

# Apprentissage de fonctions d'ordonnement semi-supervisé inductives

Vinh Truong, Massih-Reza Amini et Patrick Gallinari

Laboratoire d'Informatique de Paris 6  
Université Pierre et Marie Curie  
104 avenue du Président Kennedy 75016 Paris  
{truong, amini, gallinari}@poleia.lip6.fr

## Résumé :

Nous proposons dans ce papier une méthode inductive d'apprentissage de fonctions d'ordonnement avec des données partiellement étiquetées. Les problèmes d'ordonnement considérés ici sont des problèmes *bipartites* où il existe une information désirée fixe pour laquelle on cherche à ordonner les instances pertinentes par rapport à cette information, au-dessus des instances non-pertinentes. Pour résoudre ce genre de problème les techniques existantes sont basées sur des méthodes transductives. Elles commencent généralement avec un graphe de similarité entre l'ensemble des données d'une base et cherchent ensuite à propager les scores des données étiquetées pertinentes sur le graphe. Notre approche est basée sur l'hypothèse des clusters (*cluster assumption*) proposée en classification semi-supervisée. Elle commence à partitionner l'ensemble des exemples d'une base d'apprentissage et elle apprend ensuite une fonction de score régularisée qui pénalise les fortes variations entre deux exemples appartenant à une même partition. Pour apprendre cette fonction de score, nous utilisons la fonction  $\epsilon$ -insensitive hinge, qui permet de formaliser le problème d'optimisation comme une forme duale des Séparateurs à Vaste Marge (SVM). Une fois la fonction de score apprise, les exemples d'un nouvel ensemble pourront être ordonnés en fonction de la sortie de cette fonction. Les expériences menées sur des collections de l'état de l'art montrent que les données non-étiquetées permettent d'améliorer les performances en ordonnancement par rapport à une fonction de base apprise qu'avec des exemples étiquetés.

**Mots-clés** : ordonnancement bipartite, apprentissage semi-supervisé.

## 1 Introduction

Le paradigme de l'apprentissage semi-supervisé a suscité un grand intérêt dans la communauté d'apprentissage depuis la fin des années 90 (Blum & Mitchell, 1998). Ces travaux ont été motivés, pour la plupart, par des applications en Recherche d'Information pour lesquelles il est généralement difficile de disposer d'une base d'exemples étiquetés. L'ensemble de ces travaux s'est formalisé exclusivement autour des cadres de

la classification et de la régression (Chapelle *et al.*, 2006; Zhu, 2007; Amini & Gallinari, 2003).

Il existe cependant des applications pour lesquelles le but est d'ordonner les exemples et pour lesquelles l'étiquetage de ces derniers restent une tâche difficile. On peut considérer par exemple le cas du routage de l'information, où pour un profil stable donné et un flux de documents entrants, les documents pertinents par rapport à ce profil, doivent être ordonnés au-dessus des documents non-pertinents (Iyer *et al.*, 2000; Christopher D. Manning, Prabhakar Raghavan and Hinrich SchütZ, 2008; Robertson & Soboroff, 2001). La constitution d'une base étiquetée dans ce cas nécessite l'étiquetage de l'ensemble des documents entrants qui s'avère en général impossible.

Récemment plusieurs travaux se sont intéressés à ce genre de problèmes et ont proposé d'apprendre des fonctions d'ordonnement dans un cas transductif (Agarwal & Chakrabarti, 2007; Agarwal, 2006; Chu & Ghahramani, 2005; Weston J. & W.S., 2006; Zhou *et al.*, 2004b). Ces méthodes considèrent généralement une structure de graphe dont les sommets représentent les données et dont le but est de propager l'information des exemples étiquetés sur l'ensemble du graphe. Une fois cette fonction apprise, les exemples non-étiquetés de la base sont ordonnés en utilisant les valeurs de scores ainsi obtenues. Il est à noter qu'avec ce cadre, pour chaque nouvel ensemble, la fonction de score recherchée doit être à nouveau re-estimée. Ces méthodes bien qu'elles ont montré leur efficacité par rapport à des méthodes supervisées pures, ne sont cependant pas adaptées au cas où on cherche à inférer des scores aux nouveaux exemples entrants comme pour l'application du routage d'information évoquée plus haut.

Nous proposons dans ce papier un nouvel algorithme d'ordonnement semi-supervisé capable d'apprendre une fonction de score inductive. Notre méthode se base sur l'hypothèse des clusters (*cluster assumption*) qui spécifie que deux exemples appartenant à un même *cluster* doivent avoir le même jugement de pertinence. Pour découvrir ces structures, nous utilisons un algorithme non-supervisé. Nous essayons ensuite d'apprendre une fonction réelle de sorte que les exemples d'une même partition aient des valeurs de scores similaires. Pour cela, la fonction objective permettant d'atteindre cette solution est constituée de deux parties. La première, représente l'erreur empirique d'ordonnement des données étiquetées de la base d'apprentissage et la seconde est un terme de régularisation qui mesure à quel point la fonction apprise alloue des scores divergents aux exemples appartenant aux différentes partitions obtenues initialement. Nous considérons ensuite une fonction de perte de type *hinge loss* utilisée en régression pour des vecteurs supports (appelée  *$\epsilon$ -insensitive hinge*) dans la définition de cette fonction objective. Cette approche permet ainsi (a) de mettre le problème d'optimisation sous une forme duale comme dans le cas des SVM et (b) d'utiliser des solveurs existants pour trouver la fonction de score recherchée.

Le plan de ce papier est comme suit ; nous allons dans un premier temps aborder le formalisme de l'ordonnement bipartite supervisé (section 2). Nous présenterons ensuite notre approche d'apprentissage inductif de fonctions d'ordonnement (section 3) et discuterons des résultats obtenus sur différentes bases réelles à la section 4. La conclusion de ce travail est donnée à la section 5.

