Apprentissage des schémas de propagation dans les multi-graphes

Yann Jacob — Ludovic Denoyer — Patrick Gallinari

Université Pierre et Marie Curie - LIP6
Boîte courrier 169
4 place Jussieu 75252 PARIS cedex 05

RÉSUMÉ. *Nous considérons le problème de l'étiquetage de noeuds dans un multi-graphe - ou graphe multi-relationnel - dans lequel les noeuds peuvent être connectés simultanément par différents types de relations. De nombreux problèmes se modélisent ainsi, comme par exemple les réseaux sociaux ou bien les bases de données bibliographiques. Les relations peuvent être explicites (par exemple amitié dans un réseau social) ou bien implicite (par exemple des similarités de contenu calculées sur les données). Nous proposons ici un algorithme d'apprentissage permettant d'exploiter l'information multi-relationnelle pour la tâche d'étiquetage automatique. Cette méthode est capable d'apprendre à combiner de manière optimale l'influence des différents types de relations sur la propagation des étiquettes entre les noeuds du graphe. Nous décrivons des expériences sur quatre corpus qui montrent la capacité du modèle à tirer parti de l'information multi-relationnelle pour des tâches d'étiquetage complexes.*

ABSTRACT. *We consider the problem of labeling nodes in a multi-graph where the nodes may be connected with different types of relations. This type of problem occurs in many situations like for example the analysis of social networks or bibliographic data. Relations may be provided (e.g. friends) or computed (e.g. similarity). We propose a new learning algorithm for exploiting the rich multi-relational information in the labeling task. This method is able to optimally learn combining the influence of different relation types. It is one of the very first algorithms able to handle multi-graph information for this classification task. We describe experiments on four datasets which show the model ability to deal with complex relationships and to take benefit of multi-relational information for complex labeling problems.*

MOTS-CLÉS : *apprentissage, classification automatique, graphe multi-relationnel, réseaux sociaux*

KEYWORDS: *machine learning, automatic classfication, multi-relational graph,social network,web*

1. Introduction

Nous considérons l'analyse de données relationnelles où différents objets sont liés par des relations comme c'est le cas pour des pages Web ou des réseaux sociaux. Ce type de donnée est souvent décrit en utilisant un formalisme de graphe dans lequel les noeuds correspondent aux données et les arcs représentent les relations. L'accès à des masses de données relationnelles de plus en plus complexes et importantes dans beaucoup d'applications a motivé le développement de nouveaux paradigmes et modèles dans différentes communautés de recherche comme la communauté de l'apprentissage automatique ou bien celle de la fouille de données. Parmi les nombreux problèmes émergents, le calcul des scores des noeuds dans un graphe est un problème générique qui survient dans différents applications. Ce score peut correspondre à un indicateur de classe, un rang ou une valeur réelle indiquant l'importance ou la pertinence d'un noeud donné. Certains modèles de calcul de scores sont récemment apparus dans la littérature de l'apprentissage automatique avec deux objectifs principaux : la classification et l'ordonnancement d'objets relationnels. Une idée populaire a été l'exploitation des relations en utilisant le laplacien du graphe pour favoriser localement la régularité des scores de noeuds. Cette intuition, valide dans beaucoup de situations, stipule que les noeuds qui partagent une relation forte doivent avoir un comportement similaire. Ce modèle a été dans un premier temps introduit dans le contexte de l'apprentissage semi-supervisé (Zhou *et al.,* 2005, Zhou *et al.,* 2004, Belkin *et al.,* 2006) où les relations représentent des similarités entre données et l'hypothèse de régularité est utilisée pour propager les étiquettes des noeuds étiquetés vers les noeuds non étiquetés. Il a été étendu pour l'ordonnancement (Agarwal, 2006) et utilisé dans différents contextes comme la détection de spam (Abernethy *et al.,* 2008) ou la recherche d'informations sur internet (Qin *et al.,* 2008) par exemple.

Nous considérons ici le problème de scoring des noeuds dans un graphe et nous proposons deux extensions à des méthodes existantes :

– Tout d'abord, nous considérons des données multi-relationnelles où les objets partagent différents types de relations. Les travaux précédents se concentrent principalement sur les données mono-relationnelles où tous les arcs représentent le même type de relation tandis que beaucoup d'applications concernent des données multi-relationnelles comme les données sociales - avec des relations comme ami, ennemi, suiveur - ou dans des domaines plus classiques comme les citations, co-auteurs ou similarité du contenu dans des données bibliographiques.

– La seconde extension est d'apprendre le poids des relations pour adapter la force ou l'importance de ces poids aux données et à la tâche considérée. Ici encore, toutes les méthodes existantes se sont en deux étapes : premièrement, la force des relations entre deux noeuds est calculée manuellement - une pratique courante consistant à utiliser une fonction de similarité classique entre le contenu des noeuds - puis les scores sont propagés en utilisant ces poids prédéfinis. Ces poids représentent une connaissance a priori et ne sont pas directement corrélés à la tâche à résoudre. Dans notre contexte, apprendre les poids permet de donner plus d'importance à certains types de relations ou aux relations les plus significatives et donc de découvrir automatiquement comment

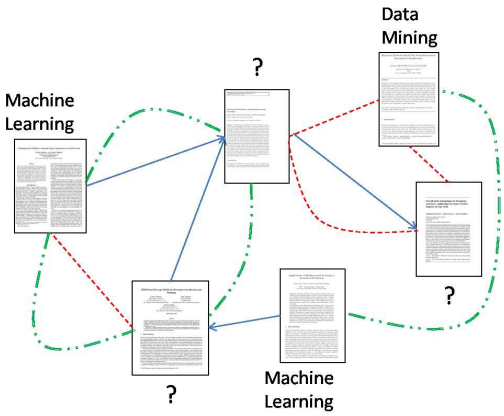


Figure 1. *Etiquetage transductif d'un graphe multi-relationnel*

l'information se propage dans les réseaux.

