RECITAL 2009, Senlis, 24–26 juin 2009

Apprentissage automatique et Co-training

Pierre Gotab
LIA / Université d'Avignon, 339 chemin des Meinajariès, 84911 Avignon
pierre.gotab@univ-avignon.fr

Résumé. Dans le domaine de la classification supervisée et semi-supervisée, cet article présente un contexte favorable à l'application de méthodes statistiques de classification. Il montre l'application d'une stratégie alternative dans le cas où les données d'apprentissage sont insuffisantes, mais où de nombreuses données non étiquetées sont à notre disposition : le co-training multi-classifieurs. Les deux vues indépendantes habituelles du co-training sont remplacées par deux classifieurs basés sur des techniques de classification différentes : icsiboost sur le boosting et LIBLINEAR sur de la régression logistique.

Abstract. In the domain of supervised and semi-supervised classification, this paper describes an experimental context suitable with statistical classification. It shows an alternative method usable when learning data is unsufficient but when many unlabeled data is avaliable : the multi-classifier co-training. Two classifiers based on different classification methods replace the two independent views of the original co-training algorithm : icsiboost based on boosting and LIBLINEAR which is a logistic regression classifier.

Mots-clés : Apprentissage automatique, classification, co-training.

Keywords: Machine learning, classification, co-training.

1 Classification et apprentissage supervisé

Les problèmes de classification présentés ici sont relatifs à l'apprentissage automatique supervisé. Il s'agit de construire un modèle représentatif d'un certain nombre de données organisées en classes - ensemble que l'on appelle généralement le *corpus d'apprentissage* - puis d'utiliser ce modèle afin de classer de nouvelles données, c'est à dire de prédire leur classe au vu de leurs caractéristiques (appelées *paramètres* ou *features*). La construction du modèle relève de l'apprentissage automatique supervisé, l'ensemble des *exemples* constituant le corpus d'apprentissage étant *annotés*, c'est à dire qu'ils portent le *label* de leur classe donné a priori.

Le processus permettant d'obtenir ce corpus d'apprentissage est généralement d'utiliser des annotateurs manuels, qui vont observer les exemples et, selon leur appréciation, leur attribuer tel label ou tel autre.

Un des problèmes majeurs inhérent à la classification supervisée en Traitement Automatique de la Langue Naturelle (TALN) est qu'obtenir un corpus d'apprentissage annoté manuellement est assez difficile, cela coute cher et n'est pas rapide. Ces corpus sont donc souvent disponibles en quantité limitée. Or la qualité des modèles des classifieurs dépend directement de la taille de ces corpus.

2 Application à la campagne DEFT

2.1 Présentation

DEFT (DÉfi Fouille de Textes) est une campagne d'évaluation ayant lieu chaque année depuis 2005[1]. Elle porte sur la fouille de textes en langue française.

Pour l'année 2008, elle traitait de la classification automatique de textes, et plus particulièrement de la détection du genre et du thème d'articles provenant du journal Le Monde et de l'encyclopédie Wikipedia.

L'intérêt de ce corpus est qu'il rend compte d'un problème de classification réel, dont les données ont été annotées par des humains. En effet, les classes ont été définies par Le Monde et Wikipedia et choisies par les auteurs des articles. Il se prête donc à des expériences à la frontière entre apprentissage théorique et linguistique appliquée.

L'inconvénient induit est qu'il existe sans doute des erreurs d'annotation, ou, plus souvent, de l'ambiguïté et des recouvrements entre les classes. Des articles peuvent traiter de plusieurs thèmes mais ne sont rangés que dans un seul (un article de littérature scientifique sera rangé dans Littérature ou dans Sciences ?), et des thèmes peuvent être fortement imbriqués (Sport et télévision par exemple). On suppose tout de même qu'il existe un lien fort entre le contenu d'un article et son thème principal, et c'est ce qui permettra de construire des modèles pour les classifier.

Le corpus fourni pour l'apprentissage est assez important, de l'ordre d'une dizaine de milliers d'exemples, c'est donc un contexte très favorable à la classification automatique supervisée à l'aide de méthodes statistiques.

