on the Learning Set

In this first approach, we propose to generate the virtual points by using a recent variation of the Optimal Transport (OT) problem [15] allowing one to transport some examples to new points corresponding to a linear combination of a set of known instances. These new points will actually correspond to our virtual points. Our approach works as follows. We begin by extracting a set of landmarks S' from the training set S. For this purpose, we use an adaptation of the landmark selection method proposed in [16] allowing us to take into account some diversity among the landmarks. To avoid to fix the number of landmarks in advance, we have just replaced it by a simple heuristic saying that the number of landmarks must be greater than the number of classes and that the maximum distance between an example and a landmark must be lower than the mean of all pairwise

Algorithm 1: Selecting S' from a set of examples S.

```
input  : $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ a set of examples; $\mathcal{Y}$ the label set.
output: $S'$ a subset of $S$
begin
    $\mu =$ mean of distances between all the examples of $S$
    $\mathbf{x}_{\max} = \arg\max_{\mathbf{x}\in S} \|\mathbf{x} - \mathbf{0}\|_2$
    $S' = \{\mathbf{x}_{\max}\}$; $S = S \setminus S'$
    $\varepsilon = \max_{\mathbf{x}\in S} \min_{\mathbf{x}'\in S'} \|\mathbf{x} - \mathbf{x}'\|_2$
    while $|S'| < |\mathcal{Y}|$ or $\varepsilon > \mu$ do
        $\mathbf{x}_{\max} = \arg\max_{\mathbf{x}\in S} \sum_{\mathbf{x}'\in S'} \|\mathbf{x} - \mathbf{x}'\|_2$
        $S' = S' \cup \{\mathbf{x}_{\max}\}$; $S = S \setminus S'$
        $\varepsilon = \max_{\mathbf{x}\in S} \min_{\mathbf{x}'\in S'} \|\mathbf{x} - \mathbf{x}'\|_2$
```

distances from the training set -allowing us to have a fully automatic procedure. It is summarized in Algorithm 1.

Then we compute an optimal transport from the training set S to the landmark set S'. For this purpose, we create a real matrix $\mathbf{C}$ of size $|S| \times |S'|$ giving the cost to transport one training instance to a landmark such that $\mathbf{C}(i, j) = \|\mathbf{x}_i - \mathbf{x}'_j\|_2$ with $\mathbf{x}_i \in S$ and $\mathbf{x}'_j \in S'$. The optimal transport is found by learning a matrix $\gamma \in \mathbb{R}^{|S|\times|S'|}$ able to minimize the cost of moving training examples to the landmark points. Let $\mathbf{S}'$ be the matrix of landmark points (one per line), the transport *w.r.t.* γ of any training instance $(\mathbf{x}_i, y_i)$ gives a new virtual point such that $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \gamma(i)\mathbf{S}'$, $\gamma(i)$ designing the i^{th} line of γ. Note that this new virtual point is a linear combination of the landmark instances to which the example is transported. The set of virtual points is then defined by $\mathbf{V} = \gamma\mathbf{S}'$. The virtual points are thus not defined a priori but are automatically learned by solving a problem of optimal transport. Note that this transportation mode is potentially non linear since there is no guarantee that there exists a matrix $\mathbf{T}$ such that $\mathbf{V} = \mathbf{XT}$. Our metric learning approach can, in this case, be seen as a an approximation of the result given by the optimal transport.

To learn γ, we use the following optimization problem proposed in [17]:

$$\arg\min_{\gamma} \langle \gamma, \mathbf{C} \rangle_{\mathcal{F}} - \frac{1}{\lambda} h(\gamma) + \eta \sum_{j} \sum_{c} \|\gamma(y_i = c, j)\|_q^p$$

where $h(\gamma) = -\sum_{i,j} \gamma(i, j) \log(\gamma(i, j))$ is the entropy of γ that allows to solve the transportation problem efficiently with the Sinkhorn-Knopp algorithm [18]. The second regularization term, where $\gamma(y_i = c, j)$ corresponds to the lines of the j^{th} column of γ where the class of the input is c, has been introduced in [17]. The goal of this term is to prevent input examples of different classes to move toward the same output examples by promoting group sparsity in the matrix γ thanks to $\|\cdot\|_q^p$ corresponding to a l_q-norm to the power of p used here with $q = 1$ and $p = \frac{1}{2}$.

3.3.2 Using a Class-based Representation Space

For this second approach, we propose to define virtual points as the unit vectors of a space of dimension $|\mathcal{Y}|$. Let $\mathbf{e}_j \in \mathbb{R}^{|\mathcal{Y}|}$ be such a unit vector $(1 \leq j \leq |\mathcal{Y}|)$ -*i.e.* a vector where all the attributes are 0 except for one attribute j which is set to 1- to which we associate a class label from $\mathcal{Y}$. Then, for any learning example $(\mathbf{x}_i, y_i)$, we define $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \mathbf{e}_{\#y_i}$ where $\#y_i = j$ if $\mathbf{e}_j$ is mapped with the class y_i. Thus, we have exactly $|\mathcal{Y}|$ virtual points, each one corresponding to a unit vector and a class label. We call this approach the class-based representation space method. If the number of classes is smaller than the number of dimensions used to represent the learning examples, then our method will perform dimensionality reduction for free. Furthermore, our approach will try to project all the examples of one class on the same axis while examples of other classes will tend to be projected on different axes. The underlying intuition behind the new space defined by $\mathbf{L}$ is to make each attribute discriminant for one class.

Table 1: Comparison of our approach with several baselines in the linear setting.

	Baselines			Our approach	
Base	1NN	LMNN	SCML	RVML-Lin-OT	RVML-Lin-Class
Amazon	41.51 ± 3.24	65.50 ± 2.28	71.68 ± 1.86	71.62 ± 1.34	**73.09 ± 2.49**
Breast	95.49 ± 0.79	95.49 ± 0.89	**96.50 ± 0.64***	95.24 ± 1.21	95.34 ± 0.95
Caltech	18.04 ± 2.20	49.68 ± 2.76	52.84 ± 1.61	52.51 ± 2.41	**55.41 ± 2.55***
DSLR	29.61 ± 4.38	**76.08 ± 4.79**	65.10 ± 9.00	74.71 ± 5.27	75.29 ± 5.08
Ionosphere	86.23 ± 1.95	88.02 ± 3.02	**90.38 ± 2.55***	87.36 ± 3.12	82.74 ± 2.81
Isolet	88.97	**95.83**	89.61	91.40	94.61
Letters	94.74 ± 0.27	**96.43 ± 0.28***	96.13 ± 0.20	90.25 ± 0.60	95.51 ± 0.26
Pima	69.91 ± 1.69	70.04 ± 2.20	69.22 ± 2.60	**70.48** ± 3.19	69.57 ± 2.85
Scale	78.68 ± 2.66	78.20 ± 1.91	**93.39 ± 1.70***	90.05 ± 2.13	87.94 ± 1.99
Splice	71.17	82.02	**85.43**	84.64	78.44
Svmguide1	**95.12**	95.03	87.38	94.83	85.25
Wine	96.18 ± 1.59	98.36 ± 1.03	96.91 ± 1.93	**98.55 ± 1.67**	98.18 ± 1.48
Webcam	42.90 ± 4.19	85.81 ± 3.75	**90.43 ± 2.70**	88.60 ± 3.63	88.60 ± 2.69
mean	69.89	82.81	83.46	**83.86**	83.07

4 Experimental results

In this section, we evaluate our approach on 13 different datasets coming from either the UCI [19] repository or used in recent works in metric learning [8, 20, 21]. For isolet, splice and svmguide1 we have access to a standard training/test partition, for the other datasets we use a 70% training/30% test partition, we perform the experiments on 10 different splits and we average the result. We normalize the examples with respect to the training set by subtracting for each attribute its mean and dividing by 3 times its standard deviation. We set our regularization parameter λ with a 5-fold cross validation. After the metric learning step, we use a 1-nearest neighbor classifier to assess the performance of the metric and report the accuracy obtained.