Le modèle proposé est de plus capable de gérer aussi bien le contenu des noeuds que les nombreux types de relations, là où les nombreux modèles existants se limitent à faire de la propagation d'étiquette sur la structure du graphe, sans prise en compte de contenu.

Dans la section 2 nous introduisons le modèle de régularisation classique pour apprendre les scores de noeuds dans un graphe, dans la section 3, nous présentons notre modèle ainsi que différentes extensions possibles, dans la section 4 nous donnons les résultats d'expériences effectuées sur quatre 4 jeux de données issus de différents domaines applicatifs.

2. Régularisation dans les graphes mono-relationnels

2.1. *Description informelle*

Un des premiers modèles d'apprentissage dans les graphes qui soit capable de gérer aussi bien le contenu que les relations est celui proposé dans (Abernethy *et al.*, 2008) pour l'étiquetage transductif . Il est présenté ici car, même si c'est un modèle mono-relationnel uniquement, il constitue une brique de base du modèle proposé dans cet article et servira de point de départ à nos extensions. Dans le cadre de l'apprentissage transductif, nous considérons un graphe partiellement étiquetés, c'est-à-dire pour lequel nous connaissons les étiquettes de certains noeuds mais pas de tous. Le but du modèle est alors d'être capable de généraliser l'étiquetage partiel à l'ensemble

des noeuds du réseau en utilisant à la fois l'information disponible depuis les noeuds étiquetés et non étiquetés. Basiquement, ce type d'algorithme opère en deux étapes :

– La première étape consiste à entraîner une machine d'apprentissage classique sur le contenu des noeuds étiquetés. Cette machine d'apprentissage sur le contenu seul peut être n'importe quel type de classifieur, comme un perceptron, un SVM ou un modèle génératif. Une fois entrainé sur les noeuds étiquetés, elle sera utilisée pour calculer un score initial, dénoté $\bar{y_i}$, pour tous les noeuds non étiquetés du graphe basé sur le contenu des noeuds uniquement.

– La seconde étape consiste à propager ces scores le long des arcs du graphe. La propagation est faite à travers un terme de régularisation qui pénalise le fait que deux noeuds connectés n'aient pas des scores proches.

A la fin du processus, le score final obtenu pour chaque noeud est alors le résultat d'un compromis entre son score de contenu seul et le score de ses voisins. Dans un premier temps, nous définissons formellement ces deux étapes puis nous montrons pourquoi cet algorithme ne convient pas aux multi-graphes. Nous présentons ensuite dans la section 3 un modèle capable d'apprendre comment propager les étiquettes sur différents types de relations.

2.2. *Notations*

Soit un multi-graphe $G = (V, E)$ défini par :

– Un ensemble de N noeuds $V = (v_1, ...v_N)$. Nous considérons que les noeuds ont une information de contenu (texte, image, ...) décrite par un vecteur. Par la suite, v_k dénotera à la fois le noeud et son vecteur associé i.e. $v_k \in \mathbb{R}^d$ où d est la dimension du vecteur. Un tel vecteur peut être par exemple l'histogramme d'une image ou la description fréquentielle des mots d'un document.

– Un ensemble de R arcs $E = \bigcup_{k=1}^{R} E^{(k)}$ où $E^{(k)}$ est l'ensemble des arcs de type k entre les éléments de V. E peut être vu comme un tenseur tri-dimensionnel $E = \{w_{i,j}^{(k)}\}$ où $w_{i,j}^{(k)}$ est le poids de l'arc de type k entre le noeud v_i et le noeud v_j. $w_{i,j}^{(k)} = 0$ signifie qu'il n'y a aucune relation de type k entre v_i et v_j.

Nous dénotons $V^\ell = \{v_1, ..., v_{N^\ell}\}$ l'ensemble des noeuds étiquetés de taille N^ℓ. Pour tous ces noeuds, y_i est le score réel et connu associé au noeud v_i. Le but est de calculer automatiquement un score $\hat{y}_i$ pour les noeuds non étiquetés restants $V^u = \{v_{N^\ell+1}, ..., v_N\}$ en utilisant à la fois le contenu des noeuds et la structure complexe du multi-graphe. Dans notre contexte transductif, nous considérons que les scores des noeuds étiquetés sont connus pendant tout le processus.