2.2 Données

Le corpus est divisé en deux tâches.

La première tâche est un ensemble de 15223 articles en texte brut. Chaque article est associé à une classe parmi : art (ART), économie (ECO), sport (SPO), télévision (TEL) réparties ainsi :

ART	ECO	SPO	TEL
30.41%	8.88%	37.88%	22.82%

La partition de test, quant à elle, contient 10596 articles.

La deuxième tâche est un ensemble de 23550 articles associés aux classes : france (FRA), international (INT), littérature (LIV), sciences (SCI), société (SOC) réparties ainsi :

FRA	INT	LIV	SCI	SOC
14.12%	22.52%	19.43%	27.87%	16.04%

La partition de test contient 15693 articles.

[1] http ://deft.limsi.fr/

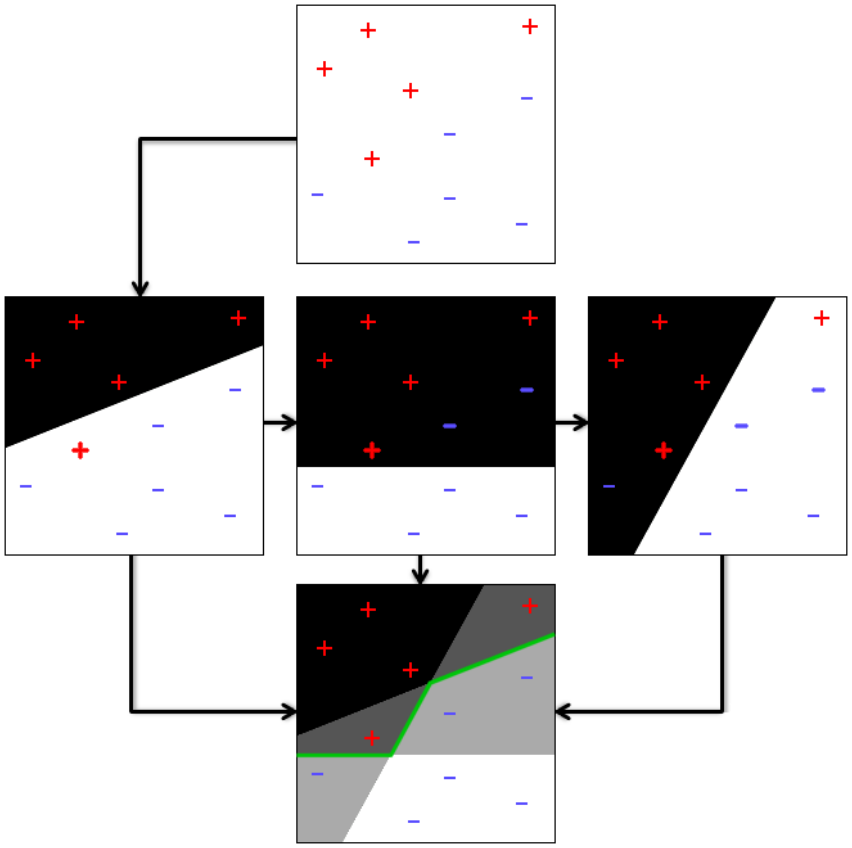
2.3 Protocole expérimental

Deux classifieurs sont utilisés pour réaliser cette tâche de classification.

Le premier est icsiboost [2] , une version open-source du classifieur BoosTexter, basé sur AdaBoost (Freund & Schapire, 1996), un algorithme de boosting. Icsiboost à l'avantage d'être facile à mettre en oeuvre, et possède des facilités telles que la génération de N-gram pour le texte. BoosTexter a été spécifiquement conçu pour la classification de textes (Schapire & Singer, 2000), icsiboost convient donc parfaitement à cette tâche.

L'algorithme AdaBoost consiste à construire une multitude d'*apprenants faibles*, qui combinés formeront l'*apprenant fort*, qui se résume à un vote pondéré des apprenants faibles. Il fonctionne de manière itérative. A chaque itération, l'apprenant faible qui minimise le nombre d'erreurs de classification est choisi. Les exemples qu'il aura mal classés auront un poids plus fort lors de l'itération suivante, afin que le prochain apprenant faible se concentre sur ces exemples difficiles à classer. Et ainsi de suite.

