Classification multi-tâches semi-supervisée en grande dimension

Victor LÉGER[1], Malik TIOMOKO[2], Romain COUILLET[1]

[1]Laboratoire d'Informatique de Grenoble
700 avenue Centrale, 38401 St Martin d'Hères, France

[2]Huawei Technologies France
18 quai du point du jour, Arcs de Seine, 92100 Boulogne-Billancourt, France

Victor.Leger@univ-grenoble-alpes.fr, Malik.Tiomoko@huawei.com
Romain.Couillet@univ-grenoble-alpes.fr

Résumé – Cet article propose un nouveau cadre pour l'apprentissage semi-supervisé multi-tâches. La méthode est justifiée par une analyse statistique de la performance sous l'hypothèse de données de grande dimension, et apporte une amélioration théorique permettant notamment d'éviter la limite classique du *transfert négatif*. Ainsi, le travail ouvre la voie à la création d'algorithmes efficaces et appuyés par la théorie qui apprennent à partir de jeux de données hétérogènes et peu ou mal étiquetés, ce qui constitue un problème à la fois ambitieux et fréquent en apprentissage statistique.

Abstract – This article proposes a novel framework for semi-supervised and multi-task learning. The method comes along with statistical performance analyses under a large dimensional data assumption, which provides a theoretical improvement, avoiding in particular the classical negative transfer limit. As such, the work opens the path toward the design of efficient and theoretically supported algorithms to learn from loosely labeled data and irregular datasets, both a challenging and frequently met problem in machine learning.

1 Introduction

On rencontre souvent, dans un contexte d'apprentissage machine, des données de grande dimension, dont l'ordre de grandeur n'est pas négligeable par rapport au nombre de données. Ainsi beaucoup de modèles d'apprentissage, fondés sur l'intuition que l'on se fait des espaces de petite dimension, échouent lorsque la dimension des données devient trop grande. La théorie des matrices aléatoires [1, 2] propose une solution à ce phénomène, qui est un exemple de la populaire *malédiction de la dimension*.

Des travaux ont d'ores et déjà été menés pour analyser à l'aide de ces outils statistiques le comportement de certains algorithmes de classification standards dans le cadre de données de grande dimension [3, 4]. Ces algorithmes ont ainsi été prouvés sous-optimaux dans leur utilisation naïve. Ces travaux ont notamment été menés sur deux contextes d'apprentissage bien connus : l'apprentissage semi-supervisé et l'apprentissage multi-tâches. Nous nous proposons ici de généraliser ces études au cadre pratique plus classique d'apprentissage semi-supervisé à tâches multiples, qui combine ces deux approches.

En effet, les méthodes récentes de classification reposent souvent sur l'existence de grandes quantités de données *étiquetées* : on parle de classification *supervisée*. Cependant, dans la plupart des applications réelles, si on a en effet accès à de grandes quantités de données, le processus d'étiquetage est quant à lui fastidieux et coûteux. Pour pallier ce problème, l'apprentissage dit *semi-supervisé* permet d'augmenter la base de donnée d'apprentissage avec les données non étiquetées. L'étude statistique en grande dimension de tels algorithmes a déjà été traitée [3], mais a pour l'instant été restreinte à une seule tâche.

Pour certaines applications réelles, il existe des jeux de données proches, mais non identiques (données médicales venant de plusieurs centres par exemple). S'ils sont trop différents, un algorithme utilisant naïvement le regroupement de ces jeux de données pourrait avoir de moins bonnes performances que s'il avait été appliqué à un jeu de données unique. C'est un écueil connu sous le nom de *transfert négatif* : les tâches additionnelles viennent parasiter la tâche originale. Les algorithmes d'apprentissage multi-tâches ont pour objectif d'intégrer les tâches additionnelles de manière à augmenter la taille de la base de données, sans pour autant dégrader l'apprentissage. De tels algorithmes ont déjà été analysés dans le cadre supervisé [4].

