Apprentissage de Métriques par Régression

Michaël Perrot[1] et Amaury Habrard[1]

[1]Université Jean Monnet, Laboratoire Hubert Curien UMR CNRS 5516

Résumé

Nous nous intéressons à l'apprentissage supervisé de distances de type Mahalanobis. Les approches existantes cherchent principalement à apprendre un nouvel espace de représentation en fonction de contraintes prenant en compte des informations de similarités et de dissimilarités entre les exemples. Dans ce papier, au lieu de rapprocher où d'éloigner les exemples selon ce type de contraintes, nous proposons d'introduire le concept de points virtuels nous servant de support pour le déplacement des exemples d'apprentissage. Ainsi, les exemples d'apprentissage sont rapprochés d'un point virtuel qui leur a été affecté à priori. Nous montrons que l'approche proposée peut être résolue en forme close puis nous en déduisons une version utilisant l'astuce du noyau nous permettant de travailler dans un espace à grande dimension sans projection explicite. De plus, en utilisant de récentes avancées dans le domaine du transport optimal, nous proposons une solution efficace au difficile problème du choix des points virtuels. Enfin, nous montrons empiriquement l'intérêt de notre approche.

Mots-clef : Apprentissage de métriques, Noyaux, Régression.

1 Introduction

Le but des algorithmes d'apprentissage de métriques est de capturer, principalement grâce à un nouvel espace de représentation, les similarités et dissimilarités entre les exemples d'apprentissage. Dans les années précédentes, ce domaine de recherche s'est principalement intéressé à des distances de type Mahalanobis [BHS13] $d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')}$ où $\mathbf{M}$ est une matrice positive semi-définie (PSD) définissant un ensemble de paramètres à apprendre [1]. En utilisant une décomposition de Cholesky, on obtient $\mathbf{M} = \mathbf{L}\mathbf{L}^T$ et on en déduit un nouvel espace de représentation des données qui est linéairement dépendant de l'espace d'origine.

Les approches proposant d'apprendre une distance de Mahalanobis se basent principalement sur un jeu de contraintes, de type must-link et cannot-link, entre les exemples d'apprentissage [DKJ+07, GRHS04]. Ainsi, dans un contexte de classification supervisée, si on considère deux exemples $\mathbf{x}_i$ et $\mathbf{x}_j$, on cherchera à rapprocher ces points s'ils sont de la même classe et à les éloigner s'ils sont de classes différentes. L'objectif étant de faire en sorte que la métrique apprise associe un score faible aux paires de points de la même classe et un score élevé aux paires de points de classes différentes. Ce score pouvant alors être utilisé dans un algorithme de classification, par exemple, de type plus proche voisin. L'ensemble des contraintes disponibles est alors de complexité quadratique par rapport à l'échantillon d'apprentissage et est utilisé pour résoudre un problème d'optimisation généralement convexe. S'il est possible d'utiliser toute les paires pour apprendre, le nombre quadratique de contraintes peut être prohibitif dans certaines applications. Il est alors possible de n'en considérer qu'un sous-ensemble mais le choix de ces paires devient peu évident et nécessite de définir une règle de décision. Dans ce papier, nous proposons de considérer une approche différente basée sur le concept de points virtuels. Définis à priori, ces points nous permettent de reconsidérer le problème d'apprentissage de métriques comme une régression où le but est de minimiser l'écart entre les exemples d'apprentissage et les points virtuels. Ainsi, les contraintes considérées ne correspondent plus à des paires d'exemples mais à des paires : exemple d'apprentissage, point virtuel. Si ces paires peuvent, à priori, sembler difficiles à définir, nous présentons deux approches nous permettant de le faire de façon automatique. La première utilise de récentes approches dans le domaine du transport optimal pour faire le lien entre les exemples d'apprentissage et les points virtuels. La seconde se concentre sur la définition d'un espace de points virtuels dans lequel les exemples d'apprentissage sont projetés.

1. Si $\mathbf{M} = \mathbf{I}$ on retrouve la distance euclidienne.

La méthode proposée présente aussi un avantage important pour l'expressivité de l'approche. Rappelons que l'apprentissage de métriques permet de définir un espace de représentation dans lequel les relations de proximité entre les données sont plus pertinentes que dans l'espace d'origine. Cependant, l'expressivité d'une simple transformation linéaire, généralement considérée dans les approches classiques, n'est pas toujours suffisante pour obtenir un espace de représentation satisfaisant. Une approche largement utilisée afin d'augmenter l'expressivité des algorithmes d'apprentissage de métriques est celle de la projection, de façon à priori, des exemples d'apprentissage dans un espace non linéairement dépendant de l'espace d'origine. Cette projection se fait bien souvent à l'aide d'une analyse en composante principale kernelisée (KPCA) [SSM97]. L'inconvénient d'une telle approche est la perte de lien entre l'étape d'apprentissage de la métrique et l'inclusion de la non linéarité. Ainsi, si les composantes sélectionnées par la KPCA ne sont pas satisfaisantes, la métrique apprise ne sera pas forcément plus performante qu'une simple métrique linéaire. Dans ce papier, nous montrons que l'approche proposée peut facilement tirer partie de l'astuce du noyau alors que cet aspect est généralement difficile pour les approches classiques d'apprentissage de métriques. Cela nous permet d'apprendre une métrique dans un espace à grande dimension sans pour autant nous y projeter explicitement.

Ce papier s'organise de la façon suivante. Dans la section 2 nous identifions différents travaux de l'état de l'art liés à l'algorithme présenté ici. La section 3 est dédiée à la présentation de l'algorithme proposé. Dans la section 4 nous présentons une évaluation empirique du modèle sur différents jeux de données classiquement utilisés en apprentissage de métriques. Enfin nous concluons dans la section 5.