Voici un exemple naif illustré, avec deux classes (positive et négative) et deux paramètres (abscisse et ordonnée). Les apprenants faibles sont de simples fonctions linéaires qui séparent l'espace en deux :



Dans notre expérience sur DEFT, icsiboost est utilisé en apprentissage sur le sac de mots (c'est à dire l'ensemble des mots d'un exemple, indépendemment de leur nombre d'occurences), le nombre d'itérations de boosting est fixé à 1000.

[2] icsiboost, an opensource implementation of BoosTexter - http ://code.google.com/p/icsiboost

Le second est LIBLINEAR (Lin *et al.*, 2007), semblable à libsvm, mais basé sur de la régression logistique et des SVM à noyau linéaire (Vapnik, 2000). Son utilisation est plus austère mais il est très rapide. Il est particulièrement indiqué dans la classification de documents comme le montre l'appendice B.2 du guide de LIBSVM (Hsu *et al.*, 2003).

Le principe est assez proche de la régression linéaire à ceci près qu'on utilise une fonction de la forme $p(X) = \frac{e^{\alpha+\beta.X}}{1+e^{\alpha+\beta.X}}$ où α et β sont des constantes déterminées par la méthode du *maximum de vraissemblance* afin de minimiser l'erreur de classification. L'avantage est, au contraire de la régression linéaire, de pouvoir obtenir une probabilité (comprise dans $[0..1]$) associée à un exemple donné.

Lorsque l'on a plus d'un paramètre, β et X deviennent des vecteurs dont chaque dimension est associée à un paramètre.

Dans notre cas, le vecteur de liblinear pour un exemple donné est un vecteur à composantes binaires dont chaque dimension représente un mot du lexique, et sa valeur est 1 si le mot est dans le sac de mots, 0 sinon.

2.4 Résultats obtenus

Sur la première tâche :

Classifieur	icsiboost			LIBLINEAR		
Classe	**Précision**	**Rappel**	**F-mesure**	**Précision**	**Rappel**	**F-mesure**
ART	81.7%	90.4%	85.8%	84.7%	90.2%	87.4%
ECO	84.3%	91.7%	87.9%	82.6%	96.2%	88.9%
SPO	95.2%	89.9%	92.5%	96.8%	91.6%	94.1%
TEL	83.5%	49.3%	62.0%	90.3%	48.0%	62.7%
Toutes classes confondues			85.4%			86.9%

Sur la deuxième tâche :

Classifieur	icsiboost			LIBLINEAR		
Classe	**Précision**	**Rappel**	**F-mesure**	**Précision**	**Rappel**	**F-mesure**
FRA	79.9%	76.1%	78.0%	84.1%	78.9%	81.4%
INT	89.2%	89.9%	89.6%	90.9%	93.2%	92.0%
LIV	92.3%	89.7%	91.0%	92.5%	93.4%	92.9%
SCI	84.6%	89.5%	87.0%	89.1%	89.8%	89.5%
SOC	67.2%	64.9%	66.1%	72.0%	71.6%	71.8%
Toutes classes confondues			83.8%			86.8%

Lors de la campagne DEFT'08 les deux meilleurs systèmes ont obtenu un f-score de 87.8% sur la première tâche (Trinh *et al.*, 2008) et de 87.9% sur la deuxième (Charton *et al.*, 2008).

En comparaison, ces résultats montrent que l'on obtient facilement des scores très satisfaisants avec des méthodes statistiques, et sans prétraitements, dès lors que l'on a à disposition une quantité suffisante de données d'apprentissage.

Il est à noter que le classifieur icsiboost a été utilisé dans la fusion avec prétraitements de l'équipe jeunes chercheurs du Laboratoire Informatique d'Avignon qui a remporté la première place sur la deuxième tâche (Charton *et al.*, 2008).

Mais lorsque nous n'avons pas accès à une quantité suffisante de données d'apprentissage, quelles sont les alternatives ?