We perform two series of experiments. First, we consider our linear formulation used with the two virtual points selection methods presented in this paper: RVML-Lin-OT based on Optimal Transport (Section 3.3.1) and RVML-Lin-Class using the class-based representation space method (Section 3.3.2). We compare them to a 1-nearest neighbor classifier without metric learning (1NN), and with two state of the art linear metric learning methods: LMNN [5] and SCML [20].
In a second series, we consider the kernelized versions of RVML, namely RVML-RBF-OT and RVML-RBF-Class, based respectively on Optimal Transport and class-based representation space methods, with a RBF kernel with the parameter σ fixed as the mean of all pairwise training set Euclidean distances [16]. We compare them to non linear methods using a KPCA with a RBF kernel[2] as a pre-process: 1NN-KPCA a 1-nearest neighbor classifier without metric learning and LMNN-KPCA corresponding to LMNN in the KPCA-space. The number of dimensions is fixed as the one of the original space for high dimensional datasets (more than 100 attributes), to 3 times the original dimension when the dimension is smaller (between 5 and 100 attributes) and to 4 times the original dimension for the lowest dimensional datasets (less than 5 attributes). We also consider some local metric learning methods: GBLMNN [8] a non linear version of LMNN and SCMLLOCAL [20] the local version of SCML. For all these methods, we use the implementations available online letting them handle hyper-parameters tuning.

The results for linear methods are presented in Table 1 while Table 2 gives the results obtained with the non linear approaches. In each table, the best result on each line is highlighted with a bold font while the second best result is underlined. A star indicates either that the best baseline is significantly better than our best result or that our best result is significantly better than the best baseline according to classical significance tests (the p-value being fixed at 0.05).

We can make the following remarks. In the linear setting, our approaches are very competitive to the state of the art and RVML-Lin-OT tends to be the best on average even though it must be noticed that SCML is very competitive on some datasets (the average difference is not significant). RVML-Lin-Class performs slightly less on average. Considering now the non linear methods, our approaches improve their performance and are significantly better than the others on average, RVML-RBF-Class has the best average behavior in this setting. These experiments show that our regressive formulation

[2] With the σ parameter fixed as previously to the mean of all pairwise training set Euclidean distances.

Table 2: Comparison of our approach with several baselines in the non-linear case.

	Baselines				Our approach	
Base	1NN-KPCA	LMNN-KPCA	GBLMNN	SCMLLOCAL	RVML-RBF-OT	RVML-RBF-Class
Amazon	20.27 ± 2.42	53.16 ± 3.73	65.53 ± 2.32	69.14 ± 1.74	73.51 ± 0.83	**76.22** ± **2.09***
Breast	92.43 ± 2.19	95.39 ± 1.32	95.58 ± 0.87	**96.31** ± **0.66**	95.73 ± 0.97	95.78 ± 0.92
Caltech	20.82 ± 8.29	29.88 ± 10.89	49.91 ± 2.80	50.56 ± 1.62	54.39 ± 1.89	**57.98** ± **2.22***
DSLR	64.90 ± 5.81	73.92 ± 7.57	76.08 ± 4.79	62.55 ± 6.94	70.39 ± 4.48	**76.67** ± **4.57**
Ionosphere	75.57 ± 2.79	85.66 ± 2.55	87.36 ± 3.02	90.94 ± 3.02	90.66 ± 3.10	**93.11** ± **3.30***
Isolet	68.70	96.28	96.02	91.40	95.96	**96.73**
Letter	95.39 ± 0.27	**97.17*** ± 0.18	96.51 ± 0.25	96.63 ± 0.26	91.26 ± 0.50	96.09 ± 0.21
Pima	69.57 ± 2.64	69.48 ± 2.04	69.52 ± 2.27	68.40 ± 2.75	69.35 ± 2.95	**70.74** ± **2.36**
Scale	78.36 ± 0.88	88.10 ± 2.26	77.88 ± 2.43	93.86 ± 1.78	**95.19** ± **1.46***	94.07 ± 2.02
Splice	66.99	**88.97**	82.21	87.13	88.51	88.32
Svmguide1	**95.72**	95.60	95.00	87.40	95.67	95.05
Wine	92.18 ± 1.23	95.82 ± 2.98	98.00 ± 1.34	96.55 ± 2.00	**98.91** ± **1.53**	98.00 ± 1.81
Webcam	73.55 ± 4.57	84.52 ± 3.83	85.81 ± 3.75	88.71 ± 2.83	88.71 ± 4.28	**88.92** ± **2.91**
mean	70.34	81.07	82.72	83.04	85.25	**86.74**

is very competitive and is even able to improve state of the art performances in a non linear setting and consequently that our virtual point selection methods automatically select correct instances.

Considering the virtual point selection, we can observe that the OT formulation performs better than the class-based representation space one in the linear case, while it is the opposite in the non-linear case. We think that this can be explained by the fact that the OT approach generates more virtual points in a potentially non linear way which brings more expressiveness for the linear case. On the other hand, in the non linear one, the relative small number of virtual points used by the class-based method seems to induce a better regularization. In Section 4 of the supplementary material, we provide additional experiments showing the interest of using explicit virtual points and the need of a careful association between examples and virtual points. We also provide some graphics showing 2D projections of the space learned by RVML-Lin-Class and RVML-RBF-Class on the Isolet dataset illustrating the capability of these approaches to learn discriminative attributes.

In terms of computational cost, our approach -implemented in closed form [22]- is competitive with classical methods but does not yield to significant improvements. Indeed, in practice, classical approaches only consider a small number of constraints *e.g.* c times the number of examples, where c is a small constant, in the case of SCML. Thus, the practical computational complexity of both our approach and classical methods is linearly dependant on the number of examples.

5 Conclusion

We present a new metric learning approach based on a regression and aiming at bringing closer the learning examples to some a priori defined virtual points. The number of constraints has the advantage to grow linearly with the size of the learning set in opposition to the quadratic grow of standard must-link cannot-link approaches. Moreover, our method can be solved in closed form and can be easily kernelized allowing us to deal with non linear problems. Additionally, we propose two methods to define the virtual points: One making use of recent advances in the field of optimal transport and one based on unit vectors of a class-based representation space allowing one to perform directly some dimensionality reduction. Theoretically, we show that our approach is consistent and we are able to link our empirical risk to the true risk of a classical metric learning formulation. Finally, we empirically show that our approach is competitive with the state of the art in the linear case and outperforms some classical approaches in the non-linear one.