2.3. *Etiquetage sur le contenu seul*

La première étape de la méthode consiste à apprendre un modèle de référence basé sur l'information de contenu seul. Dans ce cadre, un classifieur est entrainé en utilisant les noeuds étiquetés et est utilisé ensuite pour donner un score de référence à tous les noeuds du graphe. Soit θ les paramètres du classifieur et f_θ^{co} la fonction de classification correspondante. La démarche classique consiste ici à minimiser un risque empirique défini sur les noeuds étiquetés comme étant :

$$L^{co}(\theta) = \frac{1}{N^\ell} \sum_{k=1}^{N^\ell} \Delta^{co}(f_\theta^{co}(v_k), y_k) + \lambda ||\theta||^2$$

où $\Delta^{co}(f_\theta^{co}(v_k), y_k)$ est le coût de prédire $f_\theta^{co}(v_k)$ au lieu du score réel y_k et λ est un hyperparamètre de régularisation classique de type régularisation $L2$. Différentes f^{co} fonctions peuvent être utilisées ici. Dans nos expériences, nous avons utilisé une fonction de classification linéaire classique avec un fonction coût de type *hinge loss* et avons effectué la minimisation à l'aide d'un algorithme de descente de gradient stochastique - cf. 4. Le score du noeud v_i prédit par le modèle de contenu seul est ainsi $\bar{y_i} = f_\theta^{co}(v_i)$.

2.4. *Propagation d'étiquettes dans les graphe mono-relationnels*

Les modèles de propagation classiques ont été développés pour des graphes mono-relationnels, i.e. $R = 1$. La plupart de ces modèles reposent sur une hypothèse de régularité qui considère que deux noeuds connectés doivent partager des étiquettes similaires. Cette contrainte est habituellement implémentée à travers un terme de régularisation. La fonction de coût pour ces modèles a la forme générale :

$$\begin{aligned} L^{reg}(\hat{y}_1, ..., \hat{y}_N) = \frac{1}{N^\ell} \sum_{k=1}^{N^\ell} (\hat{y}_k - \bar{y_k})^2 \qquad & \text{(terme 1)} \\ + \alpha \sum_{v_i, v_j} w_{i,j}^{(1)} (\hat{y}_i - \hat{y}_j)^2 \quad & \text{(terme 2)} \end{aligned}$$

où $\hat{y}_i$ est le score prédit du noeud v_i et α est un paramètre de régularisation. $w_{i,j}^{(1)}$ est le poids du noeud entre v_i et v_j dans une graphe mono-relationnel. *Terme 1* mesure l'erreur entre le score prédit et le score de contenu seul sur les noeuds étiquetés, tandis que *terme 2* encourage la régularité. α fait le compromis entre les deux termes et nous permet de nous adapter au degré de régularité du graphe. Une valeur faible de α encourage le modèle à fournir un score final proche du score de contenu seul, alors qu'une valeur importante de α favorise plus la régularité sur le graphe. Les scores finaux des noeuds sont obtenus à travers la minimisation de cette fonction :

$$\hat{y}_1, ..., \hat{y}_N = \underset{\hat{y}_1 ... \hat{y}_N}{\operatorname{argmin}} L^{reg}(\hat{y}_1 ... \hat{y}_N)$$

Là encore, différentes variantes peuvent être utilisés pour les termes dans l'équation 2.4. Différentes techniques d'optimisation ont été proposées pour minimiser ce coût, allant de la descente de gradient (Abernethy *et al*., 2008) aux marches aléatoires (Zhou *et al*., 2005). Nous présentons ici une technique originale d'inférence de laquelle découle l'algorithme d'apprentissage de modèles de propagation multi-relationnels.

3. Propagation d'étiquettes dans les graphes multi-relationnels

3.1. *Définition du modèle*

Nous décrivons ici un nouveau modèle capable d'apprendre à étiqueter dans un contexte multi-relationnel. Ce modèle apprend une combinaison linéaire "optimale" des différents poids de relations. Il est basé sur une nouvelle idée qui exploite un mécanisme d'inférence spécifique. Nous introduisons la fonction de coût dans la section 3.1.1, la technique d'inférence dans la section 3.2 et l'algorithme d'apprentissage dans 3.3. Au lieu d'utiliser directement les poids des relations dans la fonction de coût comme cela est fait dans l'équation 2.4 à travers le facteur $w_{i,j}^{(1)}$, nous allons utiliser une fonction paramétrée $\psi_\gamma(i,j) \in [0;1]$ définie sur chaque paire de noeuds v_i, v_j où γ est l'ensemble des paramètres de cette fonction. Cette fonction sera désignée comme étant la *fonction de propagation* par la suite. Les paramètres de la fonction de propagation seront appris depuis les données comme décrit dans la partie 3.3. Cette fonction fournira des poids normalisés dans [0,1] pour les différents types de relation permettant de quantifier a quel point un certain type de relation est à même de propager une étiquette entre deux noeuds. La fonction de coût utilisée dans ce modèle est définie par $L^{multi}()$:

$$L^{multi}(\hat{y}_1 ... \hat{y}_N) = \frac{1}{N^\ell} \sum_{k=1}^{N^\ell} (\hat{y}_k - \bar{y_k})^2 \qquad \text{(terme 1)}$$

$$+ \alpha \sum_{v_i, v_j} \psi_\gamma(i,j)(\hat{y}_i - \hat{y}_j)^2 \qquad \text{(terme 2)}$$

Notez que $L^{multi}(\hat{y}_1, ... \hat{y}_N)$ a la même forme que $L^{reg}(\hat{y}_1, ... \hat{y}_N)$ à l'exception que $w_{i,j}^{(1)}$ a été remplacé par la fonction $\psi_\gamma(i,j)$. Le but de la fonction de propagation $\psi_\gamma(i,j)$ est d'agglomérer le poids des différents types de relations.