## 2 Ordonnancement bipartite supervisé

Nous nous intéressons à la suite au problème de l'ordonnancement *bipartite* qui est un cas particulier de la tâche d'ordonnancement d'instances. En ordonnancement d'instances, on cherche à ordonner les

exemples en allouant un score plus élevé à un exemple  $x$  qu'à un exemple  $x'$ , si  $x$  est préféré à  $x'$ . Dans ce cas, à chaque observation  $x \in \mathcal{X} \subseteq \mathbb{R}^d$  est associée une valeur réelle de sortie  $y \in \mathbb{R}$ , qui reflète un jugement de pertinence. Les couples d'exemples  $(x, y)$  sont supposés être générés i.i.d suivant une distribution  $\mathcal{D}$  inconnue mais fixée. La fonction de score  $h : \mathcal{X} \rightarrow \mathbb{R}$  que l'on cherche à apprendre associe ainsi une sortie réelle à chaque observation en entrée. Pour un ensemble d'apprentissage  $\mathcal{L} = \{(x_i, y_i)\}_{i \in \{1, \dots, n\}}$  échantillonné i.i.d suivant  $\mathcal{D}$  et constitué de  $n$  couples (observation, sortie réelle), on définit le coût d'ordonnancement par une fonction  $L_o : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_+$ . Le risque empirique de  $h$  sur  $\mathcal{L}$  est alors défini par

$$\widehat{\mathcal{R}}_n(h, \mathcal{L}) = L_o(h(X), Y)$$

Où  $h(X) = (h(x_1), \dots, h(x_n))$  avec  $X$  l'ensemble des observations de la base d'apprentissage et  $Y$  l'ensemble des sorties associées.

Dans le cas de l'ordonnancement bipartite, les sorties associées aux observations sont des valeurs discrètes dans  $\{-1, 1\}$ . L'exemple du routage d'information cité dans l'introduction est plus précisément issu de ce cadre. Sur un ensemble d'apprentissage  $\mathcal{L}$  de taille  $n$ , l'erreur empirique d'une fonction de score  $h$  sur  $\mathcal{L}$  est définie comme le nombre moyen d'exemples pertinents mal-ordonnés par  $h$  :

$$\widehat{\mathcal{R}}_n(h, \mathcal{L}) = \frac{1}{|\mathcal{S}_1| \cdot |\mathcal{S}_{-1}|} \sum_{x \in \mathcal{S}_1} \sum_{x' \in \mathcal{S}_{-1}} \llbracket h(x) < h(x') \rrbracket \quad (1)$$

où  $\llbracket pr \rrbracket$  est égal à 1 si le prédicat  $pr$  est vrai et 0 sinon,  $\mathcal{S}_1$  (resp.  $\mathcal{S}_{-1}$ ) représente l'ensemble des instances positives (resp. négatives) de la base d'apprentissage. On appelle paires cruciales, les paires  $(x, x') \in \mathcal{S}_1 \times \mathcal{S}_{-1}$  constituées chacune d'une instance pertinente et d'une instance non-pertinente.

Ce cadre d'ordonnancement bipartite a été le plus étudié dans la littérature aussi bien d'un point de vue pratique que théorique (Agarwal & Niyogi, 2005; Agarwal *et al.*, 2005; Freund *et al.*, 2003). Une propriété intéressante de ce cadre est le lien direct entre le risque empirique  $\widehat{\mathcal{R}}_n(h, \mathcal{L})$  d'une fonction de score  $h$  sur un ensemble d'apprentissage  $\mathcal{L}$  et l'aire sous la courbe ROC (AUC) de cette fonction sur  $\mathcal{L}$  (Cortes & Mohri, 2004) :

$$\begin{aligned} \text{AUC}(h, \mathcal{L}) &= \frac{1}{|\mathcal{S}_1| \cdot |\mathcal{S}_{-1}|} \sum_{x \in \mathcal{S}_1} \sum_{x' \in \mathcal{S}_{-1}} \left( \llbracket h(x) > h(x') \rrbracket + \frac{1}{2} \llbracket h(x) = h(x') \rrbracket \right) \\ &\geq 1 - \widehat{\mathcal{R}}_n(h, \mathcal{L}) \end{aligned}$$

L'optimisation de la fonction (1) revient à résoudre un problème d'optimisation discrète et n'est donc pas faisable analytiquement. La solution proposée dans (Freund *et al.*,

2003) est de remplacer ce coût par une fonction convexe  $\Delta(x, x', h)$  (appelée fonction de perte) le majorant. La version régularisée de cette nouvelle fonction de coût peut s'écrire comme :

$$\widehat{\mathcal{R}}_n(h, \mathcal{L}) = \lambda \|h\|_{\mathcal{H}}^2 + \frac{1}{|\mathcal{S}_1| \cdot |\mathcal{S}_{-1}|} \sum_{x \in \mathcal{S}_1} \sum_{x' \in \mathcal{S}_{-1}} \Delta(x, x', h) \quad (2)$$

Où  $\lambda$  représente le terme de régularisation et  $\mathcal{H}$  représente l'espace de recherche fonctionnel muni de la norme  $\|\cdot\|_{\mathcal{H}}$ .

### 3 Ordonnement bipartite semi-supervisé

#### 3.1 Algorithme Semi-CRank

Dans le contexte semi-supervisé, nous supposons qu'en plus des exemples étiquetés de  $\mathcal{L}$ , le système dispose d'un ensemble supplémentaire constitué de  $m$  instances non-étiquetées  $\mathcal{U} = \{x_u\}_{u=1+n}^{n+m}$ . Le but est alors de trouver une fonction de score  $h$  en utilisant l'information contenue dans les deux ensembles  $\mathcal{L}$  et  $\mathcal{U}$ . Pour exploiter l'information contenue dans  $\mathcal{U}$  nous suivons l'hypothèse  $H_0$  qui est que *deux instances non-étiquetées appartenant à une même partition (cluster) doivent avoir des jugements de préférences similaires*.

Suivant cette hypothèse, la fonction objective que nous proposons d'optimiser dans le cas d'ordonnement bipartite semi-supervisé se met sous la forme suivante :

$$\Lambda_{n+m}(h, \mathcal{L} \cup \mathcal{U}) = \widehat{\mathcal{R}}_n(h, \mathcal{L}) + \frac{\mu}{K} \sum_{C_k \in \mathcal{C}} \sum_{(x_u, x_v) \in C_k} \omega_{uv} \pi(x_u, x_v, h) \quad (3)$$

Le premier terme de cette sommation est l'erreur empirique d'ordonnement des données de  $\mathcal{L}$  comme celle présentée à la section 2 et le second, est un terme de régularisation calculé sur les données non-étiquetées. Le facteur multiplicatif  $\mu$  permet de pondérer l'apport de ce terme dans la fonction objective.