2 Contexte

Pour une vision globales des différentes avancées en apprentissage de métriques nous invitons le lecteur intéressé à se référer à [BHS13] et [Kul13]. Dans cette section nous présentons quelques algorithmes qui sont plus particulièrement liés à notre approche. Tout d'abord, une des méthodes les plus célèbres en apprentissage de métriques est LMNN [WBS05] où les auteurs proposent d'apprendre une matrice PSD dédiée à l'amélioration de l'algorithme des k-plus proches voisins. Ainsi plutôt que de considérer des paires d'exemples, ils se concentrent sur des contraintes basées sur des triplets d'exemples $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ où $\mathbf{x}_j$ et $\mathbf{x}_k$ appartiennent au voisinage de $\mathbf{x}_i$ et de sorte que $\mathbf{x}_i$ et $\mathbf{x}_j$ sont de la même classe et $\mathbf{x}_k$ d'une classe différente. L'idée est alors de rapprocher $\mathbf{x}_i$ et $\mathbf{x}_j$ tout en éloignant $\mathbf{x}_k$. Ainsi, si le nombre de contraintes parait cubique, ils considèrent en fait le rapprochement et l'éloignement de points qui étaient originellement proches et ne prennent donc pas en compte tous les triplets possibles. Prenant un point de vue différent, l'approche proposée dans [GR05] est basée sur la concentration en un point de l'ensemble des exemples d'une classe et sur l'éloignement de tous les autres exemples. Pour cela, les auteurs définissent une mesure estimant la probabilité d'obtenir un point $\mathbf{x}_j$ sachant un point $\mathbf{x}_i$ en fonction de la matrice apprise $\mathbf{M}$. Ensuite, ils minimisent, en fonction de $\mathbf{M}$ (PSD), la KL divergence entre cette mesure de probabilité et le cas idéal où la probabilité vaut 1 pour deux exemples de même classe et 0 pour deux exemples de classes différente. Une des deux approches que nous proposons se positionne comme étant l'intermédiaire de ces deux approches. En effet, nous définissons localement des points virtuels sur lesquels nous concentrons les points en fonction de leur classe et de leur distance au point virtuel.

Un problème récurrent lors de l'apprentissage de métriques de type Mahalanobis est la projection de la matrice apprise sur le cône des matrices PSD. En effet, cette projection nécessite une coûteuse décomposition en valeurs propres de la matrice. Pour palier à ce problème, dans ITML [DKJ+07] les auteurs proposent d'utiliser une divergence LogDet comme terme de régularisation. L'idée est d'apprendre une matrice proche d'une autre matrice PSD définie à priori. Les auteurs montrent que si la divergence est finie alors la matrice apprise est obligatoirement PSD. Une autre approche, développée par exemple dans [GRHS04] consiste à s'intéresser directement à l'apprentissage de la matrice $\mathbf{L}$ permettant d'obtenir $\mathbf{M} = \mathbf{L}\mathbf{L}^T$. L'inconvénient d'une telle approche est qu'elle rend souvent le problème d'optimisation non convexe et donc difficile à optimiser. Dans ce papier, nous nous intéressons aussi à l'apprentissage d'une telle matrice $\mathbf{L}$, cependant, grâce à l'utilisation de contraintes basées sur les points virtuels, nous montrons que notre formulation est non seulement convexe mais aussi qu'il existe une solution en forme close. Cela nous permet d'apprendre la métrique de façon efficace.

Le problème de l'inclusion de non linéarité dans les algorithmes d'apprentissage de métriques a été abordé dans plusieurs travaux. Une première approche consiste à apprendre directement une métrique non linéaire comme dans χ^2-LMNN [KTW+12] où les auteurs proposent d'apprendre une distance du χ^2 qui est

particulièrement adaptée pour la comparaison d'histogrammes. Notons que l'apprentissage de métriques non linéaires par nature se rapproche des méthodes d'apprentissage de noyaux qui est au delà du contexte de ce papier. Une autre approche pour l'apprentissage de métriques non-linéaires est l'apprentissage de métriques locales comme dans MM-LMNN [WS09] où l'idée est d'apprendre une métrique différente en fonction de la position dans l'espace des exemples d'apprentissage. De façon similaire, [KTW+12] proposent dans GB-LMNN de raffiner localement, par divisions successives de l'espace, la métrique apprise globalement par LMNN. Enfin, la troisième approche évoquée ici est celle de l'utilisation d'une projection des exemples d'apprentissage dans un espace non linéairement dépendant de l'espace d'origine. Cette projection peut se faire de manière à priori en utilisant une KPCA [SSM97] ou bien directement dans l'algorithme comme proposé dans [DKJ+07]. La première approche permet de rendre non linéaire la plupart des algorithmes d'apprentissage de métrique mais présente l'inconvénient de devoir choisir, à priori, les attributs conservés pour l'apprentissage. La seconde méthode présente l'inconvénient de nécessiter une réécriture, souvent non triviale, du problème d'origine [Kul13]. En effet, il est nécessaire de n'accéder aux exemples d'apprentissage qu'à l'aide d'un produit scalaire et travailler sur des distances impose de prendre en compte des paires d'exemples. Dans ce papier, nous montrons que l'utilisation des points virtuels, que l'on suppose choisis dans l'espace dans lequel on veut projeter les points d'apprentissage, permet d'appliquer facilement l'astuce du noyau et donc d'apprendre une métrique à partir d'un espace à très grande dimension sans pour autant s'y projeter explicitement.

La formulation que nous utilisons pour rapprocher les exemples d'apprentissage des points virtuels est basée sur une régression. Ceci nous permet d'établir un lien entre notre approche et plusieurs travaux utilisant la régression kernelisée pour la prédiction de sorties structurées [WCE+02, CMW05, KGP13]. Ainsi, ces approches minimisent l'écart entre une entrée et une sortie en utilisant les noyaux, i.e. en travaillant dans un espace non linéairement dépendant de l'espace d'origine. Dans notre cas, les exemples d'apprentissage peuvent être vus comme l'entrée et les points virtuels comme la sortie. Cependant, si nous projetons les points d'apprentissage dans un espace à haute dimension, ce n'est pas le cas des points virtuels pour lesquels nous considérons qu'ils appartiennent déjà à l'espace de représentation obtenu après l'apprentissage de la métrique. Ainsi, nous n'avons pas à résoudre le problème de la pré-image [KGP13]. De plus, nous ne cherchons pas à prédire le point virtuel associé à un nouvel exemple mais bien à apprendre une distance entre les exemples. Ainsi, après la phase d'apprentissage, les points virtuels ne sont plus considérés pour le calcul de la métrique. Enfin notons les travaux présentés dans [WT07] où les auteurs apprennent une métrique pour les noyaux utilisés dans la régression kernelisée. Dans notre approche nous ne cherchons pas à optimiser la performance des noyaux mais nous les considérons comme outil pour introduire de la non linéarité dans le modèle.