3.1.1. *Fonction de propagation*

Dénotons $\Phi(i,j)$ un vecteur qui décrit les différents relations entre v_i et v_j :

$$\Phi(i,j) = \begin{pmatrix} w_{i,j}^{(1)} \\ \vdots \\ w_{i,j}^{(R)} \end{pmatrix}$$

Nous définissons la fonction de propagation comme étant une fonction logistique sur le vecteur Φ :

$$\psi_\gamma(i,j) = \text{logit} < \gamma; \Phi(i,j) > \quad = \text{logit} \sum_{r=1}^{R} \gamma_r . w_{i,j}^{(t)}$$

Elle est définie à l'aide d'un paramètre γ_r pour chaque type de relation dont la valeur reflète l'importance de la relation r dans la propagation des scores entre les noeuds . Le vecteur de paramètres γ est appris depuis les données comme expliqué dans la partie 3.3 et non pas fixé manuellement, permettant au modèle de s'adapter automatiquement à différents types de relations pour lesquelles il est capable d'apprendre l'importance relative. Différentes variantes de cette fonction de propagation peuvent être définies. Par exemple le contenu sur les noeuds peut être incorporé dans l'expression de $\psi_\gamma(i,j)$. Ce point ne sera pas discuté ici.

3.2. *Inférence*

L'inférence consiste à calculer les scores prédits $\hat{y}^*_{N^\ell+1}, ...\hat{y}^*_N$ sur les noeuds non étiquetés comme étant les valeurs qui minimisent L^{multi} :

$$\hat{y}^*_{N^\ell+1}, ...\hat{y}^*_N = \underset{\hat{y}_{N^\ell+1}, ...\hat{y}_N}{\text{argmin}} \; L^{multi}(\hat{y}_{N^\ell+1}, ...\hat{y}_N)$$

Notons que cette fonction considère uniquement les noeuds non étiquetés étant donné que pour les noeuds étiquetés, nous connaissons déjà les scores i.e $\hat{y}_i = y_i$.

Nous proposons de résoudre la minimisation de cette fonction à travers un algorithme itératif basé sur une méthode de descente de coordonnées. Cette méthode consiste à minimiser la fonction L^{multi} coordonnée par coordonnée. $L^{multi}()$ étant une fonction convexe par rapport à $(\hat{y}_{N^\ell+1}, ...\hat{y}_N)$, l'algorithme converge vers le minimum de cette fonction. Cet algorithme est décrit dans la figure 2. A l'itération t, on calcule le minimum sur une coordonnée particulière k connaissant les scores prédits sur les autres noeuds $\hat{y}_1^{(t)}, ..., \hat{y}_{k-1}^{(t)}, \hat{y}_{k+1}^{(t)}, ..., \hat{y}_N^{(t)}$. Ce minimum est trouvé en résolvant l'équation suivante :

$$\hat{y}_k^{(t+1)} = \underset{\hat{y}_k}{\text{argmin}} \; L^{multi}(\hat{y}_1^{(t)}, ..., \hat{y}_{k-1}^{(t)}, \hat{y}_k, \hat{y}_{k+1}^{(t)}, ..., \hat{y}_N^{(t)})$$

Cette minimisation est calculée seulement pour les noeuds non étiquetés et obtenue quand :

$$\frac{\partial L^{multi}(\hat{y}_{N^\ell+1}^{(t)}, ..., \hat{y}_{k-1}^{(t)}, \hat{y}_k, \hat{y}_{k+1}^{(t)}, ..., \hat{y}_N^{(t)})}{\partial \hat{y}_k} = 0$$

Algorithme d'inférence

pour tout noeud non étiqueté $v_i \in V^u$ **faire**
  $\hat{y}_i \leftarrow y_i$ {Initialisation avec le contenu seul}
fin pour
pour tout noeud non étiqueté $v_i \in V^\ell$ **faire**
  $\hat{y}_i \leftarrow \bar{y_i}$ {Initialisation des noeuds étiquetés}
fin pour
t=0 ;
répéter
  pour tout noeud non étiqueté $v_i \in V^u$ **faire**
    $\hat{y}_k^{(t+1)} = \frac{\frac{1}{N^\ell}\bar{y_i} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i)+\psi_\gamma(i,k))\hat{y}_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i)+\psi_\gamma(k,i))\right)}$
    $t = t + 1$;
  fin pour
jusqu'à convergence

Figure 2. *L'algorithme d'inférence du modèle multi-relationnel*.

La solution est donc :

$$\hat{y}_k^{(t+1)} = \frac{\frac{1}{N^\ell}\bar{y_k} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\hat{y}_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i) + \psi_\gamma(k,i))\right)}$$

Cet aglorithme consiste donc à itérer l'équation 3.2 pour chaque noeud non étiqueté v_k jusque convergence - Figure 2.