La faible quantité de corpus d'apprentissage peut-être contrebalancée en en augmentant la qualité, c'est le but de l'*active-learning* (Riccardi & Hakkani-Tur, 2005). Il consiste à sélectionner préalablement les exemples à annoter manuellement afin d'avoir un corpus peu redondant qui couvre un maximum de cas de figure en éliminant les exemples trop similaires.

Deux autres méthodes sont envisageables dans le cas où l'on a accès à de grandes quantités de données non annotées :
– L'*online-learning* permet à un système de s'améliorer alors qu'il est en production, c'est un domaine de recherche récent et assez prospectif.
– Et le *co-training* (Blum & Mitchell, 1998), qui permet d'augmenter artificiellement le corpus d'apprentissage, en lui adjoignant les données non annotées, après leur avoir attribué un label. Il se rapproche du *self-training* à ceci près qu'il met en jeu plusieurs classifieurs. C'est cette technique qui sera utilisée dans la suite de cet article.

3 Co-training

3.1 Présentation

Le co-training consiste à entraîner plusieurs classifieurs, chacun basé sur une vue du corpus, puis à les améliorer entre eux à l'aide d'une masse importante de données non annotées ; les classifieurs plus à même de classer un exemple donné jouant le rôle de "professeurs" pour les autres.

L'algorithme original présenté par Blum et Mitchell (Blum & Mitchell, 1998) a subi toutes sortes de modifications et d'adaptations à travers la littérature parue depuis :

– Utilisation d'un ensemble d'exemples commun aux classifieurs (Sarkar, 2001) ou bien un ensemble distinct par classifieur (Hwa *et al.*, 2003) (Müller *et al.*, 2001)
– Dans le même ordre d'idées, séparer le corpus dès la première itération (Müller *et al.*, 2001) ou bien laisser les classifieurs apprendre sur le même corpus de base.
– Utiliser un *pool* d'exemples non annotés (comme défini dans (Blum & Mitchell, 1998)) géré de plusieurs façons :
  – Aucun pool, le non annoté est traité d'un seul bloc (Guz *et al.*, 2007)
  – Un pool de taille fixe réapprovisionné de façon aléatoire (Pierce & Cardie, 2001)
– Obliger les classifieurs à classer un certain nombre d'exemples à chaque itération (Denis *et al.*, 2002)
– Respecter la distribution des classes a priori en ne retenant que les $k * f_c$ exemples les mieux classés de la classe c de probabilité a priori f_c (k est une constante empirique)

Dans notre cas, les deux classifieurs commencent leur apprentissage sur le même corpus (A), puis se constituent un corpus personnel (G_i) qui grossira à mesure qu'il sera approvisionné par l'autre classifieur.

Le pool d'exemples non annotés (U) est séparé en autant de partitions (U_i) que l'on souhaite d'itérations de co-training, et à chaque itération i, on soumet la partition U_i aux deux classifieurs. Pour chaque exemple, si un classifieur annonce un score de confiance supérieur à un certain $seuil$, il y apposera son label et ajoutera cet exemple au corpus d'apprentissage de l'autre classifieur.

Lorsqu'il est difficile de définir plusieurs vues décorrélées sur un ensemble de données, comme dans le cas de l'expérience précédente, il est possible d'utiliser plusieurs classifieurs différents basés sur l'ensemble des paramètres P (ici, le sac de mots). Et c'est la différence de méthode de classification qui va introduire la complémentarité nécessaire au co-training.

3.2 Algorithme

Soit un ensemble de données d'apprentissage A, de données de test T et de données non étiquetées U ; et un ensemble de paramètres (features) P. Un exemple $e \in A$ est un couple $e = (v, l)$ où v est un vecteur de dimension $|P|$ représentant les différentes valeurs de chaque paramètre, et l est le label (la classe) de cet exemple.

On découpe U en N partitions de taille égale notées $U_1, U_2, ...U_N$.

On entraîne deux classifieurs C_1 et C_2 sur A. Un exemple $e = (v, l)$ classé par un classifieur C_i donne $C_i(e) = (l', s)$ où l' est la classe attribuée par C_i avec un score de confiance $s \in [0..1]$.