Pour estimer ce terme, nous utilisons d'abord un algorithme de partitionnement  $\mathcal{A}_c$  qui prend en entrée l'ensemble des données étiquetées et non-étiquetées de la base d'apprentissage et qui retourne en sortie  $K$  partitions de ces données. Nous notons les partitions ainsi obtenues par  $\mathcal{C}$  :

$$\mathcal{C} = \{C_1, \dots, C_K | C_k \in \mathcal{P}(\mathcal{L} \cup \mathcal{U})\}$$

Où  $\mathcal{P}(\mathcal{L} \cup \mathcal{U})$  est l'ensemble des parties de  $\mathcal{L} \cup \mathcal{U}$ .

Sur la base de ces partitions  $\mathcal{C}$ , nous estimons une fonction  $\pi : \mathcal{X} \times \mathcal{X} \times \mathcal{H} \rightarrow \mathbb{R}^+$  qui pénalise la fonction apprise  $h \in \mathcal{H}$  si cette dernière assigne des scores trop divergents aux exemples non-étiquetés contenus dans les mêmes partitions. Pour renforcer l'effet de l'hypothèse  $H_0$  dans la recherche de  $h$ , nous pondérons en plus la fonction de pénalité  $\pi$  par des coefficients  $\omega_{uv}$  qui mesurent le degré d'appartenance de deux exemples

non-étiquetés  $x_u$  et  $x_v$  à un cluster  $C_k \in \mathcal{C}$  donné. Dans nos expériences nous utilisons la mesure de similarité la plus utilisée dans les travaux d'apprentissage de classifieurs semi-supervisés menés dans le cadre transductif :

$$\omega_{uv} = \exp\left(-\frac{d(x_u, x_v)^2}{2\sigma^2}\right) \quad (4)$$

Où  $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$  est une distance (généralement la distance euclidienne) et  $\sigma$  une constante fixée à l'avance.

Les différentes étapes d'apprentissage de la fonction de score recherchée sont résumées dans l'algorithme suivant (appelé *Semi-CRank*) :

---

**Algorithm 1** Squelette de l'algorithme *Semi-CRank*

---

**ENTRÉES:** Un ensemble de données étiquetées  $\mathcal{L}$  et un ensemble de données non-étiquetées  $\mathcal{U}$

1: Partitionnement des données  $\mathcal{L} \cup \mathcal{U}$  avec un algorithme  $\mathcal{A}_c$

$$\mathcal{C} \leftarrow \mathcal{A}_c(\mathcal{L} \cup \mathcal{U})$$

2: Calcul de la matrice de similarité  $W = (\omega_{ij})$  à partir de  $\mathcal{C}$

3: Optimisation de la fonction régularisée  $\Lambda$  (équation 3) :

$$h \leftarrow \underset{h \in \mathcal{H}}{\operatorname{argmin}} \widehat{\mathcal{R}}_n(h, \mathcal{L}) + \frac{\mu}{K} \sum_{C_k \in \mathcal{C}} \sum_{(x_u, x_v) \in C_k} \omega_{uv} \pi(x_u, x_v, h)$$

**SORTIE:** La fonction de score  $h$

---

*Lien avec les méthodes à base de variété*

Dans le cas d'apprentissage transductif de fonctions de classification (Belkin *et al.*, 2006; X. Zhu and Z. Ghahramani and J. Lafferty, 2003), les méthodes faisant l'hypothèse de variété utilisent aussi un terme de régularisation pour la partie non-étiquetée de la fonction objective. Ce terme correspond à un terme de lissage sur la variété. Cette structure est en fait approchée par un graphe dont les sommets représentent les exemples et les arcs relient un point avec un des ses  $k$  plus proches voisins. La régularisation obtenue est alors faite au moyen d'un terme de pénalité quadratique  $\pi(x_u, x_v, h) = (h(x_u) - h(x_v))^2$ . D'autres termes de régularisation à base du laplacien normalisé ont été aussi proposés dans la littérature (Zhou *et al.*, 2004a) et sont communément utilisés dans les méthodes à base de variétés géométriques. Inspirés des travaux menés en classification, les méthodes d'ordonnement transductives dans (Agarwal, 2006; Weston J. & W.S., 2006; Zhou *et al.*, 2004b) exploitent aussi l'un des deux termes de régularisation.

## 3.2 Optimisation

Dans la partie précédente, nous avons présenté un cadre général pour apprendre une fonction d'ordonnement à partir d'exemples étiquetés et non-étiquetés. Nous allons maintenant présenter une instantiation de cet algorithme avec les fonctions de perte *hinge* pour l'erreur d'ordonnement sur l'ensemble étiqueté et  $\epsilon$ -insensitive *hinge* sur les paires issues des partitions trouvées par  $\mathcal{A}_c$ .

### 3.2.1 Forme primale

Nous considérons dans nos expériences, la classe des fonctions linéaires de la forme  $\mathcal{H}_L = \{h_\beta : \mathcal{X} \rightarrow \mathbb{R} \mid \forall x \in \mathcal{X}, h(x) = \langle \beta, x \rangle \text{ avec } \beta \in \mathbb{R}^d\}$ . Ces fonctions sont en effet très utilisées dans la plupart des applications en Recherche d'Information (RI) où il a été trouvé que les fonctions numériques performantes pour ces applications représentées par des espaces à grande dimension, sont celles qui ont une faible variance (Lebanon & Lafferty, 2004). Avec les fonctions linéaires, l'inférence est aussi une procédure peu coûteuse et présente par exemple un avantage pour des systèmes de routage d'information, qui ont besoin de traiter rapidement un grand nombre de données. Notre cadre n'est cependant pas restreint à la classe des fonctions linéaires et il peut facilement être utilisé avec des fonctions plus complexes (section 3.2.3).

Nous posons en plus des notations introduites précédemment,  $\overline{\mathcal{S}}_l$  l'ensemble des éléments dans  $\mathcal{S}_1 \times \mathcal{S}_{-1}$  et  $\overline{\mathcal{C}}$  l'ensemble des couples d'éléments dans une partition donnée trouvée par  $\mathcal{A}_c$ . En instanciant l'équation (3) par la définition de (2) et la forme de la classe de fonctions utilisée, l'apprentissage d'une fonction de score  $h_\beta \in \mathcal{H}_L$ , peut ainsi être formulée comme l'optimisation de la fonction de coût suivante :

$$\min_{\beta} \frac{1}{2} \|\beta\|^2 + C \sum_{(x_i, x_j) \in \overline{\mathcal{S}}_l} \Delta(x_i, x_j, h_\beta) + C' \sum_{(x_u, x_v) \in \overline{\mathcal{C}}} \omega_{uv} \pi(x_u, x_v, h_\beta) \quad (5)$$

Où dans ce cas,  $\lambda = \frac{1}{2}$ ,  $C = \frac{1}{|\overline{\mathcal{S}}_l|}$  et  $C' = \frac{\mu}{K}$ .