3.3. *Apprentissage des paramètres de la fonction de propagation*

Nous présentons ici une méthode originale pour apprendre les paramètres γ de la fonction de propagation. Nous considérons que les paramètres sur le contenu seul ont été appris au préalable comme expliqué dans la section 2.3. Définissons $\Delta(\hat{y}_i, y_i)$ la fonction de coût sur les scores prédits $\hat{y}_i$ par rapport aux scores réels y_i. Le coût empirique est défini sur les noeuds étiquetés comme étant :

$$\forall v_k \in V^\ell, E(\gamma) = \frac{1}{N^\ell}\sum_{i=1}^{N^\ell}\Delta_\gamma(\hat{y}_k^*; y_k)$$

où $\hat{y}_k^*$ est obtenu en utilisant la procédure d'inférence i.e. :

$$\hat{y}_1^*, ...\hat{y}_N^* = \underset{\hat{y}_1,...\hat{y}_N}{\operatorname{argmin}} L^{multi}(\hat{y}_1, ...\hat{y}_N)$$

Cette fonction est difficile à minimiser car elle implique la minimisation d'une fonction argmin.

Considérons l'étape de la procédure d'inférence qui vise à prédire le score $\hat{y}_k^{(t+1)}$ au temps t. Supposons que nous connaissons tous les scores pour les noeuds $v_i \neq v_k$. La procédure d'inférence vise à prédire une valeur $\hat{y}_k^{(t+1)}$ aussi proche que possible du score y_k de v_k. Ce score prédit dépendra directement de la valeur des paramètres γ comme expliqué dans l'équation 3.2. L'idée de notre algorithme d'apprentissage est de trouver les valeurs de γ qui prédiront correctement le score du noeud v_k connaissant le score réel de tous ces voisins. Apprendre γ consiste à minimiser le coût de prédiction Δ défini pour chacun de ses noeuds étiquetés v_k :

$$\Delta_\gamma(\hat{y}_k^*; y_k) \approx \Delta\left(\frac{\frac{1}{N^\ell}\bar{y_k} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))y_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\right)}, y_k\right)$$

Nous considérons dans ce papier l'utilisation du coût classique de l'erreur au carré :

$$\begin{aligned}\gamma^* &= \underset{\gamma}{\operatorname{argmin}} \frac{1}{N^\ell} \sum_{k=1}^{k=N^\ell} \Delta_\gamma(\hat{y}_k^*; y_k) \\ &= \underset{\gamma}{\operatorname{argmin}} \frac{1}{N^\ell} \sum_{k=1}^{k=N^\ell} \left(\frac{\frac{\bar{y_k}}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))y_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\right)} - y_k \right)^2\end{aligned}$$

Ce coût peut être minimisé par un algorithme classique de descente de gradient.

4. Expériences

4.1. *Données*

Nous avons lancé des expériences sur quatre jeux de données :

– Le corpus **Cora** est un corpus d'articles scientifiques avec 10 catégories thématiques. Le contenu de chaque noeud correspond au vecteur TF-IDF normalisé du résumé de chaque article. Il possède 5 relations différentes :

- La relation *Cite* entre v_i et v_j vaut 1 si l'article i cite l'article j.

- La relation *estCité* entre v_i et v_j vaut 1 si l'article j cite l'article i.

- La relation *mêmeAuteur* vaut 1 les deux papiers partagent le même auteur. Nous avons aussi extrait deux relations de similarité :

- La relation *Titre* est définie comme étant le produit scalaire entre les vecteurs TF-IDF normalisés du titre des deux papiers.

- La relation *Résumé* est définie comme étant le produit scalaire entre les vecteurs TF-IDF normalisés du résumé des deux papiers.

La tâche considérée avec ce corpus est la classification multi-classe, mono-label.

– Le corpus **Grand Cora** est le même corpus que ci dessus, mais avec les 25 plus grandes catégories au lieu de 10. Dans ce corpus, chaque document peut appartenir à une ou plusieurs catégories. La tâche est donc de la classification multi-label.

– Le corpus **Flickr** (Peters *et al.*, 2010) correspond aux images et à l'information textuelle associée extraite de Flickr. Chaque image est décrite par une description textuelle trouvée sur Flickr et a été annotée avec différentes étiquettes. La tache est l'étiquetage automatique, i.e. associer une ou plusieurs étiquettes à chaque image. Les images sont connectées avec 4 relations différentes :

- La relation *Auteur* : deux images sont connectées avec un poids 1 si elles partagent le même auteur.

- La relation *Ami* : deux images sont connectées avec un poids 1 si les auteurs sont amis.

- La relation *Commentaire* : deux images sont connectées avec un poids 1 si le même utilisateur a commenté les deux images.

- La relation *MêmeMois* : deux images sont connectées avec un poids 1 si elles ont été publiées pendant la même période de 30 jours. La tâche considérée avec ce corpus est la classification multi-classe, multi-label.

– Le corpus **Email** est un corpus extrait des emails des auteurs de cet article. Les emails ont été classifiés manuellement en 16 dossiers différents. Chaque email est décrit par son contenu textuel et peut être connecté avec 4 types différents de relations :

- La relation *Auteur* : deux emails sont connectés avec le poids 1 si ils ont été envoyés par le même utilisateur.

- La relation *Destinataire* : deux emails sont connectés avec le poids 1 si ils ont été envoyés au même destinataire.

- La relation *Copie* : deux emails sont connectés avec le poids 1 si ils partagent le même utilisateur en copie.

- La relation *MêmeJour* : deux emails sont connectés avec le poids 1 si ils ont été envoyés le même jour.

La tâche considérée avec ce corpus est la classification multi-classe, mono-label.