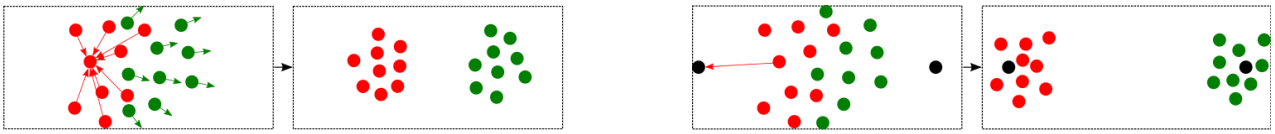
3 Contributions

L'apport principal de notre approche d'apprentissage de métriques est l'utilisation de points virtuels. Les algorithmes classiques d'apprentissage de métriques se basent sur des contraintes de type must-link et cannot-link entre les exemples d'apprentissage tandis que nous n'utilisons que des contraintes must-link entre un exemple d'apprentissage et un point virtuel. Les exemples associés au même point virtuel vont alors se rapprocher les uns des autres tout en s'éloignant des autres exemples. Notons que cela nous permet d'avoir un nombre de contraintes égal au nombre d'exemples d'apprentissage au contraire des approches classiques qui utilisent un nombre quadratique de contraintes. La figure 1 illustre les différences entre notre approche et une approche d'apprentissage de métrique classique.

Cette section s'articule autour de trois axes. Dans un premier temps nous supposons que nous avons accès à un ensemble de n paires d'apprentissage ($\mathbf{x}$,$\mathbf{v}$) où $\mathbf{x}$ est un example d'apprentissage et $\mathbf{v}$ est un point virtuel et nous présentons l'algorithme proposé ainsi que sa version kernélisée. Il s'agit en fait d'une approche par régression classique où l'originalité tient à l'utilisation de points virtuels pour apprendre une métrique. Dans un deuxième temps nous montrons un lien théorique entre notre approche et une approche d'apprentissage de métriques plus classique. Enfin, dans un troisième temps, nous proposons deux approches pour générer les paires d'apprentissage.

3.1 Apprentissage de métriques par régression

Soit une distribution de probabilité $\mathcal{D}$ définie sur $\mathcal{X} \times \mathcal{Y}$ où $\mathcal{X} \subseteq \mathbb{R}^d$ et $\mathcal{Y}$ est l'ensemble fini des étiquettes, on considère un ensemble d'exemples $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ tiré i.i.d. selon $\mathcal{D}$. Soit une fonction $f_{\mathbf{v}} : \mathcal{X} \times \mathcal{Y} \to \mathcal{V}$, où $\mathcal{V} \subseteq \mathbb{R}^{d'}$, permettant d'associer chaque exemple à un point virtuel, on définit alors l'ensemble d'ap-



(a) Apprentissage classique (b) Notre approche

FIGURE 1 – Les flêches indiquent les ccontraintes utilisées par les deux méthodes d'apprentissage pour un point dans un contexte de classification binaire. L'approche classique, figure 1(a), essaye de rapprocher les points de même classe tout en éloignant les points de classe différente en utilisant $\mathcal{O}(n^2)$ contraintes. Au contrainre, notre approche, figure 1(b), rapproche les exemples de points virtuels (en noir) et n'utilise que n contraintes. (pour une lecture optimale, voir en couleurs)

prentissage $S_{\mathbf{v}} = \{(\mathbf{x}_i, \mathbf{v}_i)\}_{i=1}^n$ où $\mathbf{v}_i = f_{\mathbf{v}}(\mathbf{x}_i, y_i)$. Par souci de simplification des écritures on désigne par $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$ et $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_n)^T$ les matrices contenant respectivement sur chaque lignes les exemples d'apprentissage et les points virtuels qui leur sont associés. Dans cette section, nous considérons que la fonction $f_{\mathbf{v}}$ est connue, nous reviendrons sur sa définition dans la section 3.3. Notons $\|\cdot\|_{\mathcal{F}}$ la norme de Frobenius, le but est d'apprendre les paramètres d'une matrice $\mathbf{M}$ d'une distance de type Mahalanobis. Pour cela nous proposons de nous concentrer sur l'apprentissage d'une matrice $\mathbf{L}$ telle que $\mathbf{M} = \mathbf{L}\mathbf{L}^T$. Le problème d'optimisation considéré est le suivant :

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

L'idée est d'apprendre un nouvel espace de représentation où les points d'apprentissage sont proches des points virtuels qui leur sont associés. Notons que $\mathbf{L}$ est une matrice de dimensions $d \times d'$. Ainsi, si $d' < d$, notre algorithme permet, en plus, de réduire la dimension des exemples.

Théorème 1. *La solution optimale du problème 1 peut être trouvée en forme close et s'écrit de deux manières :*

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

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

Démonstration. Le problème 1 est un problème de régression régularisé classique admettant une solution en forme close [CMW07] dont nous rappelons rapidement la dérivation ici par souci de complétude. On considère la dérivée de $f(\mathbf{L}, \mathbf{X}, \mathbf{V})$ par rapport à $\mathbf{L}$:

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

Ensuite on annule cette dérivée et on obtient :

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

Enfin, la solution 3 vient de l'utilisation des séries de Taylor comme proposé dans [CMW07]. □

De l'équation 2, on déduit une première forme de la matrice $\mathbf{M}$:

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

Notons que la matrice $\mathbf{M}$ est PSD par construction :

$$\begin{aligned} & \mathbf{x}^T\mathbf{M}\mathbf{x} = \mathbf{x}^T\mathbf{L}\mathbf{L}^T\mathbf{x} \\ \Leftrightarrow \quad & \mathbf{x}^T\mathbf{M}\mathbf{x} = \left(\mathbf{L}^T\mathbf{x}\right)^T \left(\mathbf{L}^T\mathbf{x}\right) \\ \Leftrightarrow \quad & \mathbf{x}^T\mathbf{M}\mathbf{x} = \|\mathbf{L}^T\mathbf{x}\|_2^2 \\ \Rightarrow \quad & \mathbf{x}^T\mathbf{M}\mathbf{x} \geq 0. \end{aligned}$$

Jusque là nous avons présenté notre algorithme d'apprentissage de métrique linéaire. Nous allons maintenant montrer qu'il est possible d'utiliser l'astuce du noyau dans notre algorithme. Ainsi, nous pouvons travailler dans un espace à grande dimension sans pour autant nous y projeter explicitement.