We think that this work opens the door to design new metric learning formulations, in particular the definition of the virtual points can bring a way to control some particular properties of the metric (rank, locality, discriminative power, . . .). As a consequence, this aspect opens new issues which are in part related to landmark selection problems but also to the ability to embed expressive semantic constraints to satisfy by means of the virtual points. Other perspectives include the development of a specific solver, of online versions, the use of low rank-inducing norms or the conception of new local metric learning methods. Another direction would be to study similarity learning extensions to perform linear classification such as in [21, 23].

References

[1] Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *Proc. of ICML*, pages 209–216, 2007.

[2] Jacob Goldberger, Sam T. Roweis, Geoffrey E. Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Proc. of NIPS*, pages 513–520, 2004.

[3] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2015.

[4] Brian Kulis. Metric learning: A survey. *Foundations and Trends in Machine Learning*, 5(4):287–364, 2013.

[5] Kilian Q. Weinberger, John Blitzer, and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Proc. of NIPS*, pages 1473–1480, 2005.

[6] Amir Globerson and Sam T. Roweis. Metric learning by collapsing classes. In *Proc. of NIPS*, pages 451–458, 2005.

[7] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009.

[8] Dor Kedem, Stephen Tyree, Kilian Q. Weinberger, Fei Sha, and Gert R. G. Lanckriet. Non-linear metric learning. In *Proc. of NIPS*, pages 2582–2590, 2012.

[9] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Proc. of ICANN*, pages 583–588, 1997.

[10] Jason Weston, Olivier Chapelle, André Elisseeff, Bernhard Schölkopf, and Vladimir Vapnik. Kernel dependency estimation. In *Proc. of NIPS*, pages 873–880, 2002.

[11] Corinna Cortes, Mehryar Mohri, and Jason Weston. A general regression technique for learning transductions. In *Proc. of ICML*, pages 153–160, 2005.

[12] Hachem Kadri, Mohammad Ghavamzadeh, and Philippe Preux. A generalized kernel approach to structured output learning. In *Proc. of ICML*, pages 471–479, 2013.

[13] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *Proc. of NIPS*, pages 862–870, 2009.

[14] Olivier Bousquet and André Elisseeff. Stability and generalization. *JMLR*, 2:499–526, 2002.

[15] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[16] Purushottam Kar and Prateek Jain. Similarity-based learning via data driven embeddings. In *Proc. of NIPS*, pages 1998–2006, 2011.

[17] Nicolas Courty, Rémi Flamary, and Devis Tuia. Domain adaptation with regularized optimal transport. In *Proc. of ECML/PKDD*, pages 274–289, 2014.

[18] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Proc. of NIPS*, pages 2292–2300, 2013.

[19] M. Lichman. UCI machine learning repository, 2013.

[20] Yuan Shi, Aurélien Bellet, and Fei Sha. Sparse compositional metric learning. In *Proc. of AAAI Conference on Artificial Intelligence*, pages 2078–2084, 2014.

[21] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Similarity learning for provably accurate sparse linear classification. In *Proc. of ICML*, 2012.

[22] The closed-form implementation of RVML is freely available on the authors' website.

[23] Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. Improved guarantees for learning via similarity functions. In *Proc. of COLT*, pages 287–298, 2008.

Regressive Virtual Metric Learning Supplementary Material

Michaël Perrot and Amaury Habrard
Université de Lyon, Université Jean Monnet de Saint-Etienne,
Laboratoire Hubert Curien, CNRS, UMR5516, F-42000, Saint-Etienne, France.
{michael.perrot,amaury.habrard}@univ-st-etienne.fr

The goal of this supplementary is to present the proofs of the main theorems of the paper along the first three sections. Moreover, in Section 4 we provide additional experiments showing the interest of using explicit virtual points and the need of a careful association between examples and virtual points. We also provide some graphics showing 2D projections of the space learned by RVML-Lin-Class and RVML-RBF-Class on the Isolet dataset illustrating the capability of these approaches to learn discriminative attributes.

First of all, before presenting the proofs, we recall our setting for the sake of completeness. Given a probability distribution $\mathcal{D}$ defined over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y}$ is a finite label set, let $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be a set of examples drawn i.i.d. from $\mathcal{D}$. Let $f_\mathbf{v} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{V}$ where $\mathcal{V} \subseteq \mathbb{R}^{d'}$ be the function which associates each example to a virtual point such that $\mathbf{v} = f_\mathbf{v}(\mathbf{x}, y)$. We denote by $\mathcal{D}_\mathbf{v}$ the probability distribution defined on $\mathcal{X} \times \mathcal{V}$ obtained from the distribution $\mathcal{D}$ after applying $f_\mathbf{v}$, *i.e.* $p_{\mathcal{D}_\mathbf{v}}(\mathbf{x}, \mathbf{v}) = p_\mathcal{D}(\mathbf{x}, y | \mathbf{v} = f_\mathbf{v}(\mathbf{x}, y))$. Thus it is equivalent to obtain the set of examples $S_\mathbf{v} = \{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^n$ from S after applying $f_\mathbf{v}$ and to draw $S_\mathbf{v}$ i.i.d. from $\mathcal{D}_\mathbf{v}$. Let $\|\cdot\|_\mathcal{F}$ be the Frobenius norm and $\|\cdot\|_2$ be the l_2 vector norm. We consider the following optimisation problem where we expanded the first Frobenius norm:

$$\mathbf{L} = \underset{\mathbf{L} \in \mathbb{R}^{d \times d'}}{\arg\min} f(\mathbf{L}) = \underset{\mathbf{L} \in \mathbb{R}^{d \times d'}}{\arg\min} \frac{1}{n} \sum_{(\mathbf{x}, \mathbf{v}) \in S_\mathbf{v}} \|\mathbf{x}^T \mathbf{L} - \mathbf{v}^T\|_2^2 + \lambda \|\mathbf{L}\|_\mathcal{F}^2. \tag{1}$$

Furthermore, we define the loss (2), the empirical risk (3) and the true risk of our algorithm (4):

$$l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) = \|\mathbf{x}^T \mathbf{L} - \mathbf{v}^T\|_2^2 \tag{2}$$

$$\hat{R}(\mathbf{L}) = \frac{1}{n} \sum_{(\mathbf{x}, \mathbf{v}) \in S_\mathbf{v}} l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) \tag{3}$$

$$R(\mathbf{L}) = \mathbb{E}_{(\mathbf{x}, \mathbf{v}) \sim \mathcal{D}_\mathbf{v}} l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) \tag{4}$$

1 Proof of Theorem 1

Theorem 1. *The optimal solution of Problem 1 can be found in closed form. Furthermore, we can derive two equivalent solutions:*

$$\mathbf{L} = \left(\mathbf{X}^T \mathbf{X} + \lambda n \mathbf{I}\right)^{-1} \mathbf{X}^T \mathbf{V} \tag{5}$$

$$\mathbf{L} = \mathbf{X}^T \left(\mathbf{X} \mathbf{X}^T + \lambda n \mathbf{I}\right)^{-1} \mathbf{V}. \tag{6}$$

Proof. Problem 1 is a classical regularized regression problem admitting a closed form solution [1]. We recall the derivation here for the sake of completeness. First we consider the derivative of