0n définit un seuil $seuil_i \in [0..1]$ de confiance minimale pour chaque classifieur C_i.

On constitue deux ensembles $G_1 = G_2 = A$.

- Pour k allant de 1 à N
  - Chaque exemple $e = (v, l) \in U_k$ est classé par C_1 et C_2 :
    $C_1(e) = (l_1, s_1)$ et $C_2(e) = (l_2, s_2)$
    - Si $s_1 > seuil_1$, $G_2 := G_2 \cup \{(v, l_1)\}$
    - Si $s_2 > seuil_2$, $G_1 := G_1 \cup \{(v, l_2)\}$
  - On entraîne C_i sur G_i
  - On teste C_i sur T

4 Co-training multi-classifieurs appliqué à DEFT

4.1 Protocole expérimental

Pour simuler une pénurie de données annotées nous allons circonscrire les données d'apprentissage à seulement quelques milliers d'exemples.

Nous retirons les labels du reste des données d'apprentissage, qui sera alors considéré comme la partition non annotée (U).

Nous utilisons toujours icsiboost et LIBLINEAR comme classifieurs.

L'apprentissage et l'évaluation sont réalisés avec la configuration suivante :
– A contient quelques exemples, U contient le reste de l'apprentissage et T est la partition de test originale
– 10 étapes de co-training ($N = 10$)
– 1000 itérations d'icsiboost pour l'apprentissage des modèles
– Le seuil de confiance d'icsiboost est fixé à 0.78 et celui de LIBLINEAR à 0.98

Les seuils de confiance sont déterminés afin d'avoir 99% des exemples annotés automatiquement avec la bonne étiquette. Ils sont calculés à partir de la partition de test.

Les deux classifieurs servent de témoins dans l'évaluation, en comparant leurs performances avant et après le co-training.

4.2 Résultats obtenus

Voici deux résultats en détails, avec entre 85 et 90% du corpus d'apprentissage considéré comme non annoté :

Résultats *avant*, avec 2000 articles en apprentissage, et *après* co-training en tirant partie des 13223 autres articles de la première tâche :

Classifieur	icsiboost		LIBLINEAR	
Classe	**avant**	**après**	**avant**	**après**
ART	81.4%	82.6%	81.2%	82.3%
ECO	83.5%	84.7%	84.7%	85.9%
SPO	90.2%	90.9%	88.6%	90.4%
TEL	50.9%	53.2%	40.3%	39.3%
Toutes	81.1%	82.2%	80.5%	81.7%

On a une progression de la f-mesure de icsiboost de 1.1 points, et 1.2 points pour LIBLINEAR.

