Apprentissage automatique de la propagation des étiquettes dans les réseaux sociaux multirelationnels

Yann Jacob, Ludovic Denoyer, Patrick Gallinari

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

RÉSUMÉ. Nous considérons le problème consistant à apprendre à annoter des documents avec des concepts ou des mots clefs dans des réseaux d'information avec contenu, où les documents peuvent partager plusieurs types de relation. Ces concepts associés au document dépendent à la fois de son contenu et de ses voisins dans le graphe à travers les différentes relations. Nous formalisons ce problème comme de la classification multiétiquette dans un multigraphe, les nœuds étant les documents et les arcs représentant les différentes relations. Nous introduisons une nouvelle méthode d'étiquetage des nœuds qui exploite à la fois le contenu et la structure multirelationnelle du graphe. L'algorithme apprend également à pondérer les différents types de relations selon leur importance pour la tâche d'annotation. Les expériences sur les différents corpora correspondent à différentes tâches d'annotation sur des articles scientifiques, des emails et des images de Flickr et montrent que le modèle est capable de tirer parti de l'information relationnelle riche.

ABSTRACT. We consider the problem of learning to annotate documents with concepts or keywords in content information networks, where documents may share multiple relation types. The concepts associated to a document will depend both on its content and on its neighbors through the different relations. We formalize this problem as multilabel classification in a multigraph, the nodes being the documents and the edges representing the different relation types. We introduce a new method for learning node labeling which exploits the document content and the multirelational structure of the graph. The algorithm also learns to weight the different relation types according to their importance for the annotation task. We perform experiments on different corpora corresponding to different annotation tasks on scientific articles, emails and Flickr images and show how the model may take advantage of the rich relational information.

MOTS-CLÉS : classification automatique, graphe multirelationnel, réseaux sociaux.

KEYWORDS: automatic classification, multirelational graph, social network.

DOI:10.3166/DN.15.1.79-99

1. Introduction

La classification de données sémantiques relationnelles existe dans différents contextes, comme la classification de pages web (Zhang *et al.*, 2006), la classification de documents (Bilgic *et al.*, 2007), la détection de web spam (Abernethy *et al.*, 2008), l'annotation d'image ou de vidéo (Cao *et al.*, 2008 ; Wang *et al.*, 2009), l'étiquetage de blog (Bhagat *et al.*, 2007). En plus de la description par l'élément de contenu, une information relationnelle est disponible et peut être utilisée comme une source d'information complémentaire pour classifier les objets.

La sémantique des relations diffère selon le problème et peut être exploitée de différentes manières. Cette information peut être implicite ; dans ce cas elle est inférée directement depuis les données. Par exemple, la similarité entre deux éléments de texte, des images ou des vidéos a été largement explorée dans divers contextes de classification. L'information peut être aussi explicite comme les hyperliens sur le web, les citations dans un réseau bibliographique ou des relations d'amitié dans un réseau social.

La plupart des travaux existants dans ce domaine ont considéré que les éléments sont connectés avec un seul type de relation, comme les citations, l'amitié etc. Il y a cependant beaucoup de problèmes concrets pour lesquels plusieurs types de relations sont disponibles et peuvent ou doivent être exploités. Considérons la tâche d'apprendre à annoter des images, qui est utile par exemple pour la recherche par mot clef. Du fait de la diversité des images et contextes d'annotation, les images seules ne fournissent pas assez d'information pour l'étiquetage. Il peut alors être nécessaire d'utiliser toute l'information, étiquettes et relations, fournie par les différents utilisateurs. Pour cela, il est nécessaire de savoir comment utiliser les différentes relations entre les images, qui ont chacune leur sémantique propre, et comment exploiter la valeur de chacune d'entre elles. Plusieurs travaux récents ont commencé à explorer cette direction. Par exemple, (Wang *et al.*, 2009) exploite les relations entre différentes structures clefs représentées par plusieurs modalités pour annoter des captures de vidéos, (Tsuda *et al.*, 2005) classifie des protéines en utilisant des réseaux de protéines multiples. Le principal problème de ces approches est que l'information de contenu est prise en compte uniquement à travers le calcul de similarités et n'est pas utilisée directement pour la classification. Par ailleurs ces algorithmes utilisent des méthodes d'optimisation complexes qui ne sont pas adaptées pour les grands réseaux.