2 Modèle et hypothèses

Soit $\mathbf{X} \in \mathbb{R}^{p\times n}$ un ensemble de n vecteurs de données indépendantes de dimension p. Les données sont divisées en T sous-ensembles associés chacun à une "Tâche". Plus précisément, si $\mathbf{X} = [\mathbf{X}^1, \ldots, \mathbf{X}^T] \in \mathbb{R}^{p\times n}$, la Tâche t est une tâche de classification binaire semi-supervisée, dont la base d'entraînement $\mathbf{X}^t \in \mathbb{R}^{p\times n^t}$ est constituée d'un ensemble de

n_ℓ^t données étiquetées $\mathbf{X}_\ell^t = \{\mathbf{x}_i^t\}_{i=1}^{n_\ell^t}$ et d'un ensemble de n_u^t données non étiquetées $\mathbf{X}_u^t = \{\mathbf{x}_i^t\}_{i=n_\ell^t+1}^{n^t}$. À chaque vecteur de données $\mathbf{x}_i^t$ de la Tâche t est associé une étiquette y_i^t, et notre objectif est de prédire les étiquettes des données non étiquetées $\mathbf{X}_u^t$.

Hypothèse 1 (Sur la distribution des données) *Les colonnes de la matrice des données* $\mathbf{X}$ *sont des variables aléatoires gaussiennes indépendantes. Plus précisément, les échantillons de données* $(\mathbf{x}_1^t, \ldots, \mathbf{x}_{n^t}^t)$ *de la tâche* t *sont des observations i.i.d. telles que* $\mathbf{x}_i^t \in \mathcal{C}_j^t \Leftrightarrow \mathbf{x}_i^t \sim \mathcal{N}(\boldsymbol{\mu}_j^t, \mathbf{I}_p)$ *où* $\mathcal{C}_j^t$ *désigne la Classe* j *de la Tâche* t.

Cette condition, en apparence restrictive, est fondée sur le fait que notre algorithme est suffisamment robuste pour être efficace sur des données qui s'écarteraient d'une telle hypothèse. En effet, les travaux récents en théorie des matrices aléatoires [5, 6, 7] suggèrent que les méthodes de calcul utilisées dans cet article pour les vecteurs gaussiens peuvent être transposées à une classe de vecteurs aléatoires plus large (appelés vecteurs concentrés [8]), qui comprend notamment les vecteurs produits par des GANs (Generative Adversarial Networks), connus entre autres pour leur capacité à générer des images réalistes.

Comme expliqué en introduction nous faisons l'hypothèse que la dimension p des données n'est pas négligeable par rapport au nombre de données d'entraînement. Plus précisément :

Hypothèse 2 (Ratio asymptotique) *Lorsque* $n \to \infty$, *alors* $p/n_\ell \to c_\ell > 0$ *et* $p/n_u \to c_u > 0$. *De plus,* $n_{\ell j}^t/n_\ell \to \rho_{\ell j}^t > 0$, $n_{uj}^t/n_u \to \rho_{uj}^t > 0$, *avec* $n_{\ell j}^t$ *(resp.* n_{uj}^t*) le nombre de données étiquettées (resp. non étiquettées) de la Tâche* t *qui appartiennent à la Classe* j.

De manière similaire à [3], notre méthode est basée sur une approche par régularisation d'un graphe. L'idée générale est de propager les étiquettes associées aux données étiquetées aux données non étiquetées, en se basant sur le fait que deux points proches dans l'espace des données doivent se voir attribuer des étiquettes proches. Cette méthode peut s'exprimer sous la forme d'un problème d'optimisation sous contrainte :

$$\min_{\mathbf{f}^1,\ldots,\mathbf{f}^T} \sum_{t,t'=1}^{T} \Lambda^{tt'} \sum_{i=1}^{n^t} \sum_{i'=1}^{n^{t'}} \omega_{ii'}^{tt'} \left(f_i^t - f_{i'}^{t'} \right)^2$$
$$\textit{tel que } f_i^t = y_i^t \; \forall \, 1 \leq i \leq n_\ell^t \textit{ et } 1 \leq t \leq T.$$