Soit $\phi(\mathbf{x})$ une fonction de projection. On définit le noyau $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}')$. Par soucis de simplicité on définit $K_{\mathbf{X}} = \phi(\mathbf{X})\phi(\mathbf{X})^T$ où $\phi(\mathbf{X})$ correspond à la matrice $(\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n))^T$. En considérant la matrice $\mathbf{L}$ solution présentée dans l'équation 3, on obtient la matrice $\mathbf{M} = \mathbf{X}^T \left(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}\right)^{-1} \mathbf{V}\mathbf{V}^T \left(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}\right)^{-1} \mathbf{X}$. La matrice $\mathbf{M}_K$ version noyau de $\mathbf{M}$ se déduit alors directement :

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

Notons que la distance de Mahalanobis au carré peut s'écrire comme $d_{\mathbf{M}}^2(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{M}\mathbf{x} + \mathbf{x}'^T\mathbf{M}\mathbf{x}' - 2\mathbf{x}^T\mathbf{M}\mathbf{x}'$.

On en déduit la version noyau $d^2_{\mathbf{M}_K}(\phi(\mathbf{x}), \phi(\mathbf{x}')) = \phi(\mathbf{x})^T \mathbf{M}_K \phi(\mathbf{x}) + \phi(\mathbf{x}')^T \mathbf{M}_K \phi(\mathbf{x}') - 2\phi(\mathbf{x})^T \mathbf{M}_K \phi(\mathbf{x}')$, avec :

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

où $K_{\mathbf{X}}(\mathbf{x})$ est le vecteur des similarités, calculées avec le noyau, entre l'exemple $\mathbf{x}$ et les exemples d'apprentissage.

D'autre part, notons qu'il est aussi possible d'obtenir une version noyau de $\mathbf{L}$:

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

Ce résultat est proche d'un résultat déjà dérivé dans [CMW05] dans un contexte de prédiction de données structurées. La différence principale vient du fait que nous n'utilisons pas de noyau sur la sortie, i.e. les points virtuels. Il nous est ainsi possible de calculer explicitement la projection d'un exemple $\mathbf{x}$ de dimension d dans un nouvel espace métrique de dimension d' :

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

Rappelons que, dans ce travail, nous sommes intéressés par une distance entre des exemples - potentiellement différents des points d'apprentissage - et pas par la prédiction de points virtuels. Ces derniers servent uniquement de base pour rapprocher les exemples similaires et éloigner les exemples dissimilaires.

Dans cette section nous avons présenté notre approche d'apprentissage de métriques, d'abord sous sa forme linéaire puis sous sa forme non linéaire. Dans la suite nous proposons de faire un lien entre notre méthode et une approche plus classique d'apprentissage de métriques.

3.2 Lien avec une approche d'apprentissage de métriques classique

Une méthode classique d'apprentissage de métriques pourrait consister à minimiser la distance entre les exemples similaires et à maximiser la distance entre les exemples dissimilaires. Considérons que deux exemples sont similaires lorsqu'ils sont de la même classe et qu'ils sont dissimilaires lorsqu'ils sont de classes différentes. Soit y_{ij} tel que $y_{ij} = 1$ si les exemples $\mathbf{x}_i$ et $\mathbf{x}_j$ sont de la même classe et -1 sinon. On dénote par $d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j)$ la distance de Mahalanobis au carré [2]. Une approche classique d'apprentissage de métriques pourrait ainsi chercher la matrice $\mathbf{L}$ solution du problème suivant :

$$\min_{\mathbf{L}} f'(\mathbf{L}, \mathbf{X}) = \min_{\mathbf{L}} \frac{1}{2(n-1)} \sum_{\mathbf{x}_i, \mathbf{x}_j} y_{ij} d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + \lambda \|\mathbf{L}\|^2_{\mathcal{F}}. \tag{6}$$

Notons que ce problème est non convexe et donc difficile à résoudre [3]. Le théorème suivant montre qu'il est possible de borner l'équation 6 par notre approche dans le cas où tous les exemples d'apprentissage de la même classe sont rapprochés du même point virtuel.

Théorème 2. *Soit $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n \in (\mathcal{X} \times \mathcal{Y})^n$ un ensemble d'apprentissage tiré i.i.d. selon $\mathcal{D}$. Soit $\mathcal{V} \subset \mathbb{R}^{d'}$ un ensemble de points virtuels de sorte que $|\mathcal{V}| = |\mathcal{Y}|$ et $f_{\mathbf{v}}$ définie comme suit $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \mathbf{v}_{y_i}$, $\mathbf{v}_{y_i} \in \mathcal{V}$. On a alors :*

$$\begin{aligned}
&\min_{\mathbf{L}} \frac{1}{2(n-1)} \sum_{\mathbf{x}_i, \mathbf{x}_j} y_{ij} d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + \lambda \|\mathbf{L}\|^2_{\mathcal{F}} \\
&\qquad \leq \min_{\mathbf{L}} \|\mathbf{X}\mathbf{L} - \mathbf{V}\|^2_{\mathcal{F}} + \lambda \|\mathbf{L}\|^2_{\mathcal{F}}.
\end{aligned}$$

Démonstration. D'une part, considérons $\mathbf{x}_i$ et $\mathbf{x}_j$ deux exemples d'apprentissage de la même classe, soit $\mathbf{v}_i = \mathbf{v}_j$ le point virtuel auquel ils sont associés. L'inégalité triangulaire sur les distances donne :

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

En prenant le carré de l'inéquation précédente puis en appliquant une conséquence de l'identité de Legendre [4] et en notant que $-\frac{1}{2} d^2(\mathbf{v}_i, \mathbf{v}_j) = 0$, on obtient :

$$\begin{aligned}
& d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) \leq d^2(\mathbf{L}\mathbf{x}_i, \mathbf{v}_i) + d^2(\mathbf{v}_j, \mathbf{L}\mathbf{x}_j) \\
&\qquad\qquad + 2d(\mathbf{L}\mathbf{x}_i, \mathbf{v}_i) d(\mathbf{v}_j, \mathbf{L}\mathbf{x}_j) \\
\Leftrightarrow \quad & d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) \leq 2d^2(\mathbf{L}\mathbf{x}_i, \mathbf{v}_i) + 2d^2(\mathbf{v}_j, \mathbf{L}\mathbf{x}_j) \\
&\qquad\qquad - \frac{1}{2} d^2(\mathbf{v}_i, \mathbf{v}_j).
\end{aligned} \tag{7}$$

D'autre part, considérons $\mathbf{x}_i$ et $\mathbf{x}_j$ deux exemples d'apprentissage de classes différentes et $\mathbf{v}_i$ et $\mathbf{v}_j$ les points virtuels qui leurs sont respectivement associés.