Nous utilisons à la place de la fonction de perte  $\Delta(x, x', h_\beta)$  pour les données étiquetées, la fonction de coût *hinge*  $H_a(t) = \max(0, a - t)$  évaluée au point  $t = h_\beta(x) - h_\beta(x')$ . Cette fonction a été proposée en ordonnancement supervisée par (Herbrich *et al.*, 2000). Cette fonction pénalise la différence de score  $h(x) - h(x')$  pour une paire cruciale  $(x, x')$  si cette dernière est inférieure à  $a$ .

Pour le terme de régularisation, nous employons la fonction *hinge* avec une tolérance de  $\epsilon$  notée  $U_\epsilon(t)$  évaluée au point  $h_\beta(x_u) - h_\beta(x_v)$ . Cette fonction vaut zéro dans l'intervalle  $[-\epsilon, \epsilon]$  et, est décroissante à gauche et croissante à droite en dehors de ce dernier :

$$U_\epsilon(t) = \begin{cases} 0 & \text{si } t \in [-\epsilon, \epsilon] \\ |t| - \epsilon & \text{sinon} \end{cases}$$

La fonction  $U_\epsilon(t)$  peut être vue comme la somme de deux fonctions *hinge* :  $U_\epsilon(t) = H_{-\epsilon}(t) + H_{-\epsilon}(-t)$  permettant de reformuler le problème d'optimisation sous une forme

proche de celle du SVM qui avec des variables ressorts  $\xi_{ij}, \theta_{uv}$  et  $\theta_{uv}^*$  mène à un problème d'optimisation sous contraintes (ou primal) :

$$\min_{\beta} \quad \frac{1}{2} \|\beta\|^2 + C \sum \xi_{ij} + C' \sum (\theta_{uv} + \theta_{uv}^*) \quad (6)$$

$$\text{s.c.} \quad \langle \beta, x_i - x_j \rangle \geq 1 - \xi_{ij} \text{ avec } (x_i, x_j) \in \overline{\mathcal{S}} \quad (7)$$

$$\omega_{uv} \langle \beta, x_u - x_v \rangle \geq -\epsilon - \theta_{uv} \text{ avec } (x_u, x_v) \in \overline{\mathcal{C}} \quad (8)$$

$$-\omega_{uv} \langle \beta, x_u - x_v \rangle \geq -\epsilon - \theta_{uv}^* \text{ avec } (x_u, x_v) \in \overline{\mathcal{C}} \quad (9)$$

$$\xi_{ij} > 0, \theta_{uv} > 0, \theta_{uv}^* > 0 \quad (10)$$

### 3.2.2 Forme duale

Pour résoudre le problème primal précédent, nous y introduisons des multiplicateurs de Lagrange qui permet d'obtenir la forme duale suivante (cf. annexe pour plus de détails) :

$$\begin{aligned} \max_{\alpha_i} \quad & \sum_{i=1}^N \kappa_i \alpha_i - \frac{1}{2} \sum_{i,j=1}^N y_i y_j \alpha_i \alpha_j \langle z_i, z_j \rangle \\ \text{sc} \quad & 0 \leq \alpha_i \leq C \text{ pour } i = 1, \dots, |\overline{\mathcal{S}}| \\ & 0 \leq \alpha_i \leq C' \text{ pour } i = |\overline{\mathcal{S}}| + 1, \dots, N \end{aligned}$$

Où  $N$  est égal aux nombres de contraintes faisant intervenir une paire d'instances dans la forme primale,  $N = |\overline{\mathcal{S}}| + 2|\overline{\mathcal{C}}|$ . Les  $\kappa_i$  sont des coefficients réels valant 1 ou  $-\epsilon$ . Les  $z_i$  représentent les paires d'exemples (pondérées si elles proviennent d'une même partition) et les étiquettes  $y_i \in \{-1, 1\}$  traduisent les conditions (8) et (9) du problème primal.

L'avantage de cette formulation est qu'elle peut être résolue d'une manière classique avec un solveur SVM (en enlevant le biais qui est non nul dans le cas de la discrimination et qui ne nous sert pas dans le cas où on cherche des scores relatifs aux données). La solution à notre problème peut ainsi s'exprimer sous la forme d'une combinaison linéaire de paires d'exemples. En utilisant la propriété de distributivité du produit scalaire nous pouvons voir que la solution s'écrit finalement sous la forme suivante :

$$h(x) = \sum_{j=1}^{n+m} \alpha'_j \langle x, x_j \rangle$$

avec  $\alpha'_{ij} \in \mathbb{R}$ . L'équation permet ainsi de faire ressortir la notion de vecteurs supports pour l'ordonnancement semi-supervisé.

### 3.2.3 Extension au cas non-linéaire

Le cas non-linéaire peut être traité en utilisant l'*astuce noyau*. La forme duale ne fait en effet intervenir que des produits scalaires entre exemples. Ce dernier peut être remplacé par n'importe quel noyau de Mercer  $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ . En fusionnant les

variables  $\alpha_i$  et  $y_i$  en  $\theta_i$ , nous pouvons écrire le problème dual sous la forme matricielle suivante :

$$\begin{aligned} \max \quad & \kappa^T B - \frac{1}{2} \mathbf{K} B \\ \text{sc} \quad & 0 \leq y_i \beta_i \leq C \text{ pour } i = 1, \dots, |S| \\ & 0 \leq y_i \beta_i \leq C' \text{ pour } i = 1 + |S| + N \end{aligned}$$

Où  $B$  est le vecteur défini par  $B_i = y_i \beta_i$ ,  $\kappa$  est le vecteur dont les composantes sont les  $\kappa_i$  et  $\mathbf{K}$ , la matrice dont chaque élément vaut  $k(x_u, x'_u) + k(x_v, x'_v) - k(x_u, x'_v) - k(x_v, x'_u)$ . Cette matrice joue le rôle de la matrice de Gram en discrimination à la différence qu'elle correspond aux paires d'exemples et non aux exemples individuels.

### 3.2.4 Remarques

Dans le cas supervisé, l'ordonnement peut être considéré comme de la classification de paires cruciales. Le cadre que nous avons développé, diffère de l'extension uniforme du cadre supervisé au cas semi-supervisé, dans le choix des critères différents pour les deux parties étiquetée et non-étiquetée de la fonction objective. En effet, l'information de paires cruciales n'est prise en compte que sur les données étiquetées. Et, pour la partie non-étiquetée, nous essayons de prendre en compte la similarité entre une paire d'exemples (grâce à l'algorithme de partitionnement  $\mathcal{A}_c$ ) via un terme de régularisation .