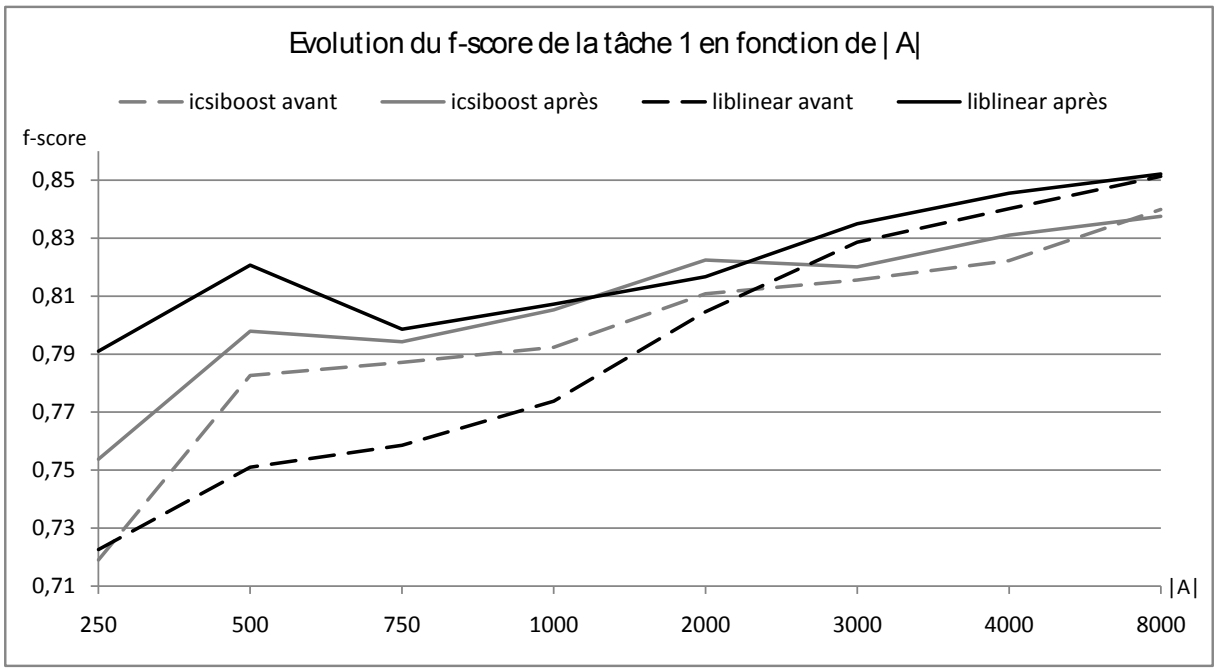
De même sur la deuxième tâche, 3000 articles en apprentissage et 20550 articles considérés comme non annotés :

Classifieur	icsiboost		LIBLINEAR	
Classe	**avant**	**après**	**avant**	**après**
FRA	73.7%	75.7%	76.9%	77.3%
INT	85.6%	85.5%	88.0%	88.3%
LIV	88.4%	88.3%	89.1%	89.4%
SCI	82.6%	84.1%	84.9%	85.4%
SOC	58.1%	59.6%	61.6%	61.8%
Toutes	79.4%	80.2%	81.9%	82.4%

On note un gain pour quasiment toutes les classes (à une exception près), la f-mesure d'icsiboost progresse de 0.8 points, et celle de LIBLINEAR de 0.5 points.

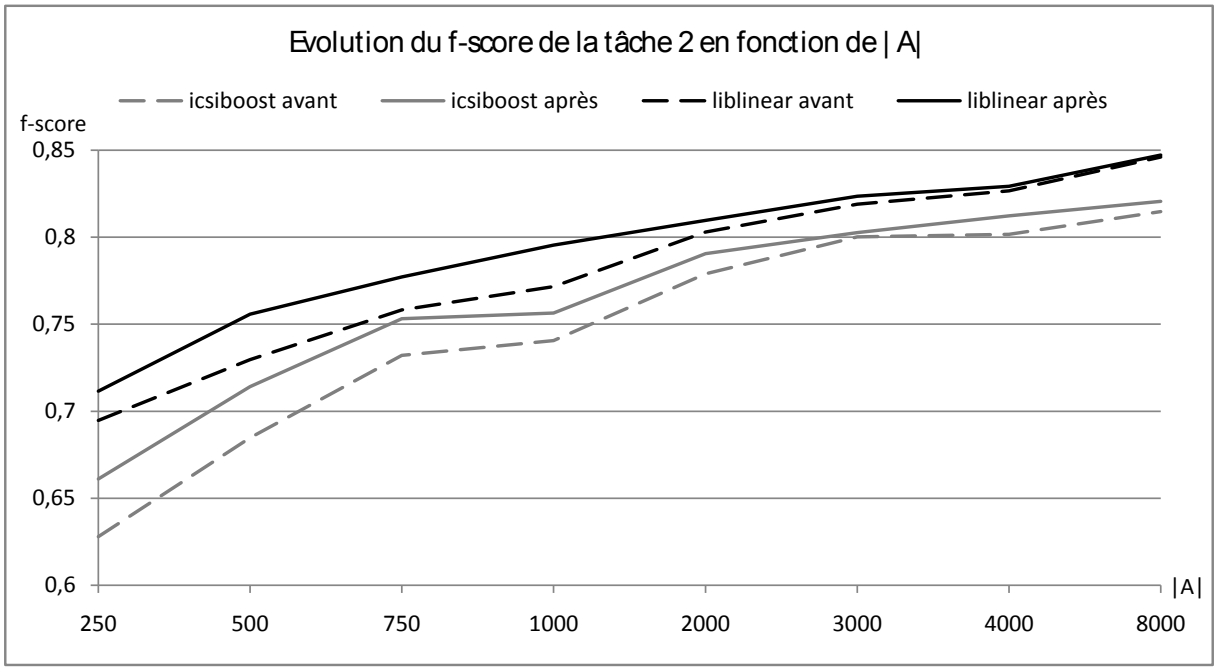
Voici les résultats des autres expériences présentant l'évolution du f-score (en ordonnée) en fonction du nombre d'exemples d'apprentissage (en abscisse) :