$f(\mathbf{L}, \mathbf{X}, \mathbf{V})$ with respect to $\mathbf{L}$:

$$\frac{\partial f(\mathbf{L}, \mathbf{X}, \mathbf{V})}{\partial \mathbf{L}} = 2\left(\frac{1}{n}\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)\mathbf{L} - \frac{2}{n}\mathbf{X}^T\mathbf{V}.$$

Then we set this derivative to zero to obtain:

$$\mathbf{L} = \left(\mathbf{X}^T\mathbf{X} + \lambda n\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{V}.$$

Finally Eq. 6 comes from using Taylor expansions as proposed in [1]. □

2 Proof of Theorem 2

The interest of this theorem is to show that our algorithm is consistent, i.e. that with a sufficient number of examples the empirical risk tends to be close to the true risk. To prove this theorem we use the uniform stability framework presented in [2]. The idea is to show that changing one example in the training set does not change much the output of the algorithm. Thus, we start by upper bounding the Frobenius norm of $\mathbf{L}$ optimal solution of Problem 1 and the loss (2) considered. Afterwards, we show the σ-admissibility of the loss which allows us to prove the uniform stability of our algorithm which, in turns, allows us to apply Theorem 12 from [2].

In the following, we assume that $\|\mathbf{x}\|_2 \leq C_\mathbf{x}$ and $\|\mathbf{v}\|_2 \leq C_\mathbf{v}$. The next lemma upper bounds the Frobenius norm of $\mathbf{L}$ optimal solution of Problem 1:

Lemma 1. *Let* $\mathbf{L}$ *be an optimal solution of Problem 1, we have:*

$$\|\mathbf{L}\|_\mathcal{F} \leq \frac{C_\mathbf{v}}{\sqrt{\lambda}}.$$

Proof. Since $\mathbf{L}$ is an optimal solution of Problem 1, we have:

$$\begin{aligned}
& & f(\mathbf{L}) \leq f(\mathbf{0}) & \\
\Leftrightarrow & & \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} l(\mathbf{L},(\mathbf{x},\mathbf{v})) + \lambda\|\mathbf{L}\|_\mathcal{F}^2 \leq \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} l(\mathbf{0},(\mathbf{x},\mathbf{v})) + \lambda\|\mathbf{0}\|_\mathcal{F}^2 & \\
\Rightarrow & & \lambda\|\mathbf{L}\|_\mathcal{F}^2 \leq \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} \|\mathbf{v}\|_2^2 & \qquad (7)\\
\Rightarrow & & \lambda\|\mathbf{L}\|_\mathcal{F}^2 \leq C_\mathbf{v}^2 & \\
\Rightarrow & & \|\mathbf{L}\|_\mathcal{F} \leq \frac{C_\mathbf{v}}{\sqrt{\lambda}} &
\end{aligned}$$

Inequality 7 is obtained by noting that our loss is always positive. □

We can now show that our loss is bounded.

Lemma 2. *The loss* $l(\mathbf{L},(\mathbf{x},\mathbf{v}))$ *is bounded by* $M = C_\mathbf{v}^2\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2$.

Proof.

$$\begin{aligned}
l(\mathbf{L},(\mathbf{x},\mathbf{v})) &= \|\mathbf{x}^T\mathbf{L} - \mathbf{v}^T\|_2^2 \\
&\leq \left(\|\mathbf{x}^T\|_2\|\mathbf{L}\|_\mathcal{F} + \|\mathbf{v}^T\|_2\right)^2 \qquad (8)\\
&\leq \left(C_\mathbf{x}\frac{C_\mathbf{v}}{\sqrt{\lambda}} + C_\mathbf{v}\right)^2 \\
&\leq C_\mathbf{v}^2\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2.
\end{aligned}$$

Inequality 8 comes from the successive application of the triangle inequality and standard properties on norms. □

We recall the definition of σ-admissibility from [2].

Definition 1. *A loss function l is σ-admissible if it is convex with respect to its first argument and the following condition holds:*

$$\forall \mathbf{L}, \mathbf{L}' \in \mathbb{R}^{d \times d'}, \forall (\mathbf{x}, \mathbf{v}) \sim \mathcal{D}_{\mathbf{v}}, |l(\mathbf{L}, (\mathbf{x}, \mathbf{v})) - l(\mathbf{L}', (\mathbf{x}, \mathbf{v}))| \leq \sigma \|\mathbf{L} - \mathbf{L}'\|_{\mathcal{F}}$$

We show that our loss is σ-admissible in the following lemma.

Lemma 3. *The loss $l(\mathbf{L}, (\mathbf{x}, \mathbf{v}))$ is σ-admissible with $\sigma = 2C_{\mathbf{v}}C_{\mathbf{x}}\left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)$.*

Proof.

$$\begin{aligned}
\big|\|\mathbf{x}^T\mathbf{L}' &- \mathbf{v}^T\|_2^2 - \|\mathbf{x}^T\mathbf{L}'' - \mathbf{v}^T\|_2^2\big| \\
&= \big|\|\mathbf{x}^T\mathbf{L}' - \mathbf{v}^T\|_2 - \|\mathbf{x}^T\mathbf{L}'' - \mathbf{v}^T\|_2\big| \, \big|\|\mathbf{x}^T\mathbf{L}' - \mathbf{v}^T\|_2 + \|\mathbf{x}^T\mathbf{L}'' - \mathbf{v}^T\|_2\big| \\
&\leq \|\mathbf{x}^T\mathbf{L}' - \mathbf{v}^T - \mathbf{x}^T\mathbf{L}'' + \mathbf{v}^T\|_2 \, \big|\|\mathbf{x}^T\mathbf{L}' - \mathbf{v}^T\|_2 + \|\mathbf{x}^T\mathbf{L}'' - \mathbf{v}^T\|_2\big| \qquad (9) \\
&\leq \|\mathbf{L}' - \mathbf{L}''\|_{\mathcal{F}} 2C_{\mathbf{v}}C_{\mathbf{x}}\left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right). \qquad (10)
\end{aligned}$$

Inequality 9 is due to the reverse triangle inequality and inequality 10 follows from Lemma 2. □

We will now prove that our algorithm is uniformly stable but before we need the following lemma. In the following $\hat{R}(\mathbf{L})$ is the empirical risk over a set $S_{\mathbf{v}}$ of examples while we design by $\hat{R}^i(\mathbf{L})$ the empirical risk over a set $S_{\mathbf{v}}^i$ obtained from $S_{\mathbf{v}}$ by replacing its i^{th} element. Similarly f and f^i denote the functions to optimize in Problem 1 using the sets of examples $S_{\mathbf{v}}$ and $S_{\mathbf{v}}^i$ respectively.