Nous considérons le problème d'annotation de documents dans des réseaux avec une information de contenu, où les éléments peuvent partager de multiples relations implicites ou explicites. Un exemple d'un tel réseau est le site Flickr avec des relations comme "ami", "auteur" ou "commentaire", des mesures de similarité ou des métadonnées qui peuvent être utilisées pour inférer des relations entre les éléments de contenu. La figure 1 montre exemple de problème d'étiquetage multirelationnel transductif dans un réseau bibliographique Le graphe présente différents types d'arcs dirigés et non dirigés modélisant différentes relations (représentées par des couleurs ou pointillés). Ici il y a trois relations : deux articles sont reliés s'ils ont le même

auteur (SameAuthor) ou publiés la même année (SameYear) ou que l'un cite l'autre (Cite). Certains nœuds sont étiquetés (par exemple Machine Learning) et d'autres sont à étiqueter par le modèle (ceux annotés par un point d'interrogation).

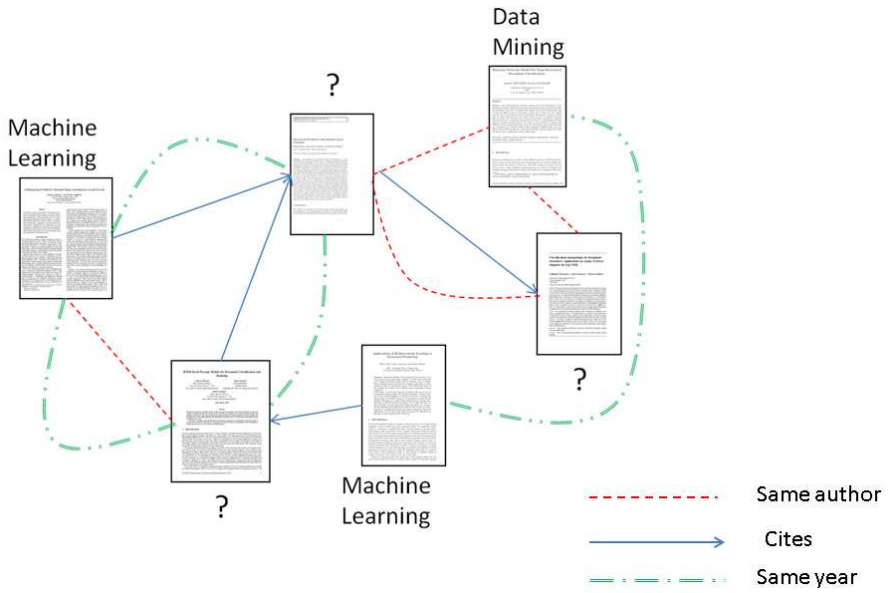


Figure 1. Étiquetage multirelationnel transductif

Nous proposons un nouveau modèle d'apprentissage pour annoter des documents. Il est basé sur le cadre de la classification transductive où sont disponibles en entraînement à la fois les données étiquetées et non étiquetées avec les différents types de relations. Ce cadre a été introduit tout d'abord dans le contexte de l'apprentissage semi-supervisé (Zhou *et al.*, 2005 ; 2004 ; Belkin *et al.*, 2006) pour des réseaux monorelationnels et nous allons l'étendre aux réseaux multirelationnels en apprenant automatiquement à pondérer les différents types de relations, en fonction de leur pertinence et importance pour la résolution de la tâche de classification ou d'annotation. Il s'agit du premier modèle pour la classification dans les graphes capable d'apprendre simultanément à partir du contenu des éléments à classifier et des relations multiples hétérogènes entre ces éléments. Les expériences sur différents jeux de données nous permettent d'évaluer la capacité du modèle à traiter une grande variété de contextes. Le modèle est comparé à des algorithmes monorelationnels et multirelationnels de l'état de l'art et démontre à la fois sa capacité à agréger l'information des différents types de relations, et à mélanger également le contenu et la structure pour améliorer la classification.

L'article est organisé comme suit. À la section 3 nous introduisons un cadre d'apprentissage transductif classique pour apprendre les scores des nœuds dans un graphe monorelationnel, à la section 4 nous introduisons notre extension aux données multi-

relationnelles, à la section 5 nous présentons les résultats d'expériences sur différents jeux de données avec des comparaisons aux méthodes de l'état de l'art. La section 6 présente l'état de l'art.

2. Notations

Nous considérons un multigraphe $G = (V, E)$. Il est défini par :

– Un ensemble de N nœuds $V = (v_1, ...v_N)$. Nous considérons que les nœuds ont une information de contenu (texte, image...) décrite par un vecteur de caractéristiques. Par la suite, v_k dénotera à la fois le nœud et son vecteur de caractéristiques associé i.e $v_k \in \mathbb{R}^d$ où d est la dimension du vecteur de caractéristiques. Ce vecteur peut être un histogramme pour une image ou une distribution de fréquences de mots pour un document textuel.