Les poids $\omega_{ii'}^{tt'} = \frac{1}{Tp} \langle \mathbf{x}_i^t, \mathbf{x}_{i'}^{t'} \rangle$ quantifient la proximité entre deux points de données : plus cette quantité est grande, plus la distance $\left(f_i^t - f_{i'}^{t'} \right)^2$ entre les étiquettes associées doit être faible. Les hyperparamètres $\Lambda^{tt'}$ quantifient quant à eux à quel point les tâches t et t' doivent être liées. Nous choisissons de fixer :

$$\Lambda^{tt'} = \frac{|\langle \boldsymbol{\mu}_1^t - \boldsymbol{\mu}_2^t, \boldsymbol{\mu}_1^{t'} - \boldsymbol{\mu}_2^{t'} \rangle|}{\|\boldsymbol{\mu}_1^t - \boldsymbol{\mu}_2^t\| \|\boldsymbol{\mu}_1^{t'} - \boldsymbol{\mu}_2^{t'}\|}. \tag{1}$$

Ainsi $\Lambda^{tt'}$ est d'autant plus grand que les tâches sont corrélées. Cette quantité est estimable facilement, et ce choix est validé empiriquement, donnant d'aussi bons résultats qu'une optimisation sur l'espace $[0,1]^{k \times k}$.

On ajoute une contrainte de régularisation sur les scores des données non étiquetées, quantifiée par $\alpha \|\mathbf{f}_u\|^2$. Le problème est convexe à condition que $\alpha > \alpha_0$, α_0 étant une constante calculable. L'hyperparamètre α quantifie le recours au données non étiquetées : plus α est proche de la borne α_0, plus l'algorithme utilise les données non étiquetées. Inversement, pour $\alpha \to \infty$, notre méthode se comporte comme un algorithme supervisé.

La solution de ce problème d'optimisation convexe sous contrainte peut être obtenue explicitement, et s'exprime sous la forme matricielle suivante :

$$\mathbf{f}_u = \left(\mathbf{I}_{n_u} - \frac{\mathbf{Z}_u^\mathsf{T} \mathbf{A} \mathbf{Z}_u}{Tp} \right)^{-1} \frac{\mathbf{Z}_u^\mathsf{T} \mathbf{A} \mathbf{Z}_\ell}{Tp} \mathbf{y}_\ell \tag{2}$$

où

$$\mathbf{A} = \tilde{\mathbf{\Lambda}} \otimes \mathbf{I}_p \quad \textit{avec} \quad \tilde{\mathbf{\Lambda}} = \frac{\mathbf{\Lambda}}{\alpha} \tag{3}$$

$$\mathbf{Z}_\ell = \sum_{t=1}^{T} \mathbf{E}_{tt} \otimes \mathbf{X}_\ell^t \quad \text{et} \quad \mathbf{Z}_u = \sum_{t=1}^{T} \mathbf{E}_{tt} \otimes \mathbf{X}_u^t \tag{4}$$

avec $\mathbf{E}_{ij} \in \mathbb{R}^{T \times T}$ la matrice canonique $[\mathbf{E}_{ij}]_{ab} = \delta_{ia}\delta_{jb}$, et $\otimes$ le produit de Kronecker.

Dans cette expression en apparence complexe de $\mathbf{f}_u$, on retrouve principalement trois ingrédients :

- Les matrices de données $\mathbf{Z}_\ell$ et $\mathbf{Z}_u$, qui rassemblent respectivement les données étiquetées et non étiquetées;
- La matrice des hyperparamètres $\mathbf{A}$, qui contient à la fois les paramètres de corrélation entre les tâches $\mathbf{\Lambda}$ et le paramètre qui quantifie le recours aux données non étiquetées α;
- Le vecteur des étiquettes $\mathbf{y}_\ell$.