Lemma 4. *Let f and f^i be the functions to optimize, $\mathbf{L}$ and $\mathbf{L}^i$ their respective minimizers and λ the regularization parameter used in our algorithm. Let $\Delta\mathbf{L} = \mathbf{L} - \mathbf{L}^i$, then, we have, for any $t \in [0, 1]$:*

$$\|\mathbf{L}\|_{\mathcal{F}}^2 - \|\mathbf{L} - t\Delta\mathbf{L}\|_{\mathcal{F}}^2 + \|\mathbf{L}^i\|_{\mathcal{F}}^2 - \|\mathbf{L}^i + t\Delta\mathbf{L}\|_{\mathcal{F}}^2 \leq \frac{4tC_{\mathbf{v}}C_{\mathbf{x}}}{\lambda n}\left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)\|\Delta\mathbf{L}\|_{\mathcal{F}} \qquad (11)$$

Proof. This proof is similar to the proof in Lemma 20 in [2] which we recall here for the sake of completeness. First, note that $\hat{R}$ is a convex function, thus, for any $t \in [0, 1]$, we have:

$$\hat{R}^i(\mathbf{L} - t\Delta\mathbf{L}) - \hat{R}^i(\mathbf{L}) \leq t(\hat{R}^i(\mathbf{L}^i) - \hat{R}^i(\mathbf{L})) \qquad (12)$$

$$\hat{R}^i(\mathbf{L}^i + t\Delta\mathbf{L}) - \hat{R}^i(\mathbf{L}^i) \leq t(\hat{R}^i(\mathbf{L}) - \hat{R}^i(\mathbf{L}^i)) \qquad (13)$$

Summing inequalities (12) and (13) gives:

$$\hat{R}^i(\mathbf{L} - t\Delta\mathbf{L}) - \hat{R}^i(\mathbf{L}) + \hat{R}^i(\mathbf{L}^i + t\Delta\mathbf{L}) - \hat{R}^i(\mathbf{L}^i) \leq 0 \qquad (14)$$

$\mathbf{L}$ and $\mathbf{L}^i$ respectively minimize f and f^i, we have:

$$f(\mathbf{L}) - f(\mathbf{L} - t\Delta\mathbf{L}) \leq 0 \qquad (15)$$

$$f^i(\mathbf{L}^i) - f^i(\mathbf{L}^i + t\Delta\mathbf{L}) \leq 0 \qquad (16)$$

Summing inequalities (14), (15) and (16) gives:

$$\begin{aligned}
&\hat{R}^i(\mathbf{L} - t\Delta\mathbf{L}) - \hat{R}^i(\mathbf{L}) + \hat{R}(\mathbf{L}) - \hat{R}(\mathbf{L} - t\Delta\mathbf{L}) \\
&\qquad + \lambda\|\mathbf{L}\|_{\mathcal{F}}^2 - \lambda\|\mathbf{L} - t\Delta\mathbf{L}\|_{\mathcal{F}}^2 + \lambda\|\mathbf{L}^i\|_{\mathcal{F}}^2 - \lambda\|\mathbf{L}^i + t\Delta\mathbf{L}\|_{\mathcal{F}}^2 \leq 0. \qquad (17)
\end{aligned}$$

From (17), we can write:

$$\lambda\|\mathbf{L}\|_{\mathcal{F}}^2 - \lambda\|\mathbf{L} - t\Delta\mathbf{L}\|_{\mathcal{F}}^2 + \lambda\|\mathbf{L}^i\|_{\mathcal{F}}^2 - \lambda\|\mathbf{L}^i + t\Delta\mathbf{L}\|_{\mathcal{F}}^2 \leq B \qquad (18)$$

with

$$B = \hat{R}^i(\mathbf{L}) - \hat{R}^i(\mathbf{L} - t\Delta\mathbf{L}) + \hat{R}(\mathbf{L} - t\Delta\mathbf{L}) - \hat{R}(\mathbf{L}).$$

Using Lemma 3 we can bound B:

$$\begin{aligned}
B &\leq \left|\hat{R}^i(\mathbf{L}) - \hat{R}^i(\mathbf{L} - t\Delta\mathbf{L}) + \hat{R}(\mathbf{L} - t\Delta\mathbf{L}) - \hat{R}(\mathbf{L})\right| \\
&\leq \left|\frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x},\mathbf{v})) - \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})^i\in S_\mathbf{v}^i} l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x},\mathbf{v})^i)\right. \\
&\quad \left. + \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})^i\in S_\mathbf{v}^i} l(\mathbf{L}, (\mathbf{x},\mathbf{v})^i) - \frac{1}{n}\sum_{(\mathbf{x},\mathbf{v})\in S_\mathbf{v}} l(\mathbf{L}, (\mathbf{x},\mathbf{v}))\right| \qquad (19) \\
&\leq \frac{1}{n}\left|l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)) - l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)^i) + l(\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)^i) - l(\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i))\right| \qquad (20) \\
&\leq \frac{1}{n}\left|l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)) - l(\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i))\right| + \frac{1}{n}\left|l(\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)^i) - l(\mathbf{L} - t\Delta\mathbf{L}, (\mathbf{x}_i,\mathbf{v}_i)^i)\right| \\
&\leq \frac{4tC_\mathbf{v}C_\mathbf{x}}{n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)\|\Delta\mathbf{L}\|_\mathcal{F}. \qquad (21)
\end{aligned}$$

Inequality 19 comes from the definition of the empirical risk and Inequality 20 is deduced by noting that the sums only differ by their i^{th} element. Finally, we apply Lemma 3 twice to obtain Inequality 21. □

We recall the definition of uniform stability [2] in the next definition.

Definition 2. *An algorithm A has uniform stability β with respect to the loss function l if the following holds*

$$\forall S_\mathbf{v} \sim \mathcal{D}_\mathbf{v}^n, \forall i \in \{1,\dots,n\}, \sup_{(\mathbf{x},\mathbf{v})\sim\mathcal{D}_\mathbf{v}} \left|l(A_{S_\mathbf{v}}, (\mathbf{x},\mathbf{v})) - l(A_{S_\mathbf{v}^i}, (\mathbf{x},\mathbf{v}))\right| \leq \beta$$

where $S_\mathbf{v}^i$ is a training set obtained from $S_\mathbf{v}$ when replacing its i^{th} example with a new independent example and $A_{S_\mathbf{v}}$ and $A_{S_\mathbf{v}^i}$ stand for the optimal solution of algorithm A with respect to a given training set.

Lemma 5. *Our algorithm has a uniform stability in $\beta = \frac{8C_\mathbf{v}^2C_\mathbf{x}^2}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2$.*

Proof. By setting $t = \frac{1}{2}$ in Lemma 4, one can obtain for the left hand side:

$$\|\mathbf{L}\|_\mathcal{F}^2 - \|\mathbf{L} - \frac{1}{2}\Delta\mathbf{L}\|_\mathcal{F}^2 + \|\mathbf{L}^i\|_\mathcal{F}^2 - \|\mathbf{L}^i + \frac{1}{2}\Delta\mathbf{L}\|_\mathcal{F}^2 = \frac{1}{2}\|\Delta\mathbf{L}\|_\mathcal{F}^2$$

and thus:

$$\begin{aligned}
& \frac{1}{2}\|\Delta\mathbf{L}\|_\mathcal{F}^2 \leq \frac{2C_\mathbf{v}C_\mathbf{x}}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)\|\Delta\mathbf{L}\|_\mathcal{F} \\
\Rightarrow \quad & \|\Delta\mathbf{L}\|_\mathcal{F} \leq \frac{4C_\mathbf{v}C_\mathbf{x}}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)
\end{aligned}$$