2. On suppose ici que $\mathbf{M} = \mathbf{L}\mathbf{L}^T$, ce qui est vrai si $\mathbf{M}$ est PSD en utilisant la décomposition de Cholesky.

3. De nombreux algorithmes d'apprentissage de métriques peuvent être vus comme une relaxation de ce problème par l'utilisation de fonctions de pertes convexes [JWZ09, DKJ+07].

4. L'identité de Legendre est $(a+b)^2 - (a-b)^2 = 4ab$ de laquelle on déduit $a^2 + b^2 \geq 2ab$.

Toujours en utilisant l'inégalité triangulaire sur les distances, on peut écrire :

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

En élevant au carré l'inégalité précédente puis en utilisant plusieurs fois l'inégalité de Legendre, on obtient :

$$\begin{aligned} & d^2(\mathbf{v}_i, \mathbf{v}_j) \leq d^2(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i) + d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + d^2(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) \\ & \quad + 2d(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i)d(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + 2d(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i)d(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) \\ & \quad + 2d(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j)d(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) \\ \Leftrightarrow \quad & d^2(\mathbf{v}_i, \mathbf{v}_j) \leq 4d^2(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i) + 2d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + 4d^2(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) \\ \Leftrightarrow & -d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) \leq 2d^2(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i) + 2d^2(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) - \frac{1}{2}d^2(\mathbf{v}_i, \mathbf{v}_j). \end{aligned} \tag{8}$$

En utilisant les inégalités 7 et 8 et en notant que y_{ij} vaut 1 pour des exemples de même classe et -1 pour des exemples de classes différentes, on obtient :

$$\begin{aligned} & \min_{\mathbf{L}} \frac{1}{2(n-1)} \sum_{\mathbf{x}_i, \mathbf{x}_j} y_{ij} d^2(\mathbf{L}\mathbf{x}_i, \mathbf{L}\mathbf{x}_j) + \lambda \|\mathbf{L}\|_{\mathcal{F}}^2 \\ & \quad \leq \min_{\mathbf{L}} \frac{1}{2(n-1)} \sum_{\mathbf{x}_i, \mathbf{x}_j} (2d^2(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i) + 2d^2(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j) \\ & \qquad - \frac{1}{2}d^2(\mathbf{v}_i, \mathbf{v}_j)) + \lambda \|\mathbf{L}\|_{\mathcal{F}}^2 \\ & \quad \leq \min_{\mathbf{L}} \frac{1}{(n-1)} \sum_{\mathbf{x}_i, \mathbf{x}_j} \left(d^2(\mathbf{v}_i, \mathbf{L}\mathbf{x}_i) + d^2(\mathbf{L}\mathbf{x}_j, \mathbf{v}_j)\right) + \lambda \|\mathbf{L}\|_{\mathcal{F}}^2 \\ & \qquad - \sum_{\mathbf{x}_i, \mathbf{x}_j} \frac{1}{4(n-1)} d^2(\mathbf{v}_i, \mathbf{v}_j) \\ & \quad \leq \min_{\mathbf{L}} \|\mathbf{X}\mathbf{L} - \mathbf{V}\|_{\mathcal{F}}^2 + \lambda \|\mathbf{L}\|_{\mathcal{F}}^2 - \sum_{\mathbf{x}_i, \mathbf{x}_j} \frac{1}{4(n-1)} d^2(\mathbf{v}_i, \mathbf{v}_j). \end{aligned}$$

La dernière inéquation est obtenue en notant que, pour un exemple $\mathbf{x}$, le terme $d^2(\mathbf{v}, \mathbf{L}\mathbf{x})$ apparait $(n-1)$ fois dans la somme. Enfin, remarquer que $\sum_{\mathbf{x}_i, \mathbf{x}_j} \frac{1}{4} d^2(\mathbf{v}_i, \mathbf{v}_j)$ est une constante strictement positive donne le théorème. □

Dans cette section, nous avons montré qu'il était possible de faire un lien entre un algorithme d'apprentissage de métriques plus classique et notre approche. Dans la section suivante nous proposons deux méthodes pour sélectionner les paires d'apprentissages que nous avions supposées données jusque là.

3.3 Points virtuels

Nous proposons ici de nous intéresser au problème de la génération des paires d'exemples ($\mathbf{x}$,$\mathbf{v}$). Dans un premier temps nous proposons de nous intéresser à l'utilisation des exemples d'apprentissage en temps que base de génération des points virtuels tandis que dans un second temps nous proposons d'utiliser des points virtuels qui correspondent aux vecteurs unitaires d'un nouvel espace de représentation dans lequel seront projetés les exemples d'apprentissage.

3.3.1 Utilisation des exemples d'apprentissage et transport optimal

Une première méthode pour sélectionner les points virtuels est de les construire à l'aide d'une variante du transport optimal [Vil08] et d'un sous ensemble $\mathbf{V}'$ des exemples d'apprentissage [BBS08, BHS12]. Ainsi, contrairement au travail proposé dans [BHS12], nous n'allons pas chercher à rapprocher directement les exemples d'apprentissages des exemples de $\mathbf{V}'$ de la même classe de façon globale et moyenne. À la place nous proposons d'utiliser de récentes avancées dans le domaine du transport optimal [CFT14] pour associer chaque exemple d'apprentissage à un nouveau point virtuel.