Dans la plupart des modèles de classification binaire, les étiquettes de chaque vecteur $\mathbf{x}$ prennent les valeurs ± 1 selon que $\mathbf{x}$ appartient à une classe ou à l'autre. Cependant, nous décidons ici de laisser flottante la valeur de ces étiquettes. Nous montrerons dans la Section 3 la pertinence de ce choix apparemment "contre-nature". La seule contrainte que l'on impose est que toutes les étiquettes associées à une classe donnée aient la même valeur. Autrement dit, pour chaque $\mathbf{x}_i^t \in \mathcal{C}_j^t$, $y_i^t = \tilde{y}_j^t$. Ainsi, les valeurs attribuées aux étiquettes peuvent être résumées dans le vecteur $\tilde{\mathbf{y}} = [\tilde{y}_1^1, \tilde{y}_2^1, \ldots, \tilde{y}_2^T]^\mathsf{T} \in \mathbb{R}^{2T}$.

3 Résultats principaux

Le cœur de notre méthode réside dans le fait que l'on peut prédire, à partir des hypothèses 1 et 2, le comportement asymptotique de la fonction de décision. Cette fonction joue un rôle central dans le processus de classification, et prédire sa loi nous permet de quantifier la probabilité d'erreur de notre algorithme. On a ainsi les clés pour optimiser les hyperparamètres $\mathbf{\Lambda}$ et α

ainsi que les étiquettes $\tilde{\mathbf{y}}$ afin de minimiser la probabilité d'erreur.

Théoreme 1 *Sous les hypothèses 1 et 2, pour tout* $\mathbf{x} \in \mathcal{C}_j^t$ *non étiqueté,* f *étant le score associé,*

$$f \to \mathcal{N}\left(m_j^t, {\sigma^t}^2\right), \qquad \textit{avec}$$
$$m_j^t = {\mathbf{a}_j^t}^\mathsf{T}\tilde{\mathbf{y}} \qquad \textit{et} \qquad {\sigma^t}^2 = \tilde{\mathbf{y}}^\mathsf{T}\mathbf{B}^t\tilde{\mathbf{y}},$$

où $\mathbf{a}_j^t \in \mathbb{R}^{2T}$ *et* $\mathbf{B}^t \in \mathbb{R}^{2T\times 2T}$ *sont fonctions des paramètres déterministes du modèle.*

L'essentiel du message derrière ce théorème est que la fonction de décision f suit asymptotiquement une loi normale, dont on connaît les paramètres. Ces paramètres dépendent de deux choses :

- Le vecteur $\mathbf{a}_j^t$ et la matrice $\mathbf{B}^t$. Ces quantités, de petites dimensions, ne dépendent que des données déterministes du problème (connues ou estimables) et des hyperparamètres choisis.
- Le vecteur des étiquettes $\tilde{\mathbf{y}}$, que l'on peut dès lors optimiser afin de réduire la probabilité d'erreur.

Commençons tout d'abord par expliciter cette probabilité d'erreur que l'on souhaite minimiser.

Définition 1 *Pour une Tâche cible* t *donnée, la probabilité de mal classifier chaque élément non étiqueté* $\mathbf{x}$ *est :*

$$\epsilon^t = \frac{n_{u1}^t}{n_u^t}\mathbb{P}(\mathbf{x} \to \mathcal{C}_2^t | \mathbf{x} \in \mathcal{C}_1^t) + \frac{n_{u2}^t}{n_u^t}\mathbb{P}(\mathbf{x} \to \mathcal{C}_1^t | \mathbf{x} \in \mathcal{C}_2^t). \quad (5)$$

Proposition 1 *Pour une Tâche cible* t *donnée, il existe un unique (à une contante multiplicative près) vecteur de score* $\tilde{\mathbf{y}}^\star$ *qui minimise la probabilité d'erreur, donné par:*

$$\tilde{\mathbf{y}}^\star = \mathbf{B}^{t^{-1}}\mathbf{a}_j^t. \quad (6)$$

Proposition 2 *Pour une tâche cible* t *donnée, la probabilité d'erreur minimale (en supposant l'équiprobabilité a priori d'appartenir à chacune des classes, et avec les étiquettes optimales* $\tilde{\mathbf{y}}^\star$*) est asymptotiquement égale à :*

$$\epsilon_\star^t = \mathcal{Q}\left(\frac{1}{2}\sqrt{{\mathbf{a}_j^t}^\mathsf{T}\mathbf{B}^{t^{-1}}\mathbf{a}_j^t}\right), \quad (7)$$

où $\mathcal{Q}(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{u^2}{2}}\,\mathrm{d}u$.