From Lemma 3 we have:

$$\begin{aligned}
\left|l(\mathbf{L}, (\mathbf{x},\mathbf{v})) - l(\mathbf{L}^i, (\mathbf{x},\mathbf{v}))\right| &\leq 2C_\mathbf{v}C_\mathbf{x}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)\|\Delta\mathbf{L}\|_\mathcal{F} \\
&\leq \frac{8C_\mathbf{v}^2C_\mathbf{x}^2}{\lambda n}\left(1 + \frac{C_\mathbf{x}}{\sqrt{\lambda}}\right)^2
\end{aligned}$$

□

We recall Theorem 12 from [2] for the sake of completeness:

Theorem 12 ([2])**.** *Let A be an algorithm with uniform stability β w.r.t. a loss function l such that $0 \leq l(A_{S_{\mathbf{v}}}, (\mathbf{x}, \mathbf{v})) \leq M$ for all $(\mathbf{x}, \mathbf{v}) \sim \mathcal{D}_{\mathbf{v}}$ and all sets $S_{\mathbf{v}}$. Then for any $n \geq 1$ the following bound holds with probability at least $1 - \delta$ over the random draw of the sample $S_{\mathbf{v}}$,*

$$R(A_{S_{\mathbf{v}}}) \leq \hat{R}(A_{S_{\mathbf{v}}}) + \beta + (2n\beta + M)\sqrt{\frac{\ln \frac{1}{\delta}}{2n}}.$$

We have shown that our algorithm is uniformly stable and that our loss is bounded, hence we can apply this theorem to get Theorem 2.

Theorem 2. *Let $\|\mathbf{v}\|_2 \leq C_{\mathbf{v}}$ for any $\mathbf{v} \in \mathcal{V}$ and $\|\mathbf{x}\|_2 \leq C_{\mathbf{x}}$ for any $\mathbf{x} \in \mathcal{X}$. With probability $1 - \delta$, for any matrix $\mathbf{L}$ optimal solution of Problem 1, we have:*

$$R(\mathbf{L}) \leq \hat{R}(\mathbf{L}) + \frac{8C_{\mathbf{v}}^2 C_{\mathbf{x}}^2}{\lambda n}\left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)^2 + \left(\left(\frac{16C_{\mathbf{x}}^2}{\lambda} + 1\right) C_{\mathbf{v}}^2 \left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)^2\right)\sqrt{\frac{\ln \frac{1}{\delta}}{2n}}.$$

Proof. This theorem is a direct application of Theorem 12 from [2] using the bound on the loss presented in Lemma 2 and the uniform stability of our algorithm proven in Lemma 5. □

Kernelized case Recall that in the linear case we assumed that $\|\mathbf{x}\|_2 \leq C_{\mathbf{x}}$. In the kernelized case, we only have to bound $\|\phi(\mathbf{x})\|_2$ where ϕ is the projection function associated to the used kernel. A common assumption [3] when studying kernels is that $\exists\kappa$ such that $0 < \kappa < \infty$ and $K(\mathbf{x}, \mathbf{x}) \leq \kappa^2$. Hence, we have $\|\phi(\mathbf{x})\|_2^2 \leq \kappa^2$. Thus setting $C_{\mathbf{x}} = \kappa$ allows us to use the same proof than in the linear case leading us to the same generalization bound (the only difference being the value of $C_{\mathbf{x}}$).

3 Proof of Theorem 3

For the sake of readability we recall the loss for the classical metric learning approach [4] considered here:

$$l(\mathbf{L}, (\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j)) = \left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ \tag{22}$$

and the theorem:

Theorem 3. *Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \mathcal{Y}$. Let $\mathcal{V} \subset \mathbb{R}^{d'}$ be a finite set of virtual points and $f_{\mathbf{v}}$ is defined as $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \mathbf{v}_i$, $\mathbf{v}_i \in \mathcal{V}$. Let $\|\mathbf{v}\|_2 \leq C_{\mathbf{v}}$ for any $\mathbf{v} \in \mathcal{V}$ and $\|\mathbf{x}\|_2 \leq C_{\mathbf{x}}$ for any $\mathbf{x} \in \mathcal{X}$. Let $\gamma_1 = 2\max_{\mathbf{x}_k, \mathbf{x}_l, y_{kl}=1} d^2(\mathbf{v}_k, \mathbf{v}_l)$ and $\gamma_{-1} = \frac{1}{2}\min_{\mathbf{x}_k, \mathbf{x}_l, y_{kl}=-1} d^2(\mathbf{v}_k, \mathbf{v}_l)$, we have:*

$$\begin{aligned}&\mathbb{E}_{(\mathbf{x}_i, y_i)\sim\mathcal{D}, (\mathbf{x}_j, y_j)\sim\mathcal{D}}\left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ \\ &\quad\leq 8\left(\hat{R}(\mathbf{L}) + \frac{8C_{\mathbf{v}}^2 C_{\mathbf{x}}^2}{\lambda n}\left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)^2 + \left(\left(\frac{16C_{\mathbf{x}}^2}{\lambda} + 1\right) C_{\mathbf{v}}^2 \left(1 + \frac{C_{\mathbf{x}}}{\sqrt{\lambda}}\right)^2\right)\sqrt{\frac{\ln \frac{1}{\delta}}{2n}})\right).\end{aligned}$$

Proof. First of all, let us consider two examples $\mathbf{x}_i$ and $\mathbf{x}_j$ and their associated virtual points $\mathbf{v}_i$ and $\mathbf{v}_j$.

Using the fact that distances respect the triangle inequality, one can obtain:

$$d(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) \leq d(\mathbf{L}^T\mathbf{x}_i, \mathbf{v}_i) + d(\mathbf{v}_i, \mathbf{v}_j) + d(\mathbf{v}_j, \mathbf{L}^T\mathbf{x}_j).$$

Then squaring both sides of the inequality gives:

$$\begin{aligned}d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) \leq{}& d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{v}_i) + d^2(\mathbf{v}_i, \mathbf{v}_j) + d^2(\mathbf{v}_j, \mathbf{L}^T\mathbf{x}_j) \\ &+ 2(d(\mathbf{L}^T\mathbf{x}_i, \mathbf{v}_i) + d(\mathbf{v}_j, \mathbf{L}^T\mathbf{x}_j))d(\mathbf{v}_i, \mathbf{v}_j) + 2d(\mathbf{L}^T\mathbf{x}_i, \mathbf{v}_i)d(\mathbf{v}_j, \mathbf{L}^T\mathbf{x}_j).\end{aligned}$$

Finally, using Legendre identity[1] twice, we obtain:

$$d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j) \leq 4d^2(\mathbf{L}^T\mathbf{x}_i, \mathbf{v}_i) + 2d^2(\mathbf{v}_i, \mathbf{v}_j) + 4d^2(\mathbf{v}_j, \mathbf{L}^T\mathbf{x}_j).$$

[1] Legendre identity is $(a + b)^2 - (a - b)^2 = 4ab$ from which we deduce $a^2 + b^2 \geq 2ab$.