Le problème du transport optimal [Vil08] est, étant donnés un ensemble de k exemples d'entrée et un ensemble de k' exemples de sortie, de calculer la meilleure association possible entre l'entrée et la sortie sachant que l'on a accès à une matrice $\mathbf{C}$ de taille $k \times k'$ indiquant le coût pour transporter un point d'entrée vers un point de sortie. Le problème du transport est alors d'apprendre une matrice $\gamma \in \mathbb{R}^{k \times k'}$ qui minimise le coût de déplacer les entrées vers les sorties. Pour cela, dans [CFT14], les auteurs proposent de résoudre le problème suivant :

$$\gamma_0 = \arg\min_{\gamma} \langle \gamma, \mathbf{C} \rangle_{\mathcal{F}} - \frac{1}{\lambda} h(\gamma) + \eta \sum_j \sum_c \|\gamma(y_i = c, j)\|_q^p$$

où $h(\gamma) = -\sum_{i,j} \gamma(i,j) \log(\gamma(i,j))$ correspond à l'entropie de γ et permet un calcul plus efficace du transport [Cut13]. Le second terme de régularisation, où $\gamma(y_i = c, j)$ désigne les lignes de la colonne j de γ où la classe de l'entrée est c, a été introduit dans [CFT14]. Le but est d'empêcher les exemples d'entrée de classes différentes de se rapprocher des mêmes exemples de sortie en promouvant une parcimonie de groupe dans la matrice γ. Nous proposons ici de réutiliser cette approche pour associer les exemples d'apprentissage (exemples d'entrée) aux éléments de $\mathbf{V}'$ (exemples de sortie), la matrice de coût étant fixée à la distance euclidienne, i.e. $\mathbf{C}(i,j) = \|\mathbf{x}_i - \mathbf{v}'_j\|_2$. Ainsi, nous obtenons la fonction $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \gamma(i)\mathbf{V}'$, $\gamma(i)$ désignant la ligne i de γ, qui nous permet de générer nos paires d'apprentissage. Notons que la matrice des points virtuels $\mathbf{V}$ peut être obtenue comme $\mathbf{V} = \gamma\mathbf{V}'$, chaque point virtuel correspondant alors à une combinaison linéaire des points de $\mathbf{V}'$. Le transport permet ainsi d'associer chaque exemple d'apprentissage $\mathbf{x}$ à un point virtuel

$\mathbf{v}$ éventuellement de façon non linéaire, i.e. il n'existe pas forcément une matrice $\mathbf{T}$ telle que $\mathbf{V} = \mathbf{XT}$. Notre méthode d'apprentissage de métriques peut, dans ce cas, être vue comme une approximation linéaire ou non linéaire du transport optimal en considérant respectivement les solutions 2 et 5 présentées précédemment. Remarquons qu'ici les points virtuels ne sont pas définis à priori mais qu'ils sont automatiquement construits par la résolution du problème de transport optimal.

Dans cette première approche, nous avons pris le parti de créer des paires d'apprentissage qui correspondent à un exemple d'apprentissage et un point virtuel lui même combinaison linéaire d'un sous-ensemble d'exemples. Notons qu'une approche simple, utilisée ici, pour sélectionner ce sous-ensemble, consiste à effectuer un tirage aléatoire parmi les exemples d'apprentissage. Cependant, il pourrait être intéressant d'étudier le comportement de notre algorithme avec d'autres approches de sélection de sous-ensemble d'exemples [KJ11, KJ12]. Dans la section suivante, nous proposons de prendre une autre perspective sur la sélection des points virtuels et de les voir comme les vecteurs unitaires d'un nouvel espace de représentation.

3.3.2 Points virtuels et nouvel espace de représentation

Pour cette seconde méthode de sélection des points virtuels, nous proposons de les considérer comme les vecteurs unitaires d'un nouvel espace de représentation. Nous verrons que cela permet, d'une part, de réduire dans de nombreux cas la dimension de l'espace d'arrivée et, d'autre part, de chercher un espace de représentation où chaque attribut est discriminant pour une classe donnée.

On considère un problème d'apprentissage à k classes. L'idée est d'apprendre une métrique qui projette les exemples d'apprentissage dans un nouvel espace à k dimensions. Pour cela, nous proposons de fixer chaque point virtuel comme un vecteur unitaire de ce nouvel espace. Soit $\mathbf{e}_j \in \mathbb{R}^k$ le vecteur unitaire dont seul l'attribut j est à 1 tous les autres attributs étant à 0, pour tout exemple $(\mathbf{x}_i, y_i)$, on définit $f_{\mathbf{v}}(\mathbf{x}_i, y_i) = \mathbf{e}_{\#y_i}$ où $\#y_i = j$ si $\mathbf{x}_i$ est de la $j^{\text{ème}}$ classe. Ainsi, les paires d'apprentissage correspondent à l'association d'un exemple d'apprentissage et du vecteur unitaire correspondant à sa classe. Si le nombre de classes est petit les exemples d'apprentissage seront projetés dans un nouvel espace à faible dimension. De plus, tous les exemples d'une classe sont projetés sur un axe et un seul, cela implique qu'après apprentissage l'attribut j sera discriminant pour la $j^{\text{ème}}$ classe.

Dans cette section nous avons présenté notre algorithme ainsi qu'une version non linéaire de celui-ci. Nous avons aussi montré qu'il est possible, dans certains cas, de lier notre approche à une approche plus classique d'apprentissage de métriques. Enfin nous avons proposé deux méthodes de génération des paires d'apprentissage. Dans la suite de ce document, nous montrons que notre approche est empiriquement fondée en l'utilisant sur plusieurs jeux de données classiques en apprentissage de métriques.

4 Résultats expérimentaux

Nous proposons de montrer la performance empirique de notre algorithme sur 7 jeux de données classiques en apprentissage de métriques. Le tableau 1 reprend les différentes caractéristiques de ces bases. Nous normalisons les données en soustrayant la moyenne et en divisant par 3 fois l'écart type pour chaque attribut. Enfin, sauf pour Splice et Svmguide-1 où nous avons accès à un ensemble de test et un ensemble d'apprentissage, les résultats présentés sont la moyenne de 10 exécutions où 70% des données sont utilisées pour apprendre et 30% sont utilisées pour tester. Nous comparons notre méthode avec différents algorithmes de l'état de l'art. Le classifieur utilisé est un 1-plus proche voisin. Nous proposons deux séries d'expérimentations. Une première visant à apprendre des transformations linéaires et une seconde s'intéressant à des métriques non-linéaires.