Si les étiquettes optimales $\tilde{\mathbf{y}}^\star$ sont données explicitement par l'équation (6), ce n'est pas le cas de l'hyperparamètre α. Cependant, comme nous disposons d'une formule explicite donnant la probabilité d'erreur en fonction des paramètres du problème (et en particulier en fonction de α), nous pouvons optimiser α en procédant à une recherche de minimum à une dimension. L'un des grands avantages de notre méthode est que l'on peut optimiser les hyperparamètres sans recourir à une validation croisée, coûteuse en temps de calcul et en nombre de données.

Tous les résultats et remarques précédentes peuvent être résumés dans l'algorithme 1.

algorithme 1 Algorithme optimal

Entrée: Données étiquetées $\mathbf{X}_\ell = [\mathbf{X}_\ell^1, \ldots, \mathbf{X}_\ell^T]$ et données non étiquetées $\mathbf{X}_u = [\mathbf{X}_u^1, \ldots, \mathbf{X}_u^T]$.

Sortie: Classe estimée $\hat{j} \in \{1,2\}$ de chaque vecteur non étiqueté de la tâche cible.

Construire les matrices de données $\mathbf{Z}_\ell$ et $\mathbf{Z}_u$ à partir de $\mathbf{X}_\ell^t$ et $\mathbf{X}_u^t$.

Créer $\mathbf{\Lambda}$ d'après (1) et les scores $\tilde{\mathbf{y}}^\star$ à partir de (6).

Estimer la probabilité d'erreur $\epsilon_\star^t$ selon (7) et optimiser par rapport à α en utilisant une recherche linéaire.

Calculer les scores de classification $\mathbf{f}_u$ selon (2).

Sortie: $\hat{j}$ tel que $f_i \underset{\hat{j}=1}{\overset{\hat{j}=2}{\gtrless}} \frac{m_1^t + m_2^t}{2}$.

4 Simulations

Par souci de simplicité, les résultats de cette section sont obtenus dans le cadre $T = 2$ où l'on a une tâche cible (que l'on souhaite réaliser) et une tâche source (qui nous aide à réaliser la tâche cible). Nous nous restreignons tout d'abord au cas où les données sont générées en simulant des variables aléatoires suivant l'hypothèse 1, afin de montrer l'optimalité de notre méthode dans ce cas-ci. Nous présentons ensuite les performances sur des jeux de données réelles ne suivant pas de distribution gaussienne isotrope, afin de montrer que notre méthode se généralise bien à des distributions plus exotiques.

4.1 Données synthétiques

Les tâches cible et source sont des mélanges de deux gaussiennes associées aux deux classes : $\mathcal{N}(\pm\boldsymbol{\mu}, \mathbf{I}_p)$ pour la source et $\mathcal{N}(\pm\beta\boldsymbol{\mu} + \sqrt{1-\beta^2}\boldsymbol{\mu}^\perp, \mathbf{I}_p)$ pour la cible, où $\boldsymbol{\mu}^\perp$ est un vecteur orthogonal au vecteur $\boldsymbol{\mu}$. Dans cette configuration, on peut contrôler la similarité entre les tâches à travers β. En particulier, pour $\beta = 0$ les tâches sont complètement décorrélées, tandis que pour $\beta = 1$ elles sont identiques. Notons que $\beta = -1$ correspond à un scénario où les classes des deux tâches sont inversées. La Figure 4.1 présente l'évolution du comportement de notre algorithme en fonction de β. Le premier graphe montre l'évolution de la valeur des étiquettes optimisées, tandis que le second graphe montre l'évolution de la probabilité d'erreur pour les différentes versions de l'algorithme :