Similarly, switching the role of $d(\mathbf{L}^T\mathbf{x}_i, \mathbf{L}^T\mathbf{x}_j)$ and $d(\mathbf{v}_i, \mathbf{v}_j)$ we have:

$$\begin{aligned}
& d^2(\mathbf{v}_i,\mathbf{v}_j) \leq 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 2d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) \\
\Leftrightarrow \quad & -d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) \leq 2d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 2d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) - \frac{1}{2}d^2(\mathbf{v}_i,\mathbf{v}_j) \\
\Leftrightarrow \quad & -d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) \leq 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) - \frac{1}{2}d^2(\mathbf{v}_i,\mathbf{v}_j)
\end{aligned}$$

Now, let us consider two examples of the same class, *i.e.* $y_{ij} = 1$, we have:

$$\begin{aligned}
\left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ &= \left[d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) - \gamma_1\right]_+ \\
&\leq \left[4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) + 2d^2(\mathbf{v}_i,\mathbf{v}_j) - \gamma_1\right]_+ \\
&\leq 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j). \qquad (23)
\end{aligned}$$

Inequality 23 comes from the fact that $\gamma_1 \geq 2d^2(\mathbf{v}_i, \mathbf{v}_j)$ and by noting that a distance is always positive.

Similarly, we consider two examples of different classes, *i.e.* $y_{ij} = -1$, and we obtain:

$$\begin{aligned}
\left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ &= \left[-d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) + \gamma_{-1}\right]_+ \\
&\leq \left[4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) - \frac{1}{2}d^2(\mathbf{v}_i,\mathbf{v}_j) + \gamma_{-1}\right]_+ \\
&\leq 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j). \qquad (24)
\end{aligned}$$

Inequality 24 comes from the fact that $\gamma_{-1} \leq \frac{1}{2}d^2(\mathbf{v}_i, \mathbf{v}_j)$ and by noting that a distance is always positive.

Taking the expectation on both sides of Inequality 24 gives:

$$\begin{aligned}
& \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}} \left[y_{ij}(d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{L}^T\mathbf{x}_j) - \gamma_{y_{ij}})\right]_+ \qquad (25) \\
&\leq \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}} 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) \\
&\leq \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}} 4d^2(\mathbf{L}^T\mathbf{x}_i,\mathbf{v}_i) + \mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D},(\mathbf{x}_j,y_j)\sim\mathcal{D}} 4d^2(\mathbf{v}_j,\mathbf{L}^T\mathbf{x}_j) \\
&\leq 8\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} d^2(\mathbf{L}^T\mathbf{x},\mathbf{v}) \\
&\leq 8R(\mathbf{L}).
\end{aligned}$$

Applying Theorem 2 to the last inequality gives the theorem. □

4 Extended Experiments

In this section, we propose several experiments showing the interest of using explicit virtual points and the need of a careful association between examples and virtual points. We also provide some graphics showing 2D projections of the space learned by RVML-Lin-Class and RVML-RBF-Class on the isolet dataset illustrating the capability of these approaches to learn discriminative attributes.

4.1 Interest of Explicit Virtual Points

In [5] the authors propose to collapse similar examples on a single point, an implicit virtual point, while pushing far away dissimilar examples. This behavior can, in fact, be achieved by any margin based metric learning approach by setting the margin between similar examples to 0 and the margin between dissimilar examples to a high value. Thus to illustrate the interest of using explicit virtual points, we propose to compare our approach to ITML when considering the aforementioned margins (ITML-Collapse). For the sake of completeness we also consider ITML with tuned margins (ITML). The results are presented in Table 1 and show that, on average, ITML-Collapse is less accurate than RVML-Lin-Class hinting that considering explicit virtual points is better than considering implicit ones.

Table 1: Comparison between a method with explicit virtual points (RVML-Lin-Class) and a method with implicit virtual points (ITML-Collapse).

Base	RVML-Lin-Class	ITML-Collapse	ITML
Amazon	**73.09** ± **2.49**	57.97 ± 3.36	65.91 ± 2.64
Breast	95.34 ± 0.95	94.56 ± 1.41	**95.49** ± **1.15**
Caltech	**55.41** ± **2.55**	37.34 ± 2.01	47.31 ± 2.75
DSLR	75.29 ± 5.08	**77.25** ± **4.15**	77.25 ± 4.91
Ionosphere	82.74 ± 2.81	85.75 ± 6.23	**88.11** ± **1.68**
Isolet	**94.61**	74.53	92.88
Letters	95.51 ± 0.26	**95.67** ± **0.30**	95.00 ± 0.64
Pima	69.57 ± 2.85	**71.08** ± **2.13**	70.26 ± 1.38
Scale	**87.94** ± **1.99**	87.51 ± 4.39	87.67 ± 2.71
Splice	**78.44**	66.80	71.49
Svmguide1	85.25	94.62	**95.00**
Wine	**98.18** ± **1.48**	85.91 ± 3.74	96.91 ± 1.93
Webcam	88.60 ± 2.69	**97.64** ± **2.43**	86.56 ± 2.88
mean	**83.07**	78.97	82.30

Table 2: Comparison of our OT based formulation to a random selection approach when learning a linear metric.

	OT based approach	Random		
Base	RVML-Lin-OT	1 VP per class	2 VP per class	5 VP per class
Amazon	71.62 ± 1.34	**74.23** ± **2.15**	72.92 ± 2.31	70.31 ± 2.82
Breast	95.24 ± 1.21	**95.34** ± **0.95**	95.29 ± 1.32	94.90 ± 1.92
Caltech	52.51 ± 2.41	**55.09** ± **2.38**	53.63 ± 2.12	49.59 ± 1.69
DSLR	**74.71** ± **5.27**	70.59 ± 6.06	63.53 ± 5.08	52.16 ± 8.68
Ionosphere	87.36 ± 3.12	82.74 ± 2.81	88.40 ± 4.05	**90.28** ± **3.33**
Isolet	91.40	92.75	**94.16**	92.43
Letters	90.25 ± 0.60	89.90 ± 1.02	90.54 ± 1.24	**91.13** ± **0.74**
Pima	**70.48** ± **3.19**	69.57 ± 2.85	69.35 ± 2.44	69.26 ± 2.60
Scale	**90.05** ± **2.13**	88.10 ± 2.57	89.47 ± 2.99	89.21 ± 2.68
Splice	**84.64**	78.44	78.94	80.87
Svmguide1	**94.83**	85.25	86.90	94.70
Wine	**98.55** ± **1.67**	98.55 ± 1.43	97.64 ± 2.43	98.00 ± 1.34
Webcam	88.60 ± 3.63	**88.92** ± **3.21**	86.24 ± 2.95	81.18 ± 3.56
mean	**83.86**	82.27	82.08	81.08

4.2 Association of Examples and Virtual Points

To further assess the interest of using our OT based formulation to select virtual points and associate them to examples, we propose to compare it with a random based approach (Random). In this latter setting, we randomly select a subset of examples for each class to act as virtual points and we randomly associate each example of this class to these virtual points. The results in the linear case are presented in Table 2 while the results in the non-linear case are presented in Table 3. Overall, randomly selecting the virtual points is less accurate than using the OT based formulation. This is especially true in the linear case where the metric is less expressive than in the kernelized case and thus requires more meaningful virtual points. Hence, selecting virtual points and correctly associating them to the examples is key to obtain a good performance.