En discrimination, une étude récente s'est intéressée à l'apprentissage avec des exemples ne provenant pas de la même distribution que les données étiquetées appelés l'Universum (Weston *et al.*, 2006; Sinz *et al.*, 2008). Ces *extra*-données proviennent néanmoins du même domaine que l'application considérée et sont supposées contenir de l'information utile à la tâche de discrimination étudiée. Dans (Weston *et al.*, 2006), les auteurs ont proposé une méthode inductive qui trouve une fonction telle que les exemples de l'Universum se trouvent près de la frontière de décision. Pour ce faire, ils proposent d'appliquer une fonction de perte qui pénalise les exemples dont la sortie de la fonction est loin de 0.

Pour l'ordonnement semi-supervisé, une paire d'exemples non-étiquetés peut ainsi être considérée comme un élément de l'Universum. Dans notre étude et dans celle de (Agarwal (2006)), forcer deux exemples similaires à avoir des scores similaires revient à forcer cette paire d'exemples à se trouver près de la frontière de décision.

## 3.3 Complexité

La méthode proposée peut ainsi se décomposer en trois blocs distincts : la recherche des partitions par  $\mathcal{A}_c$ , le calcul des poids (équation 4) et enfin la résolution du problème

sous la forme duale. La complexité dépend en fait de la méthode non supervisée utilisée, puisqu'elle permet de générer les paires d'exemples que nous utilisons par la suite. Pour en avoir une idée, nous allons étudier le cas où  $\mathcal{A}_c$  est l'algorithme des centres mobiles ( $k$ -moyennes).

Le calcul des complexités est donné dans le tableau 1. Expérimentalement, l'algorithme des  $k$ -moyennes converge rapidement (en quelques itérations). Cependant, s'il y a peu de centres, l'algorithme peut fournir un grand nombre de paires. Dans le cas extrême ( $k = 1$ ), le nombre de paires est de l'ordre de  $m^2$ , ce qui peut rapidement devenir compliqué pour un solveur de programmation quadratique. Nous suggérons ici d'utiliser un grand nombre de centres pour diminuer le nombre de paires intervenant dans les calculs. Dans la partie expérimentale, nous avons utilisé par exemple  $K = \lfloor \frac{m}{p} \rfloor$ , avec  $\lfloor x \rfloor$ , la partie entière de  $x$  et  $p = 10$ . Dans ce cas la complexité de l'algorithme  $k$ -moyennes est de  $O(K \times m \times I)$  (Avec  $I$  le nombre d'itérations de l'algorithme) et le nombre de paires se réduit à  $pm$ . Au final, la complexité de l'algorithme peut être estimée à  $O((n + pm)^\gamma)$  avec  $\gamma \in [2, 3]$ . La complexité des différentes étapes de notre algorithme est résumée au tableau 1. Nous remarquons aussi que l'exemple montre ici que le nombre de partition est important et a une grande influence sur le coût final.

étape	complexité
k-moyenne	$O(Km)$
calcul de poids	$O(d \cdot \sum_k  \mathcal{C}_k ^2)$
Solveur SVM	$O(( \mathcal{S}  + 2 \sum_k  \mathcal{C}_k ^2)^\gamma)$ avec $\gamma \in [2, 3]$

TAB. 1 – tableaux récapitulatif de la complexité :  $m$  représente le nombre d'exemples non-étiquetés,  $I$  le nombre d'itérations,  $K$  le nombre de centres,  $|\mathcal{C}_k|$  le nombre de paires d'exemples dans la partition et  $|\mathcal{S}|$  le nombre de paires cruciales.

Nous pouvons noter que l'algorithme des  $k$ -moyennes peut se faire en ligne (Bottou & Bengio, 1994) pour réduire le coût au niveau de la mémoire et que la complexité de résolution du problème d'optimisation pourrait être diminuée en utilisant des études récentes sur ce sujet ( en particulier ceux concernant l'apprentissage sur des grandes bases de données). Par exemple des algorithmes comme *LaSVM* (Bordes *et al.*, 2005) ou *SimpleSVM* (Vishwanathan *et al.*, 2003) utilisent des contraintes actives et peuvent être adaptés à notre cadre.

## 4 Expériences

### 4.1 Protocole expérimental

Pour valider notre approche, nous avons utilisé plusieurs bases réelles : ces bases proviennent des tâches de discrimination et des applications en RI. Pour chacune des bases, nous avons formé aléatoirement 10 jeux de données constitués chacun d'un ensemble réduit d'exemples étiquetés, d'un ensemble plus large d'exemples non-étiquetés

et enfin d'un ensemble de test. Les deux premières bases forment ainsi la base d'apprentissage. En suivant les protocoles dans (Chapelle & Zien, 2005; Chapelle *et al.*, 2006), nous avons réglé les hyperparamètres sur l'ensemble de test en faisant une recherche exhaustive. Les résultats présentés permettent ainsi de montrer le *potentiel* de notre méthode. La sélection de modèles en apprentissage semi-supervisé constitue en effet un thème de recherche à part entière. Pour chacune des méthodes, nous avons sélectionné celui qui maximise l'AUC. La fonction objective est en effet basé sur ce critère.

L'algorithme désigné par *Semi-CRank* dans cette partie correspond à notre méthode en utilisant l'algorithme des  $k$ -moyennes comme algorithme de partitionnement et les fonctions de perte *hinge*. La similarité utilisée (équation 4) est la similarité gaussienne  $exp(-\frac{\|x_u-x_v\|^2}{2\sigma^2})$  avec  $\sigma$  initialisé en fonction de la moyenne des distances euclidiennes entre les exemples d'une paire. Pour l'ensemble des bases, nous avons utilisé un noyau linéaire. Notre méthode a été implémentée à partir de l'*universm* (Sinz, 2007) basé sur le solveur *SVQP2* de (Bottou, 2007). Nous avons laissé les valeurs par défaut pour les paramètres d'optimisation. En particulier,  $\epsilon$  est égal à 0.5 pour l'ensemble des expériences.

parametre	valeur
$k$	$\lfloor \frac{n}{10} \rfloor, \lfloor \frac{n}{5} \rfloor$
$C$	$10^{-4}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^4$
$C'$	$10^{-4}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^4$

TAB. 2 – Grille de recherche pour différents hyperparamètres du modèle.