- Version naive avec $\tilde{y}_j^t = (-1)^j$, $\mathbf{\Lambda} = \mathbb{1}_k\mathbb{1}_k^\mathsf{T}$;
- Version optimale, avec $\tilde{\mathbf{y}} = \tilde{\mathbf{y}}^\star$ et $\mathbf{\Lambda}$ d'après (1);
- Nous ajoutons également une borne théorique établie par la théorie de l'information, qui nous donne l'erreur minimale atteignable par n'importe quel algorithme étant donnés les paramètres de notre expérience (nombre de données n_j^t dans chaque classe, dimension p des données, éloignement entre les classes $\|\boldsymbol{\mu}_1^t - \boldsymbol{\mu}_2^t\|$).

Remarquons tout d'abord que notre algorithme optimal obtient des performances très proches de l'optimum donné par

la théorie de l'information, suggérant l'optimalité de notre modèle. Par ailleurs, la prédiction théorique de la probabilité d'erreur suit bien l'erreur constatée effectivement sur les données non étiquetées.

La symétrie de la courbe de performance de notre algorithme nous indique que notre méthode est parfaitement robuste au transfert négatif. Lorsque la corrélation entre les tâches β devient négative, les performances de l'algorithme naïf s'effondrent, à l'inverse de notre algorithme qui bénéficie toujours de l'ajout d'information. Nous pouvons mieux comprendre ce comportement en suivant l'évolution des étiquettes optimisées en fonction de β : plus la tâche source est corrélée à la tâche cible, plus l'amplitude des étiquettes $\tilde{y}_1^{(2)}$ et $\tilde{y}_2^{(2)}$ associées à la tâche source est grande. Lorsque β devient négatif, on observe une inversion du signe de ces étiquettes : l'algorithme comprend qu'il faut intervertir les deux classes pour mieux ressembler à la tâche cible.

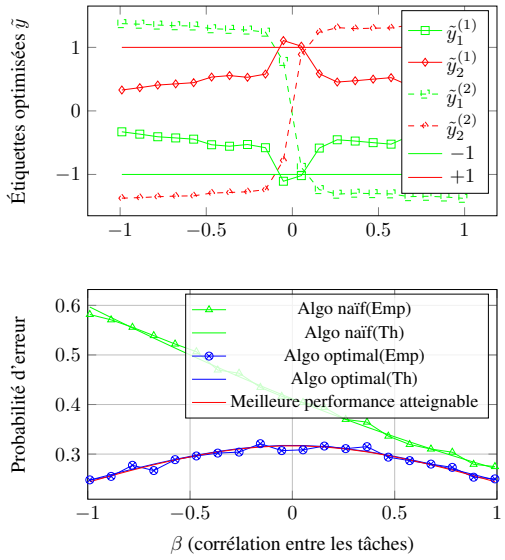

FIG. 1: Évolution de la probabilité d'erreur et des étiquettes en fonction de la corrélation entre les tâches. Les étiquettes optimisées s'adaptent d'elles-mêmes pour bénéficier au maximum de la tâche source, tandis que les étiquettes naïves, fixées, induisent un *transfert négatif* lorsque les tâches diffèrent trop.

4.2 Données réelles

Nous utilisons dans cette section deux jeux de données réelles :

- Le dataset MNIST [9], constitué d'images de chiffres manuscrits.
- Le dataset Multi Domain Sentiment [10], constitué d'avis déposés sur le site web *amazon.com* pour 4 catégories de produits : *Books*, *DVD*, *Electronics*, et *Kitchen*. La classification consiste à distinguer les avis posititfs (plus de 3 étoiles) des avis négatifs (3 étoiles ou moins).

Pour les 3 jeux de données, la probabilité d'erreur diminue lorsque l'on rajoute des données non étiquetées. Ainsi notre travail se généralise bien à des jeux de données correspondant à des applications réelles de classification.