La tableau 1 donne des statistiques de base sur les différents corpora.

Corpus	Nb noeuds	Nb arcs	Nb relations	Nb catégories	Tâche
Cora	24 245	712 440	5	10	Monolabel
Grand Cora	24 245	712 440	5	25	Multilabel
Flickr	1 995	316 002	4	10	Multilabel
Emails	4 408	$\approx$ 1 M	4	15	Monolabel

Tableau 1. *Statistiques sur les corpora*

Alors que le modèle présenté est un modèle permettant le calcul des scores sur les noeuds d'un multi-graphe, nous apprenons ici un modèle d'apprentissage pour chaque catégorie possible du problème de classification. Dans le cas de la classification mono-label, la catégorie avec le plus grand score est la catégorie assignée au noeud. Dans le cas multi-label, les catégories possibles sont ordonnées par ordre de score prédit.

4.2. *Modèles et mesures d'évaluation*

Pour ces expériences, nous avons comparé trois modèles différents :

– Le modèle **Contenu Seul**. Nous avons utilisé un Perceptron avec un coût de type *Hinge Loss*.

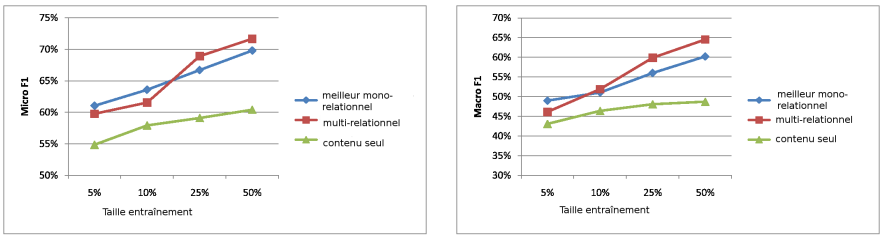
– Le modèle **Mono-Relationnel** qui correspond au modèle régularisé décrit dans la partie 2.4. Ce modèle ne prend en compte qu'une seule relation à la fois. Des tests ont été faits avec chaque différente relation pour chaque corpus.

– Le modèle **Multi-Relationnel** qui a été présenté dans la section 3 et qui est capable de considérer simultanément toutes les relations.

Nous avons évalué toutes les tâches de classification mono-label avec les mesures classiques de micro-F_1 et macro-F_1. Le macro-F_1 est la moyenne des F_1 sur toutes les catégories du problème alors que le micro-F_1 est la moyenne pondérée par la taille de chaque catégorie. Pour les taches de classification multi-label, nous avons utilisé la précision au rang 1 (p@1). Nous avons fait des expériences avec différentes tailles de base d'entraînement et avec des valeurs de α différentes : 0.01, 1, 100. Chaque expérience a été lancée trois fois et les résultats sont présentés comme étant leur moyenne. Quand cela n'est pas précisé, nous avons décrit les meilleurs résultats par rapport à α.

4.3. *Résultats*

Nous présentons dans la figure 3 la performance obtenue sur le corpus CORA en fonction de la taille en entraînement. Les performances sur les autres corpus sont données dans les tableaux 2, 3 et 4. Dans ces figures, 5% signifie que 5% des noeuds ont été utilisés en entraînement pendant l'apprentissage. Les micro et macro performances montrent le même comportement : l'information relationnelle aide à augmenter la performance par rapport au classifieur sur le contenu seul. Plus on utilise de données



Micro F1 Macro F1

Figure 3. *F1 en fonction de la taille en entraînement*

Taille	Contenu Seul	Meilleur Mono-relationnel	Multi-relationnel
0.05	0.56	**0.62**	0.57
0.1	0.57	**0.65**	0.63
0.25	0.58	0.67	**0.72**

Tableau 2. *P@1 pour le Grand Cora en fonction de la taille en entraînement*

en entraînement, plus l'augmentation est importante. Le modèle multi-relationnel demande plus de donnée que le mono-relationnel pour apprendre ce qui est intuitivement logique puisque qu'il nécessite l'apprentissage d'un plus grand nombre de paramètres. Quand la quantité de données d'apprentissage est suffisante (respectivement plus de 10% et 15% sur les deux figures), le modèle multi-relationnel est capable de tirer profit de l'information additionnelle fournie par les relations multiples et améliore significativement la performance par rapport au meilleur modèle mono-relationnel. La figure 4 montre la performance du modèle mono-relationnel en utilisant les différentes relations et celles du modèle multi-relationnel. On peut voir que dans deux corpus, une relation est clairement meilleure que toutes les autres. Pour ces corpus, le modèle multi-relationnel est capable d'apprendre quelle est la meilleure relation à sélectionner pour faire au moins aussi bien que le modèle mono-relationnel. Le même comportement peut être observé dans les corpus Flickr et le Grand Cora (tableaux 3 et 2). Notons que, en plus d'améliorer les performances du modèle mono-relationnel, le modèle multi-relationnel n'a pas besoin de savoir au préalable quelle relation il faut utiliser et trouve automatiquement comment combiner de manière optimale les différents types de relations.