Dans la première série d'expérimentations nous considérons les méthodes linéaires suivantes :

- Id : la métrique est une simple distance euclidienne.
- LMNN : la métrique est apprise en utilisant LMNN [WBS05].
- ITML : la métrique est apprise en utilisant ITML [DKJ$^+$07].
- RML-Linear-OT : notre approche linéaire en utilisant la première méthode de sélection des points virtuels basée sur le transport optimal. La taille du sous-ensemble d'exemples utilisé est fixée par cross-validation entre 5, 10, 15, 20 et 25 % des données d'apprentissage.
- RML-Linear-Vect : notre approche linéaire en utilisant la seconde méthode de sélection des points virtuels basée sur les classes des exemples. Notons que dans ce cas la dimension des exemples après apprentissage est réduite.

Pour la seconde série d'expérimentations, nous comparons les méthodes suivantes :

- Id-KPCA : la métrique est la distance euclidienne

	Breast	Pima	Scale	Wine	Ionosphere	Splice	Svmguide-1
# d'exemples	683	768	625	178	351	1000/2175	3089/4000
# de classes	2	2	3	3	2	2	2
# d'attributs	10	8	4	13	34	60	4
# d'attributs après KPCA	30	24	16	39	102	180	16

TABLE 1 – Principales caractéristiques des différents jeux de données utilisés. Pour Splice et Svmguide-1 le nombre d'exemples indiqué correspond à exemples d'apprentissage/exemples de test.

après avoir effectué une KPCA sur les données.
- LMNN-KPCA : la métrique est apprise en utilisant LMNN après avoir effectué une KPCA sur les données.
- ITML-KPCA : la métrique est apprise en utilisant ITML après avoir effectué une KPCA sur les données.
- GB-LMNN : la métrique est apprise en utilisant GB-LMNN [KTW+12].
- RML-RBF-OT : notre approche en utilisant un noyau gaussien et la première méthode de sélection des points virtuels basée sur le transport optimal. La taille du sous-ensemble d'exemples utilisé est fixée par cross-validation entre 5, 10, 15, 20 et 25 % des données d'apprentissage.
- RML-RBF-Vect : notre approche en utilisant un noyau gaussien et la seconde méthode de sélection des points virtuels basée sur les classes des exemples. Notons que dans ce cas la dimension des exemples après apprentissage est réduite.

La KPCA est calculée avec un noyau gaussien où sigma est fixé à la moyenne des distances entre les exemples d'apprentissage comme proposé dans [BHS12]. Nous utilisons la même heuristique pour notre approche. Le paramètre λ de notre approche est fixé par validation croisée dans $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$.

Pour les méthodes linéaires, les résultats, obtenus en termes de pourcentage de classifications correctes, sont présentés dans le tableau 2. Notre approche utilisant la première méthode de sélection des points virtuels donne dans 5 cas sur 7 les meilleurs ou seconds meilleurs résultats. Notons que dans les deux cas où notre approche n'est pas présente parmi les meilleurs méthodes, c'est l'algorithme du plus proche voisin basé sur la distance euclidienne qui s'en sort le mieux et toutes les autres méthodes entrainent une baisse de performance. Nous interprétons cela comme un signe que ces deux jeux de données ne sont pas adaptés pour l'apprentissage de métriques linéaires. Dans le cas linéaire, notre approche qui utilise des points virtuels définissant un nouvel espace de représentation de plus petite dimension présente des résultats légèrement moins bons que les autres méthodes. Nous expliquons cela par le fait que nous réduisons très fortement la dimension de représentation des exemples, par exemple pour Ionosphère nous passons de 34 à 2 dimensions.

Les résultats pour les méthodes non linéaires sont présentés dans le tableau 3. Dans ce cas, nos deux méthodes monopolisent au moins l'une des deux premières places sur tous les jeux de données et obtiennent le meilleur résultat sur 6 des 7 bases. Cela montre l'intérêt de notre approche dans le cadre de l'apprentissage d'une métrique non linéaire. Notons que contrairement au cas linéaire, notre seconde approche qui réduit la dimension des exemples n'est pas pénalisée par cette diminution du nombre d'attributs. Au contraire, elle obtient les meilleurs résultats sur 2 jeux de données et fait partie des deux premières méthodes sur 5 des 7 jeux de données.

Dans cette section nous avons montré que notre approche utilisant une combinaison linéaire des données en points virtuels est performante que ce soit dans un cadre linéaire ou non linéaire. Pour notre méthode qui définit un espace de représentation de plus petite dimension les résultats sont légèrement moins bons dans le cas de l'apprentissage d'une métrique linéaire, au contraire ils sont très compétitifs dans le cas d'une métrique non linéaire. Notons que la réduction de dimension opérée est très forte puisque l'espace des points virtuels est défini par 2 ou 3 dimensions selon les jeux de données. Pour terminer, nous avons constaté que notre approche était substantiellement plus rapide en termes de temps d'exécution notamment grâce à l'utilisation d'un nombre linéaire de contraintes.

5 Conclusion

Nous avons présenté une nouvelle méthode d'apprentissage de métriques pouvant être facilement kernelisée. Au lieu d'utiliser un problème convexe avec les contraintes de type must-link et cannot-link habituelles, nous proposons d'utiliser une simple régression pour rapprocher les exemples d'apprentissage de points vir-

	Baselines			Notre approche	
Base	Id	LMNN	ITML	RML-Linear-OT	RML-Linear-Vect
Breast	**95.49 ± 0.79**	95.49 ± 0.89	95.39 ± 0.89	95.10 ± 1.51	95.44 ± 1.17
Pima	69.91 ± 1.69	70.04 ± 2.20	*70.43 ± 1.54*	**71.26 ± 2.24**	70.09 ± 2.32
Scale	78.68 ± 2.66	78.25 ± 1.89	88.99 ± 2.99	**89.58 ± 2.37**	87.83 ± 2.16
Wine	96.18 ± 1.59	**98.36 ± 1.03**	96.36 ± 2.42	98.18 ± 1.48	98.18 ± 1.48
Ionosphere	86.23 ± 1.95	**88.1 ± 2.82**	86.79 ± 2.13	87.64 ± 3.49	83.11 ± 3.71
Splice	71.17	82.02	73.24	**83.95**	78.44
Svmguide-1	**95.12**	95.03	95.10	94.10	83.30

TABLE 2 – Performance des métriques linéaires. Pour chaque jeu de données les deux meilleurs résultats sont respectivement intégrés en gras et en italique.