Sur la première tâche :



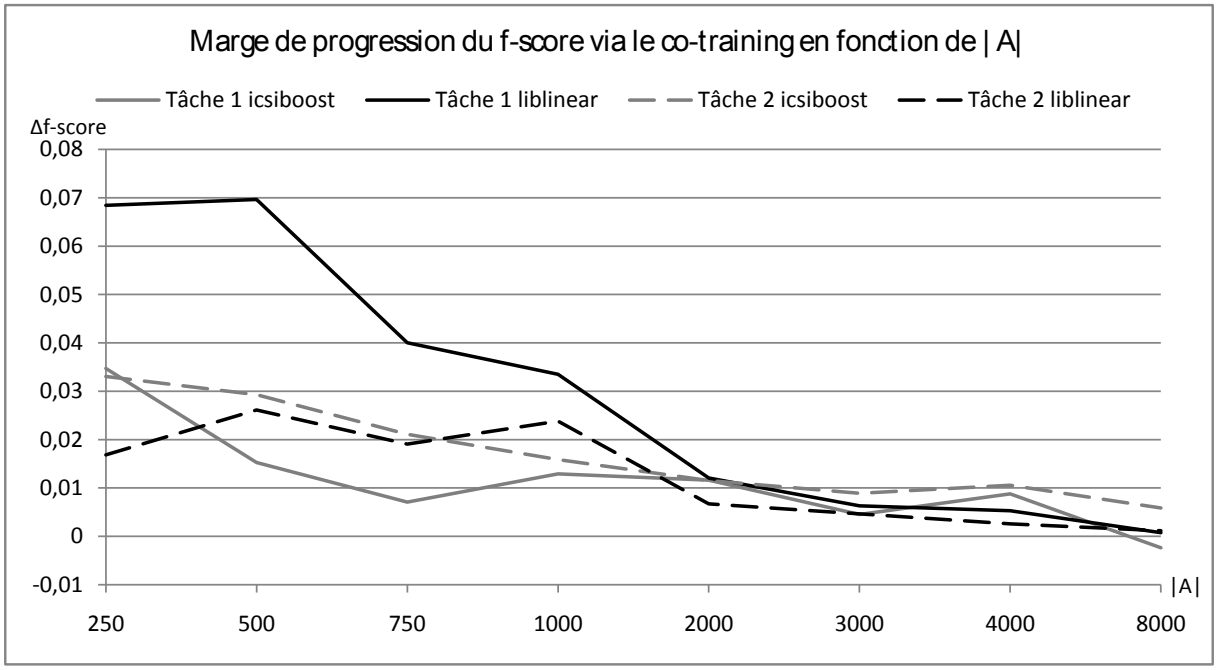
Avec 2000 articles en apprentissage par exemple, même si on reste loin des performances obtenues en ayant des annotations parfaites (en prenant l'intégralité du corpus annoté comme dans la première expérience), grâce au co-training icsiboost atteint le score qu'on aurait eu en entraînant les modèles sur 4000 articles, soit le double d'annotations manuelles.

Sur la deuxième tâche :



Là encore on remarque que le co-training permet d'atteindre un f-score donné en minimisant l'effort d'annotation manuelle.