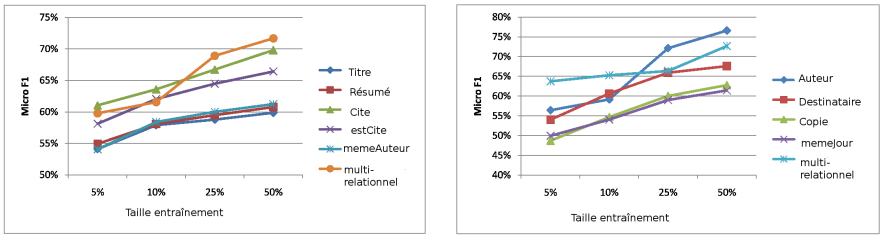
La figure 5 montre la performance du modèle multi-relationnel sur le corpus Cora en fonction de l'itération d'inférence (voir l'algorithme 2). Elle montre qu'environ 5 itérations sont assez pour obtenir presque la meilleure performance. Il s'agit donc d'un processus d'inférence rapide.

Taille	Contenu Seul	Meilleur Mono-relationnel	Multi-relationnel
0.05	0.20	0.27	**0.29**
0.1	0.23	**0.31**	0.30
0.3	0.27	0.35	**0.35**
0.5	0.30	0.37	**0.58**

Tableau 3. *P@1 pour Flickr en fonction de la taille en entraînement*

Taille	Contenu Seul	Meilleur Mono-relationnel	Multi-relationnel
0.05	0.47	0.56	**0.63**
0.1	0.53	0.60	**0.65**
0.25	0.54	**0.72**	0.66
0.5	0.59	**0.76**	0.72

Tableau 4. *Micro F1 pour le corpus Emails*



CORA Emails

Figure 4. *Micro-F1 du modèle Mono-Relationnel pour chaque relation vs Performance du modèle multi-relationnel*

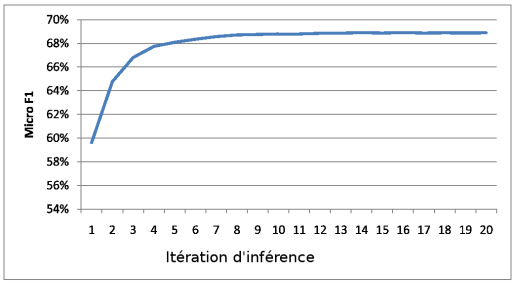


Figure 5. *Micro-F1 sur CORA en fonction de l'itération d'inférence*

5. Etat de l'art

5.1. *Calcul de score dans les graphes*

Comme présenté dans l'introduction, la plupart des modèles d'étiquettage de graphe considèrent uniquement des graphes mono-relationnels. Dans la communauté de l'apprentissage automatique, l'étiquettage de graphe a été étudié avec deux approches principales : les méthodes régularisées et les méthodes itératives. Nous ne présentons ici que l'état de l'art dans le cadre des méthodes régularisée qui sont les méthodes les plus proches du travail présenté ici. Cette famille de méthodes a été initiallement motivée par le travail sur l'apprentissage semi-supervisé dans un contexte transductif. Cette branche de la recherche a été initiée par le travail de Zhou (Zhou *et al.,* 2004) et Belkin (Belkin *et al.,* 2006). Tous ces modèles reposant sur la minimisation d'une fonction objectif combinant une fonction de coût de classification classique sur les données étiquettées et un terme de régularisation sur toutes les données de la forme $L = \sum_{i,j} \omega_{ij}(f_i - f_j)^2$ où f_i est le score du noeud i et ω_{ij} est le poids de l'arc entre i et j. Beaucoup de variantes différentes de ce modèle existent. La plupart des modèles ont été développés pour la propagation d'étiquette seulement et dans le cas binaire, i.e. l'algorithme part des noeuds étiquettés et essaie de propager cette information a travers l'hypothèse de régularité. C'est le cas dans (Zhou *et al.,* 2004, Belkin *et al.,* 2006) qui considèrent des graphes non orientés. Notons que (Zhou *et al.,* 2004) présente un lien intéressant entre les modèles régularisés et la formulation des marches aléatoires dans le cas de la propagation d'une étiquette binaire. Certaines extensions pour les graphes orientés ont été proposées, par exemple (Zhou *et al.,* 2005). Le premier modèle général capable de traiter aussi bien l'information de contenu que la propagation est proposé dans (Abernethy *et al.,* 2008). Un modèle d'ordonnancement est décrit dans (Agarwal, 2006). Ces travaux ont inspiré des auteurs dans différents domaines, comme (Qin *et al.,* 2008) pour ordonner des pages web dans une recherche. (Qin *et al.,* 2008) définit le problème d'apprentissage relationnel comme l'apprentissage d'une correspondance $y = f(h(X), R)$ où $h(X)$ est une fonction sur le contenu seul et R définit la structure relationnelle du problème. En utilisant ce formalisme, leur travail correspond à apprendre la fonction h tandis que f est fixé. Notre modèle consiste ici au cas où f et h sont appris.