Table 3: Comparison of our OT based formulation to a random selection approach when learning a non linear metric.

	OT based approach	Random		
Base	RVML-RBF-OT	1 VP per class	2 VP per class	5 VP per class
Amazon	73.51 ± 0.83	**75.74 ± 2.35**	72.68 ± 2.02	70.07 ± 2.86
Breast	95.73 ± 0.97	95.73 ± 1.07	**95.83 ± 0.80**	95.58 ± 1.38
Caltech	54.39 ± 1.89	**58.33 ± 2.05**	53.98 ± 3.18	50.35 ± 1.89
DSLR	**70.39 ± 4.48**	65.29 ± 7.51	58.24 ± 7.79	48.82 ± 8.03
Ionosphere	**90.66 ± 3.10**	90.57 ± 3.05	89.25 ± 3.73	90.38 ± 3.26
Isolet	95.96	**96.99**	96.54	95.25
Letters	91.26 ± 0.50	91.77 ± 0.43	91.87 ± 0.52	**92.04 ± 0.62**
Pima	69.35 ± 2.95	70.82 ± 4.60	**71.26 ± 2.84**	70.00 ± 2.56
Scale	**95.19 ± 1.46**	93.39 ± 2.19	91.96 ± 1.69	91.32 ± 1.95
Splice	**88.51**	88.37	88.46	87.22
Svmguide1	**95.67**	95.03	95.55	95.88
Wine	**98.91 ± 1.53**	97.82 ± 1.88	97.27 ± 1.96	97.82 ± 1.67
Webcam	**88.71 ± 4.28**	87.31 ± 2.99	83.01 ± 3.28	76.67 ± 4.78
mean	**85.25**	85.17	83.53	81.65

4.3 Illustration of the Behavior of Our Approach on One Dataset

To illustrate the capability of RVML-Lin-Class and RVML-RBF-Class to learn discriminative attributes we propose to select two dimensions out of the 26 of the space learned by these approaches on the isolet dataset. We selected 3 pairs of axis and the images obtained are presented in Fig. 1. On the same line, we plot two images corresponding to the same axis pair: on the left column for RVML-Lin-Class and on the right column for RVML-RBF-Class. Note that for each axis, there is only one class for which the value of the attribute tends to be 1, for all the other classes this feature tends to be 0. Furthermore, we can note that the kernelized version of our metric outputs a more discriminative space: the examples are brought closer to their corresponding virtual point than in the linear version.

References

[1] Corinna Cortes, Mehryar Mohri, and Jason Weston. A general regression framework for learning string-to-string mappings. In *Predicting Structured Data*. MIT Press, 2007.

[2] Olivier Bousquet and André Elisseeff. Stability and generalization. *JMLR*, 2:499–526, 2002.

[3] Julien Audiffren and Hachem Kadri. Stability of multi-task kernel regression algorithms. In *Proc. of ACML*, pages 1–16, 2013.

[4] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning: Theory and algorithm. In *Proc. of NIPS*, pages 862–870, 2009.

[5] Amir Globerson and Sam T. Roweis. Metric learning by collapsing classes. In *Proc. of NIPS*, pages 451–458, 2005.

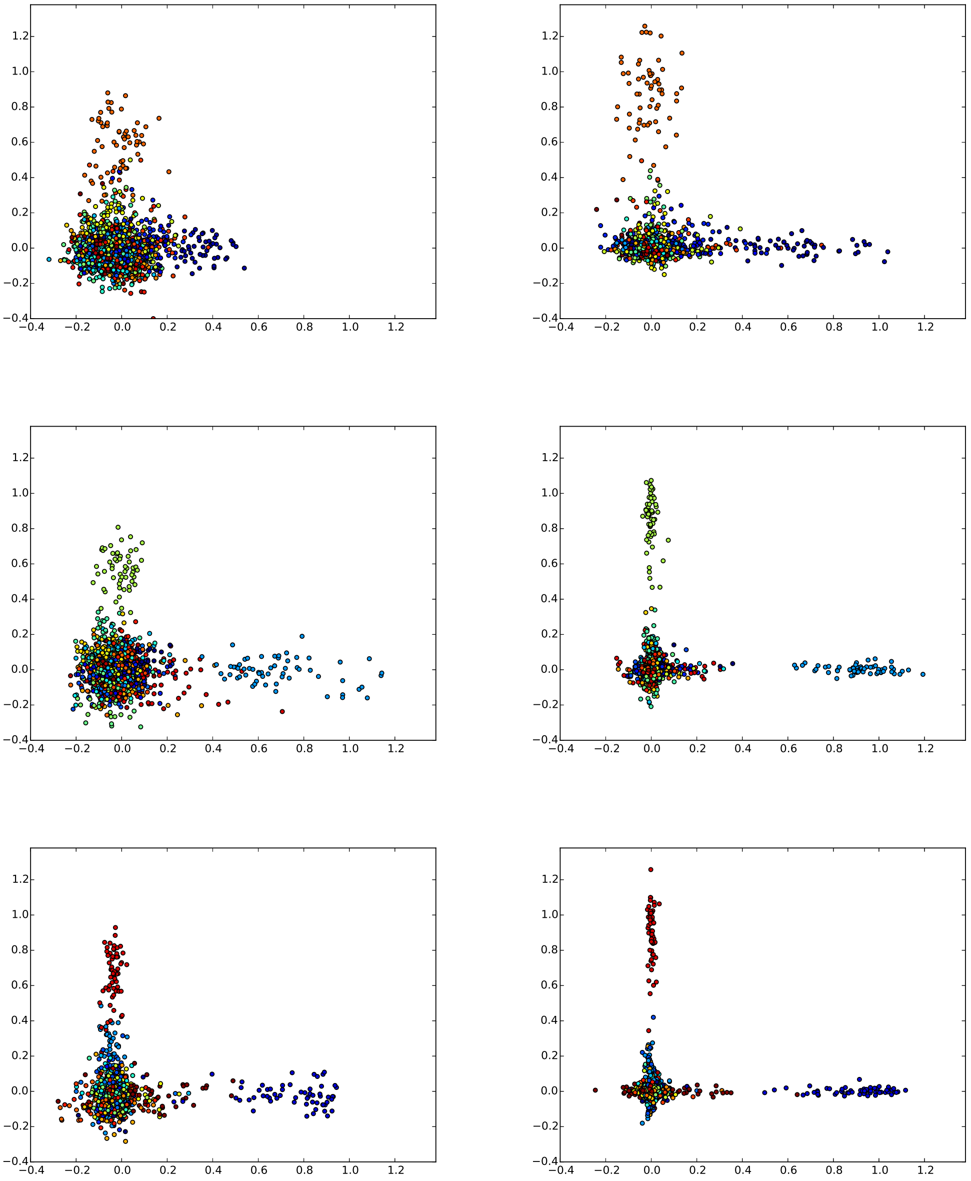


Figure 1: In the learned space from the isolet dataset, we randomly select 2 attributes three times and plot the 2D projection on each pair. The first line corresponds to features 1 and 20, the second line to features 7 and 14 and the third line to features 2 and 23. The left column corresponds to the space learned by RVML-Lin-Class (linear) and the right column to the one learned by RVML-RBF-Class (non linear). (Best viewed in color)