	Baselines				Notre approche	
Base	Id-KPCA	LMNN-KPCA	ITML-KPCA	GB-LMNN	RML-RBF-OT	RML-RBF-Vect
Breast	90.68 ± 2.73	95.10 ± 1.63	91.94 ± 2.12	95.58 ± 0.87	95.97 ± 1.00	**96.07 ± 0.96**
Pima	69.61 ± 1.99	68.83 ± 3.03	69.35 ± 2.22	69.52 ± 2.27	**71.73 ± 2.16**	70.56 ± 4.34
Scale	78.36 ± 0.88	88.10 ± 2.26	86.19 ± 2.49	77.78 ± 2.41	**94.92 ± 1.92**	94.60 ± 1.67
Wine	88.55 ± 3.84	94.00 ± 3.43	91.64 ± 4.55	98.00 ± 1.34	**98.36 ± 1.81**	98.18 ± 1.71
Ionosphere	74.72 ± 1.59	82.55 ± 3.39	75.38 ± 1.44	87.45 ± 2.85	91.51 ± 2.63	**91.98 ± 2.82**
Splice	65.93	**89.84**	66.34	82.25	88.41	88.32
Svmguide-1	95.72	95.60	95.67	95.00	**96.03**	95.58

TABLE 3 – Performance des métriques non linéaires. Pour chaque jeu de données les deux meilleurs résultats sont respectivement intégrés en gras et en italique.

tuels. Nous présentons deux solutions pour choisir ces points virtuels. La première est basée sur l'utilisation d'un sous-ensemble des données d'apprentissage tandis que la seconde propose d'apprendre un espace de représentation où chaque point virtuel correspond à un vecteur unitaire. Nous avons montré qu'il est possible de relier notre approche à un algorithme plus classique d'apprentissage de métriques. Enfin, nous avons obtenus de bons résultats empiriques que ce soit pour l'apprentissage d'une métrique linéaire ou non linéaire. Nous pensons que cette approche peut permettre la conception de nouvelles méthodes d'apprentissage de métriques pouvant tirer parti des points virtuels pour définir d'autres contraintes sur les exemples en comparaison des approches classiques must-link et cannot-link.

Pour les perspectives de nos travaux, nous envisageons de développer une version en ligne efficace qui pourrait s'inspirer de travaux existants sur les problèmes de régression. D'autre part, d'un point de vue théorique, nous pensons qu'il est possible de relier la notion de marge utilisée dans les approches classiques avec une information de distance entre les points virtuels. Par ailleurs, nous souhaitons étudier la possibilité de dériver des garanties de consistance.

Références

[BBS08] Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. Improved guarantees for learning via similarity functions. In *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 287–298, 2008.

[BHS12] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Similarity learning for provably accurate sparse linear classification. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[BHS13] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

[CFT14] Nicolas Courty, Rémi Flamary, and Devis Tuia. Domain adaptation with regularized optimal transport. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014,*

Nancy, France, September 15-19, 2014. Proceedings, Part I, pages 274–289, 2014.

[CMW05] Corinna Cortes, Mehryar Mohri, and Jason Weston. A general regression technique for learning transductions. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, pages 153–160, 2005.

[CMW07] Corinna Cortes, Mehryar Mohri, and Jason Weston. A general regression framework for learning string-to-string mappings. In *Predicting Structured Data*. MIT Press, 2007.

[Cut13] Marco Cuturi. Sinkhorn distances : Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems 26*, pages 2292–2300, 2013.

[DKJ+07] Jason V. Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S. Dhillon. Information-theoretic metric learning. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, pages 209–216, 2007.

[GR05] Amir Globerson and Sam T. Roweis. Metric learning by collapsing classes. In *Advances in Neural Information Processing Systems 18*, pages 451–458, 2005.

[GRHS04] Jacob Goldberger, Sam T. Roweis, Geoffrey E. Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520, 2004.

[JWZ09] Rong Jin, Shijun Wang, and Yang Zhou. Regularized distance metric learning : Theory and algorithm. In *Advances in Neural Information Processing Systems 22*, pages 862–870, 2009.

[KGP13] Hachem Kadri, Mohammad Ghavamzadeh, and Philippe Preux. A generalized kernel approach to structured output learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 471–479, 2013.

[KJ11] Purushottam Kar and Prateek Jain. Similarity-based learning via data driven embeddings. In *Advances in Neural Information Processing Systems 24*, pages 1998–2006, 2011.

[KJ12] Purushottam Kar and Prateek Jain. Supervised learning with similarity functions. In *Advances in Neural Information Processing Systems 25*, pages 215–223, 2012.

[KTW+12] Dor Kedem, Stephen Tyree, Kilian Q. Weinberger, Fei Sha, and Gert R. G. Lanckriet. Non-linear metric learning. In *Advances in Neural Information Processing Systems 25*, pages 2582–2590, 2012.

[Kul13] Brian Kulis. Metric learning : A survey. *Foundations and Trends in Machine Learning*, 5(4) :287–364, 2013.

[SSM97] Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Artificial Neural Networks - ICANN '97, 7th International Conference, Lausanne, Switzerland, October 8-10, 1997, Proceedings*, pages 583–588, 1997.

[Vil08] Cédric Villani. *Optimal transport : old and new*, volume 338. Springer Science & Business Media, 2008.

[WBS05] Kilian Q. Weinberger, John Blitzer, and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems 18*, pages 1473–1480, 2005.

[WCE+02] Jason Weston, Olivier Chapelle, André Elisseeff, Bernhard Schölkopf, and Vladimir Vapnik. Kernel dependency estimation. In *Advances in Neural Information Processing Systems 15*, pages 873–880, 2002.

[WS09] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10 :207–244, 2009.

[WT07] Kilian Q. Weinberger and Gerald Tesauro. Metric learning for kernel regression. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, AISTATS 2007, San Juan, Puerto Rico, March 21-24, 2007*, pages 612–619, 2007.