Pour évaluer notre méthode, nous avons comparé les performances de notre approche avec des SVM appris sur les paires d'exemples cruciales (*RankSVM*). La comparaison a été faite par rapport aux mesures suivantes :

- *AUC* : Le nombre de paires cruciales bien ordonnées.
- *prec@k* : La précision obtenue si nous présentons à l'utilisateur les  $k$  instances les mieux ordonnées de la base test. Elle représente la proportion d'exemples positifs parmi ces  $k$  exemples. Nous avons fixé ce paramètre à 10,50 et 100.
- *précision moyenne* : la précision moyenne est la moyenne des précisions calculées en tronquant la liste ordonnée des exemples de test après chaque exemple positif :

$$AvPrec = \frac{1}{\#positive\ examples} \sum (Prec@r.rel(r))$$

avec  $rel(r)$  une fonction qui retourne 1 si  $r$  est un exemple positif et 0 sinon.

Ces mesures sont couramment utilisées dans les applications de RI : L'AUC et la précision moyenne sont des critères qui dépendent de la totalité des exemples étiquetés alors que *prec@k* concerne seulement les  $k$  premiers exemples.

## 4.2 Bases utilisées

Nous avons testé dans ce papier notre méthode sur plusieurs bases réelles : USPS<sup>1</sup>, COIL<sup>1</sup>, TEXT<sup>1</sup>, SEGMENT<sup>2</sup>, COIL<sup>1</sup>, RCV1 (Lewis *et al.*, 2004).

Les trois premières bases proviennent de la discrimination semi-supervisée (Chapelle *et al.*, 2006). L'ordonnement consiste à ordonner les exemples positifs au-dessus des exemples négatifs. La base TEXT est une base dérivée de 20-NEWSGROUP, qui regroupe des messages textes concernant des sujets en informatique. Il s'agit alors de retrouver ceux dans le groupe IBM. Les deux autres bases sont des bases d'images, USPS rassemble des chiffres manuscrites (2 et 5 sont pertinents) et COIL est un ensemble d'images de 100 objets différents pris sous différentes angles.

Nous avons aussi utilisé deux autres bases, SEGMENT du projet StatLog et RCV1. Les classes cible sont ici *brickface* et la classe CCAT. Pour RCV1, la représentation<sup>3</sup> TF-IDF a été calculée sur l'ensemble de la base d'apprentissage. Nous avons ensuite extrait une sous-base pour nos expériences. Le tableau 4.2 récapitule les caractéristiques des bases utilisées.

Dataset	$d$	$n$	$ \mathcal{S}_t $	$n + m$	$Card_t$
USPS	241	10	17,7	1 000	500
COIL	241	10	23	1 000	500
TEXT	11 960	10	21,8	1 000	200
TREC	44	10	9.1	800	200
SEGMENT	19	10	13.5	1 500	800
RCV1	47 236	20	92.4	5 020	9 980

## 4.3 Résultats

### 4.3.1 Comparaison entre RankSVM et Semi-CRank

Nous avons reporté les résultats obtenus pour l'AUC et la précision moyenne dans les tableaux 3 et 4 des expériences sur l'ensemble des bases considérées. Les résultats pour les précisions locales sont reportées dans les tableaux 5, 6 et 7. Le symbole  $\star$  indique le cas où un résultat est significativement meilleur au sens du test de Wilcoxon avec une confiance à 98%.

Pour les bases USPS, COIL et SEGMENT, nous observons une amélioration pour l'ensemble des mesures globales (AUC et précision moyenne). Pour la base USPS, l'amélioration est légère. Cependant, nous pouvons constater qu'avec l'erreur en AUC (càd  $1 - AUC$ ), nous obtenons une amélioration de 4.8%, qui est comparable avec le gain

<sup>1</sup>disponible sur <http://www.kyb.tuebingen.mpg.de/ssl-book/benchmarks.html>

<sup>2</sup>disponible sur <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

<sup>3</sup><http://leon.bottou.org/projects/sgd>

	<i>RankSVM</i>	<i>Semi-Crank</i>
USPS	77.3	<b>78.4*</b>
COIL	62.9	<b>65.5*</b>
SEGMENT	86.6	<b>92.2*</b>
TEXT	73.6	73.6
RCV-1	73.6	73.6

TAB. 3 – Résultats pour l’AUC : les résultats sont donnés en pourcentage

	<i>RankSVM</i>	<i>Semi-Crank</i>
USPS	53.2 (53.2)	<b>56.4*</b> (56.8)
COIL	61.8 (63.6)	<b>65.5*</b> (65.5)
SEGMENT	53.2 (53.8)	<b>71.2*</b> (71.2)
TEXT	74.2 (74.2)	74.2 (74.2)
RCV-1	74.2 (74.2)	74.2 (74.2)

TAB. 4 – Résultats des précisions moyennes. Entre parenthèses, la meilleure performance que l’on peut atteindre sur l’ensemble de test.

obtenu en discrimination transductive par *LapRLS*, une méthode à base de graphe. Cette méthode fait partie des références en discrimination semi-supervisée. Cette comparaison est néanmoins à relativiser puisque le cadre d’expérimentation et le critère utilisé (erreur en discrimination pour *LapRLS*) ne sont pas les mêmes. Nous obtenons toutefois des bonnes performances au niveau de la précision moyenne. Dans ce cas, nous pouvons conclure (a) que l’hypothèse de clustering utilisée est valide et (b) qu’apprendre une fonction de score en utilisant des données étiquetées et non-étiquetées permet d’améliorer les différentes mesures classiques utilisées en RI.

Pour les bases TEXT et RCV-1, les résultats montrent des performances identiques pour les deux méthodes. L’apport des exemples non étiquetés semble être négligeable dans ces cas. En regardant de plus près les solutions obtenues, nous avons remarqué qu’elles étaient proches de 0 pour l’ensemble des hyperparamètres. Pour ces vecteurs, l’influence des régulariseurs devient négligeables. Dans le cas de l’ordonnement, pour les moteurs de recherche, les auteurs de (Chapelle *et al.*, 2007) ont observé des résultats similaires (dans le cadre supervisé). Le problème serait lié au sous-apprentissage. Il semblerait que l’algorithme tente de minimiser la fonction perte et les régulariseurs en diminuant simplement la norme du vecteur. La grande dimension de ces problèmes est un vrai problème avec notre approche, puisque nous cherchons à partitionner les exemples dans ces espaces en utilisant la distance euclidienne qui dans ces cas perd de son sens.