Voici la progression du f-score obtenue grâce au co-training en fonction du nombre d'exemples d'apprentissage (c'est la différence entre les courbes précédentes) :



De manière générale on constate que la marge de progression du co-training est inversement proportionnelle à la qualité des modèles de départ. Le co-training semble donc particulièrement adapté lorsque le corpus d'apprentissage est insuffisant.

Avec 8000 exemples en apprentissage, on approche le f-score qu'on aurait avec l'ensemble du corpus d'apprentissage. Le co-training semble alors montrer ses limites, avec de faibles progressions des scores, voire même une diminution de la f-mesure d'icsiboost sur la première tâche.

5 Conclusion

Les deux expériences présentées et leurs résultats encourageants montrent que, dès lors que nous avons accès à une quantité suffisante de données non annotées, le co-training peut améliorer les performances de classification, et ce à moindre coût.

Le "bond" que l'on observe sur la première tâche avec 500 exemples montre que la sélection des exemples servant de corpus de départ est primordiale. Allier active-learning et co-training devrait donc donner de meilleurs résultats.

Les corpus à disposition étaient totalement annotés, ce qui a permis de surveiller la qualité de l'annotation automatique de la partition non annotée (notée précédemment U). Les résultats amèneraient à penser que le respect de la distribution à priori des classes est important. Le fait que les courbes issues du co-training épousent vaguement celles de base semble montrer un manque de latitude laissé aux classifieurs pour choisir les exemples à annoter. Il serait donc intéressant de se pencher sur l'algorithme de co-training en lui même, en confrontant plusieurs façons de gérer le pool d'exemples non annotés, et de sélectionner les exemples à intégrer aux modèles.

Références

BLUM A. & MITCHELL T. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the eleventh annual conference on Computational learning theory*, p. 92–100.

CHARTON E., CAMELIN N., ACUNA-AGOST R., GOTAB P., LAVALLEY R., KESSLER R. & FERNANDEZ S. (2008). Pré-traitements classiques ou par analyse distributionnelle : application aux méthodes de classification automatique déployées pour DEFT08. *DEFT'08*.

DENIS F., GILLERON R. & TOMMASI M. (2002). Classification de textes et co-training à partir de textes positifs et non étiquetés. *Actes de la Conférence Francophone sur l'Apprentissage (CAP 2002)*, p. 205–220.

FREUND Y. & SCHAPIRE R. E. (1996). Experiments with a new boosting algorithm. In *In Proceedings of the Thirteenth International Conference on Machine Learning*, p. 148–156 : Morgan Kaufmann.

GUZ U., CUENDET S., HAKKANI-TÜR D. & TUR G. (2007). Co-training Using Prosodic and Lexical Information for Sentence Segmentation. *Interspeech*, p. 2597–2600.

HSU C. W., CHANG C. C. & LIN C. J. (2003). *A practical guide to support vector classification*. Rapport interne, Taipei.

HWA R., OSBORNE M., SARKAR A. & STEEDMAN M. (2003). Corrected co-training for statistical parsers. *ICML-03 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, Washington DC*.

LIN C., WENG R. & KEERTHI S. (2007). Trust region Newton methods for large-scale logistic regression. *Proceedings of the 24th international conference on Machine learning*, p. 561–568.

MÜLLER C., RAPP S. & STRUBE M. (2001). Applying Co-Training to reference resolution. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, p. 352–359.

PIERCE D. & CARDIE C. (2001). Limitations of co-training for natural language learning from large datasets. *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP-2001)*, p. 1–9.

RICCARDI G. & HAKKANI-TUR D. (2005). Active learning : theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, **13**(4), 504–511.

SARKAR A. (2001). Applying co-training methods to statistical parsing. *North American Chapter Of The Association For Computational Linguistics*, p. 1–8.

SCHAPIRE R. & SINGER Y. (2000). BoosTexter : A Boosting-based System for Text Categorization. *Machine Learning*, **39**(2), 135–168.

TRINH A., BUFFONI D. & GALLINARI P. (2008). Classifieur probabiliste avec Support Vector Machine (SVM) et Okapi. *DEFT'08*.

VAPNIK V. (2000). *The Nature of Statistical Learning Theory*. Springer.