5.2. *Calcul de score dans les multi-graphes*

Peu de travaux se sont intéressés aux données multi-relationnelles et aux multi-graphes. (Gjoka *et al.,* 2010) a étudié le problème de l'échantillonage dans les multi-graphes. (Chen *et al.,* 2009) a étudié les bornes d'erreurs des classifieurs régularisés basés sur les méthodes de noyau dans les multi-graphes. (Bhagat *et al.,* 2007) considère les blogs étiquettés selon différents liens web. Ces liens sont catégorisés selon l'endroit où apparaît l'hyperlien du blog : dans une entrée du blog, dans un commentaire ou dans la section amis. Un algorithme pour l'étiquettage de multi-graphe est

proposé. Toutes les relations ont le même poids et ne sont pas apprises. Cet algorithme est donc incapable d'adapter le poids des différentes relations selon leur importance pour la tâche d'étiquettage. Dans (Peters *et al.,* 2010), un algorithme est proposé pour l'annotation de réseaux multi-relationnels et multi-label comme une extension du *Modèle itératif* ICA. (Zhou *et al.,* 2006) étend les idées introduites dans (Zhou *et al.,* 2005) aux multi-graphes. Ils développent un algorithme de clustering des noeuds dans un hypergraphe en utilisant une forme de clustering spectral et en dérivent une méthode de classification. Cela n'est valide que pour la classification binaire et là encore, aucune contenu n'est considéré pour les noeuds.
Dans (Kato *et al.,* 2008), une méthode d'inférence transductive pour les graphes multi-relationnels est proposée. Ils utilisent un algorithme d'inférence EM, qui optimise alternativement le score des noeuds dans l'étape E et le poids des arcs dans l'étape M, jusque convergence, en se basant au préalable sur une distribution de student pour les scores. (Wang *et al.,* 2009) considère le problème de la régularisation de multi-graphes pour l'annotation de vidéos. Les multi-graphes sont fusionnés en un seul graphe par une combinaison linéaire, les poids de la combinaison linéaire étant fixés par un paramètre qui fait un compromis entre prendre les relations les plus régulières et considérer toute l'information relationnelle. Cette algorithme a comme inconvénient de ne pas optimiser le critère d'évaluation mais un coût paramétré à la main.

6. Conclusion

Nous avons proposé un modèle pour la classification multi-classe, multi-label dans les multi-graphes. Ce modèle repose sur une idée originale d'inférence et sur un algorithme d'apprentissage. Il a comme cas particuliers différents modèles mono-relationnels existants. Les expériences lancées sur différents jeux de données montrent la capacité du modèle à extraire l'information pertinente parmi une information multi-relationnelle riche. Ce modèle obtient de meilleures performances que sa contrepartie mono-relationnelle pour peu qu'il y'aie assez de données d'entraînement. Nos perspectives de travail consistent à envisager différentes extensions de ce modèle, particulièrement aux données temporelles et dynamiques afin d'analyser en profondeur le comportement d'un tel algorithme.

Remerciements

Ce travail a été en partie soutenu par l'Agence Nationale de la Recherche (Projet Fragrances, ANR-08-CORD-008-01 et Projet ExDeus/Cedres, ANR-09-CORD-010-04).

-

7. Bibliographie

Abernethy J., Chapelle O., Castillo C., WITCH : A New Approach to Web Spam Detection, Technical report, Yahoo ! Research, 2008.

Agarwal S., « Ranking on graph data », *Proceedings of the 23rd international conference on Machine learning*, ICML '06, ACM, New York, NY, USA, p. 25-32, 2006.

Belkin M., Niyogi P., Sindhwani V., « Manifold Regularization : A Geometric Framework for Learning from Labeled and Unlabeled Examples », *J. Mach. Learn. Res.*, vol. 7, p. 2399-2434, December, 2006.

Bhagat S., Rozenbaum I., Cormode G., « Applying link-based classification to label blogs », *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, WebKDD/SNA-KDD '07, ACM, New York, NY, USA, p. 92-101, 2007.

Chen H., Li L., Peng J., « Error bounds of multi-graph regularized semi-supervised classification », *Information Sciences*, vol. 179, n° 12, p. 1960 - 1969, 2009. Special Section : Web Search.

Gjoka M., Butts C. T., Kurant M., Markopoulou A., « Multigraph Sampling of Online Social Networks », *CoRR*, 2010.

Kato T., Kashima H., Sugiyama M., « Integration of Multiple Networks for Robust Label Propagation », *SDM*, p. 716-726, 2008.

Peters S., Denoyer L., Gallinari P., « Iterative Annotation of Multi-relational Social Networks », *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '10, IEEE Computer Society, Washington, DC, USA, p. 96-103, 2010.

Qin T., Liu T.-Y., Zhang X.-D., Wang D.-S., Xiong W.-Y., Li H., « Learning to rank relational objects and its application to web search », *Proceeding of the 17th international conference on World Wide Web*, WWW '08, ACM, New York, NY, USA, p. 407-416, 2008.

Wang M., Hua X.-S., Hong R., Tang J., Qi G.-J., Song Y., « Unified video annotation via multigraph learning », *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 19, p. 733-746, May, 2009.

Zhou D., Bousquet O., Lal T. N., Weston J., Schölkopf B., « Learning with local and global consistency », *Advances in Neural Information Processing Systems 16*, MIT Press, p. 321-328, 2004.

Zhou D., Huang J., Schölkopf B., « Learning from labeled and unlabeled data on a directed graph », *Proceedings of the 22nd international conference on Machine learning*, ICML '05, ACM, New York, NY, USA, p. 1036-1043, 2005.

Zhou D., Huang J., Schölkopf B., « Learning with hypergraphs : Clustering, classification, and embedding », *Advances in Neural Information Processing Systems (NIPS) 19*, MIT Press, p. 2006, 2006.