## 5 Conclusion et perspectives

Nous avons présenté une méthode d’apprentissage semi-supervisée inductif de fonctions d’ordonnement dans le cas *bipartite*. Contrairement aux études transductives précédentes pour l’utilisation de données partiellement étiquetées pour l’apprentissage de fonctions de scores, notre méthode peut inférer un ordre sur une liste d’exemples qui n’ont pas été vus durant la phase d’apprentissage. Les études expérimentales ont montré des bons résultats et semblent indiquer que les données non-étiquetées peuvent

	RankSVM	Semi-Crank
USPS	76.0 (76.0)	<b>80.0*</b> (88.0)
COIL	52.0 (100)	<b>71.0*</b> (100)
SEGMENT	39.0 (49.0)	<b>77.0*</b> (77.0)
TEXT	88.0 (90.0)	88.0 (90.0)
RCV-1	100	100

TAB. 5 – Performances des  $prec@10$ 

	RankSVM	Semi-Crank
USPS	62.2 (62.2)	<b>64.0*</b> (64.4)
COIL	64.6 (73.8)	<b>69.2*</b> (73.8)
SEGMENT	52.4 (55.0)	<b>74.8*</b> (74.8)
TEXT	84.4 (85.2)	84.4 (85.2)
RCV-1	97.6 (97.6)	97.6 (97.6)

TAB. 6 – Performances des  $prec@50$ 

	RankSVM	Semi-Crank
USPS	49,9 (50,0)	50,4 (50,6)
COIL	67,1 (67,1)	72,4 (72,8)
SEGMENT	55,9 (56,4)	67,3 (67,3)
TEXT	80,6 (80,7)	80,6 (80,7)
RCV-1	96,2 (96,2)	96,2 (96,2)

TAB. 7 – Performances pour la  $prec@100$ 

être utilisées pour améliorer les performances. Cependant, pour les bases à grande dimension, les termes de régularisation n'ont pas d'influence sur les performances de la fonction de score. (Chapelle *et al.*, 2007) ont observé un phénomène identique pour les moteurs de recherche et a proposé un nouveau coût pour remédier à ce problème.

Pour notre algorithme, l'étape d'optimisation fait intervenir deux fois les paires d'exemples issues d'une partition, ce qui augmente considérablement le nombre de variables. La résolution exacte dans la forme duale est ainsi fortement limitée par le nombre d'exemples d'apprentissage. Nous pouvons noter qu'en discrimination semi-supervisée, la majorité des méthodes ne peuvent traiter un grand nombre d'exemples et que le passage à l'échelle pour *TSVM* où les méthodes à base de graphe est récent. Dans notre cas, notre méthode peut profiter des dernières études dans l'optimisation du SVM. Le principal facteur limitant reste en fait le calcul des poids, problème qui, à notre connaissance, n'a pas encore été abordé et constitue une direction majeure pour nos travaux futurs. Enfin notre étude s'est entièrement focalisée sur le nombre de paires mal-ordonnées. Or, dans beaucoup d'applications comme les moteurs de recherche, seuls les exemples les mieux ordonnés intéressent les utilisateurs. En apprentissage supervisé, des méthodes ont ainsi été proposées pour optimiser directement d'autres critères comme la précision moyenne ou  $prec@k$ . L'utilisation des exemples non-étiquetés pour ces critères d'optimisation constitue aussi une direction intéressante.

## REMERCIEMENTS

Ce travail a été partiellement financé par le programme IST de la Communauté Européenne, dans le cadre du réseau d'excellence PASCAL, IST-2002-506778

## Références

- AGARWAL A. & CHAKRABARTI S. (2007). Learning random walks to rank nodes in graphs. In *Proceedings of the 24th international conference on Machine learning (ICML)*, p. 9–16.
- AGARWAL S. (2006). Ranking on graph data. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, p. 25–32.
- AGARWAL S., GRAEPEL T., HERBRICH R., HAR-PELED S. & ROTH D. (2005). Generalization bounds for the area under the ROC curve. *Journal of Machine Learning Research*, **6**, 393–425.
- AGARWAL S. & NIYOGI P. (2005). Stability and generalization of bipartite ranking algorithms. In *Proceedings of the 18th Annual Conference on Learning Theory (COLT)*.
- AMINI M.-R. & GALLINARI P. (2003). Semi-supervised learning with explicit misclassification modeling. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*, p. 555–560.
- BELKIN M., NIYOGI P. & SINDHWANI V. (2006). Manifold regularization : A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning*, **7**, 2399–2434.
- BLUM A. & MITCHELL T. (1998). Combining labeled and unlabeled data with co-training. In *COLT : Proceedings of the Workshop on Computational Learning Theory*, p. 92–100.
- BORDES A., ERTEKIN S., WESTON J. & BOTTOU L. (2005). Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, **6**, 1579–1619.
- BOTTOU L. (2007). Svqp. <http://mloss.org/software/view/22/>.
- BOTTOU L. & BENGIO Y. (1994). Convergence properties of the k-means algorithms. In *Advances in Neural Information Processing Systems, 7 (NIPS)*, p. 585–592.
- CHAPELLE O., LE Q. & SMOLA A. (2007). Large margin optimization of ranking measures. In *NIPS'07 workshop on Machine Learning for Web Search*.
- O. CHAPELLE, B. SCHÖLKOPF & A. ZIEN, Eds. (2006). *Semi-Supervised Learning*. Cambridge, MA : MIT Press.
- CHAPELLE O. & ZIEN A. (2005). Semi-supervised classification by low density separation. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AI STATs'05)*.
- CHRISTOPHER D. MANNING, PRABHAKAR RAGHAVAN AND HINRICH SCHÜTZ (2008). *Introduction to Information Retrieval*. Cambridge University Press.
- CHU W. & GHAHRAMANI Z. (2005). Extensions of gaussian processes for ranking : semi-supervised and active learning. In *NIPS'05 Workshop on Learning to Rank (NIPS'05-LR)*, Whistler, Canada.