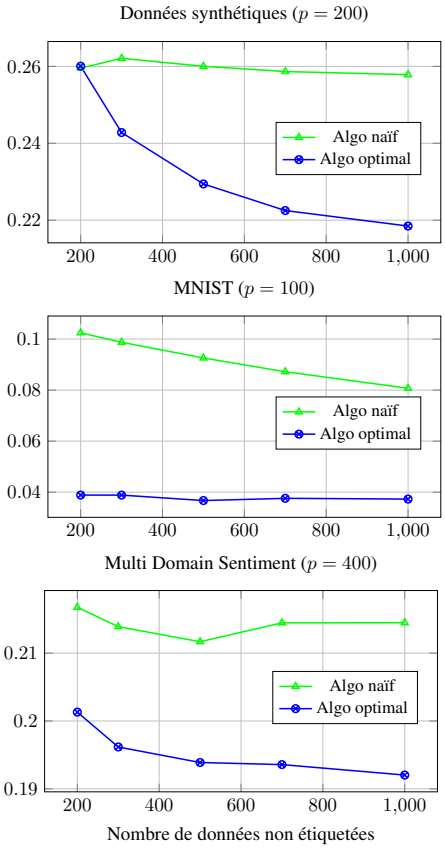

FIG. 2: Probabilité d'erreur en fonction du nombre de données non étiquetées dans la tâche cible. $n_{\ell 1}^1 = n_{\ell 2}^1 = 50, n_{u1}^2 = n_{u2}^2 = 0$ and $n_{\ell 1}^2 = n_{\ell 2}^2 = 100$ **(Haut)** Données gaussiennes avec une corrélation entre les tâches $\beta = 0.7$, $p = 200$ **(Milieu)** MNIST Dataset avec les chiffres $(1,4)$ comme cible et $(7,9)$ comme source. Une PCA est réalisée pour extraire les $p = 100$ composantes principales. **(Bas)** Multi Domain Sentiment Dataset avec *Book* comme source et *Kitchen* comme cible.

5 Conclusion

L'algorithme que nous avons proposé et analysé est simple et puissant, et permet de tirer profit au maximum à la fois de l'ajout de données non étiquetées et de l'ajout de tâches semblables, sans tomber dans l'écueil du *transfert négatif*. L'utilisation des outils issus de la théorie des matrices aléatoires nous a permis d'ajuster les hyperparamètres sans recourir à une validation croisée. Ces travaux ouvrent la voie à la création d'algorithmes peu coûteux, efficaces et flexibles.

Références

[1] E. Wigner *On the Distribution of the Roots of Certain Symmetric Matrices* Annals of Mathematics, 1958.

[2] V.A. Marčenko et L. Pastur *Distribution of eigenvalues for some sets of random matrices* Math USSR Sb, 1967.

[3] X. Mai et R. Couillet *A random matrix analysis and improvement of semi-supervised learning for large dimensional data* The Journal of Machine Learning Research, 2018.

[4] M. Tiomoko, R. Couillet et H. Tiomoko *Large Dimensional Analysis and Improvement of Multi Task Learning* arXiv preprint arXiv:2009.01591, 2020.

[5] M. Seddik, C. Louart, R. Couillet et M. Tamaazousti *The Unexpected Deterministic and Universal Behavior of Large Softmax Classifiers* International Conference on Artificial Intelligence and Statistics, 2021.

[6] M. Seddik, C. Louart, M. Tamaazouti et R. Couillet *Random Matrix Theory Proves that Deep Learning Representations of GAN-data Behave as Gaussian Mixtures* arXiv preprint arXiv:2001.08370, 2020.

[7] C. Louart, Z. Liao, R. Couillet et autres *A random matrix approach to neural networks* The Annals of Applied Probability, 2018.

[8] M. Ledoux *The concentration of measure phenomenon* Mathematical surveys and monographs, 2001.

[9] L. Deng *The MNIST database of handwritten digit images for machine learning research [best of the web]* IEEE Signal Processing Magazine, 2012.

[10] J. Blitzer, M. Dredze et F. Pereira *Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification* Proceedings of the 45th annual meeting of the association of computational linguistics, 2007.