- CORTES C. & MOHRI M. (2004). Auc optimization vs. error rate minimization. In *Advances in Neural Information Processing Systems 16 (NIPS)*.
- FREUND Y., IYER R., SCHAPIRE R. E. & SINGER Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning*, **4**, 933–969.
- HERBRICH R., GRAEPEL T. & OBERMAYER K. (2000). Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, p. 115–132.
- IYER R. D., LEWIS D. D., SCHAPIRE R. E., SINGER Y. & SINGHAL A. (2000). Boosting for document routing. In *Proceedings of the ninth international conference on Information and knowledge management*, p. 70–77.
- LEBANON G. & LAFFERTY J. (2004). Hyperplane margin classifiers on the multinomial manifold. In *Proceedings of the twenty-first international conference on Machine learning (ICML)*.
- LEWIS D. D., YANG Y., ROSE T. G. & LI F. (2004). Rcv1 : A new benchmark collection for text categorization research. *Journal of Machine Learning*, **5**, 361–397.
- ROBERTSON S. E. & SOBOROFF I. (2001). The TREC 2001 filtering track report. In *Text REtrieval Conference*.
- SINZ F. (2007). Universvm. <http://mloss.org/software/view/19/>.
- SINZ F. H., CHAPELLE O., AGARWAL A. & SCHÖLKOPF B. (2008). An analysis of inference with the universum. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems (NIPS)*, p. 1–8.
- VISHWANATHAN S., MURTY M. N. & SMOLA A. J. (2003). SSVM : A simple SVM algorithm. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- WESTON J., COLLOBERT R., SINZ F., BOTTOU L. & VAPNIK V. (2006). Inference with the universum. In *23rd International Conference on Machine Learning (ICML)*, p. 127.
- WESTON J., KUANG R. L. C. & W.S. N. (2006). Protein ranking by semi-supervised network propagation. *BMC Bioinformatics*, **special issue**.
- X. ZHU AND Z. GHAHRAMANI AND J. LAFFERTY (2003). Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *Proceedings of International Conference on Machine Learning (ICML)*.
- ZHOU D., BOUSQUET O., LAL T. N., WESTON J. & SCHOLKOPF B. (2004a). Learning with local and global consistency. *Advances in Neural Information Processing Systems*, **16**, 321–328.
- ZHOU D., WESTON J., GRETTON A., BOUSQUET O. & SCHÖLKOPF B. (2004b). Ranking on data manifolds. In *Advances in Neural Information Processing Systems 16 (NIPS)*.
- ZHU X. (2007). Semi-supervised learning literature survey. <http://pages.cs.wisc.edu/~jerryzhu/research/ssl/semireview.html>.

## A Mise sous la forme duale

Dans cet annexe, nous donnons les détails du passage de la forme primale à la forme duale. Les notations introduites dans le papiers sont notamment plus explicites. Rappelons le problème d’optimisation :

$$\min_{\beta} \quad \frac{1}{2} \|\beta\|^2 + C \sum \xi_{ij} + C' \sum (\theta_{uv} + \theta_{uv}^*) \quad (11)$$

$$\text{sc} \quad \langle \beta, x_i - x_j \rangle \geq 1 - \xi_{ij} \text{ avec } (x_i, x_j) \in \overline{\mathcal{S}} \quad (12)$$

$$\omega_{uv} \langle \beta, x_u - x_v \rangle \geq -\epsilon - \theta_{uv} \text{ avec } (x_u, x_v) \in \overline{\mathcal{C}} \quad (13)$$

$$-\omega_{uv} \langle \beta, x_u - x_v \rangle \geq -\epsilon - \theta_{uv}^* \text{ avec } (x_u, x_v) \in \overline{\mathcal{C}} \quad (14)$$

$$\xi_{ij} > 0, \theta_{uv} > 0, \theta_{uv}^* > 0 \quad (15)$$

Les contraintes font ainsi intervenir : l'ensemble des paires cruciales de la base étiquetée ainsi que les paires provenant d'une partition. Nous pouvons remarquer que les paires d'exemples non-étiquetés interviennent deux fois ( dans les contraintes (13) et (14) ). Soit  $\mathcal{L}$  la matrice de données étiquetées telle que les vecteurs colonnes  $z_i$  représentent une paire d'instances cruciales. De même  $\mathbf{Z}_u$  la matrice constituée de vecteurs colonnes représentant les paires issues d'une partition pondérée par  $\omega_{uv}$ . Nous formons la matrice  $\mathbf{Z}$  comme ci dessus :

$$\mathbf{Z} = (\mathcal{L}, \mathbf{Z}_u, \mathbf{Z}_u)$$

Ainsi chaque vecteur colonne  $z_i$  représente une paire d'exemples. Soit  $N$ , le nombre de vecteurs colonnes de  $\mathbf{Z}$ . Nous associons de plus à chaque vecteur  $z_i$  une étiquette fictive  $y_i$  telle que pour toutes les paires cruciales l'étiquette est positive. Pour les paires d'instances non étiquetées, nous donnons l'étiquette  $-1$  si  $i < \dots, |\overline{\mathcal{S}}| + |\overline{\mathcal{C}}|$  et  $1$  sinon. Ainsi chaque paires d'un regroupement apparaît deux fois avec des étiquettes différentes.

Avec cette nouvelle notation, nous pouvons réécrire le problème d'optimisation en introduisant

$$\min_{\beta} \quad \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^{|\overline{\mathcal{S}}|} \xi_i + C' \sum_{i=1+|\overline{\mathcal{S}}|}^N (\xi_i) \quad (16)$$

$$\text{sc} \quad y_i \langle \beta, z_i \rangle \leq 1 - \xi_i \text{ pour } i = 1, \dots, |\overline{\mathcal{S}}| \quad (17)$$

$$-y_i \langle \beta, z_i \rangle \leq -\epsilon - \xi_i \text{ pour } i = |\overline{\mathcal{S}}|, \dots, N \quad (18)$$

$$\xi_i > 0 \quad (19)$$

La résolution peut passer par la méthode des multiplicateurs de Lagrange (variables duales) de façon similaire aux SVMs. Comme la solution est un point de col du lagrangien associé, les conditions de Kuhn-Tucker permettent d'obtenir des conditions sur les différentes variables. Ces conditions permettent d'éliminer les variables de la forme primale dans le lagrangien, qui peut alors se mettre sous la forme suivante :

$$\max_{\alpha_i} \quad \sum_{i=1}^N \kappa_i \alpha_i - \frac{1}{2} \sum_{i,j=1}^N y_i y_j \alpha_i \alpha_j \langle z_i, z_j \rangle$$

$$\text{sc} \quad 0 \leq \alpha_i \leq C \text{ pour } i = 1, \dots, |\overline{\mathcal{S}}|$$

$$0 \leq \alpha_i \leq C' \text{ pour } i = |\overline{\mathcal{S}}| + 1, \dots, N$$