– Un ensemble de R types d'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 est 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 nœud v_i et le nœud v_j. Si $w_{i,j}^{(k)} = 0$ alors il n'y a pas de relation de type k entre v_i et v_j.

Nous notons $V^\ell = \{v_1, ..., v_{N^\ell}\}$ l'ensemble des nœuds étiquetés où N^ℓ est le nombre de nœuds étiquetés. Pour tous ces nœuds, y_i est un score connu associé au nœud v_i. Le but est de calculer automatiquement un score $\hat{y}_i$ pour tous les nœuds non étiquetés restants $V^u = \{v_{N^\ell+1}, ..., v_N\}$, où $N^u = N - N^\ell$ est le nombre de nœuds non étiquetés, en utilisant à la fois le contenu des nœuds et la structure complexe du multigraphe. Pour $v_i \in V^\ell$, nous posons par définition $\hat{y}_i = y_i$.

Dans notre contexte transductif, nous considérons que les scores des nœuds étiquetés sont connus durant tout le processus.

3. Classification dans les graphes monorelationnels

3.1. Description informelle

Nous introduisons ci-dessous un classifieur transductif général pour les graphes monorelationnels. La plupart des modèles dans l'approche transductive font de la propagation d'étiquette depuis un faible nombre de nœuds étiquetés vers les nœuds non étiquetés. Ils n'utilisent pas de classifieur sur le contenu durant le processus d'étiquetage et ne font que propager les étiquettes. Un des premiers modèles de scorage pour les graphes, capable d'utiliser à la fois le contenu des nœuds et les relations durant l'inférence, a été récemment proposé dans Abernethy *et al.* (2008). Un algorithme similaire, opérant dans un contexte monorelationnel sera utilisé ici comme référence et point de départ pour nos extensions. Nous considérons qu'il y a un ensemble de nœuds étiquetés et non étiquetés disponible. Le but est d'attribuer un score à tous les

nœuds non étiquetés en utilisant toute l'information disponible sur à la fois les nœuds étiquetés et non étiquetés. Cela est fait en deux étapes :

– La première étape consiste à entraîner un classifieur classique sur le contenu des nœuds étiquetés. Ce classifieur sur le contenu seul peut être n'importe quel modèle classique comme un perceptron, un SVM ou un modèle génératif. Une fois entraîné sur les nœuds étiquetés, il sera utilisé pour calculer un score initial, $\bar{y_i}$, pour tous les nœuds non étiquetés du graphe. À la fin de cette première étape, tous les nœuds dans le graphe auront un score initial.

– La seconde étape consiste à propager ces scores le long des arcs du graphe sous la contrainte que la valeur propagée reste relativement proche du score initial calculé dans l'étape 1. La propagation va régulariser les scores de manière à ce que les voisins dans le graphe aient des scores proches.

À la fin du processus, le score final d'un nœud sera un compromis entre son score de contenu seul et le score de ses voisins. Nous détaillons ci-dessous les deux étapes puis nous présentons à la section 4 un modèle capable d'apprendre comment propager les étiquettes sur différents types de relations.

Notons que dans notre contexte transductif, nous considérons que les scores des nœuds étiquetés sont connus durant tout le processus. Si les nœuds du graphe n'ont pas de contenu, cet algorithme peut toujours être utilisé en propageant simplement les étiquettes (étape 2 dans 3.1) sans calculer le score intermediaire $\bar{y_i}$ (étape 1 dans 3.1).

3.2. Contexte classique : étiquetage à partir du contenu seul

La première étape de l'algorithme est l'apprentissage inductif classique. Un classifieur est entraîné en utilisant les nœuds étiquetés pour classifier les nœuds non étiquetés. Dénotons θ les paramètres du classifieur et f_θ^{co} la fonction de classification correspondante. La méthode classique consiste à minimiser un risque empirique défini sur les nœuds étiquetés tel que :

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

où $\Delta^{co}(f_\theta^{co}(v_k), y_k)$ est le coût de l'erreur consistant à prédire $f_\theta^{co}(v_k)$ au lieu de y_k et λ un hyperparamètre de régularisation L2.

Différentes fonctions f^{co} peuvent être utilisées ici. Dans nos expériences, nous avons utilisé une fonction linéaire avec un « hinge loss » minimisé par une descente de gradient (cf. 5).

Le score du nœud v_i prédit par le modèle sur le contenu seul est $\bar{y_i} = f_\theta^{co}(v_i)$. Quand les nœuds du graphe n'ont pas d'information de contenu, ces scores peuvent être tirés au hasard.

3.3. Propagation par la régularisation dans les graphes monorelationnels

Les modèles de propagation classiques ont été développés pour les graphes monorelationnels, où $R = 1$. La plupart des modèles (Zhou *et al.*, 2004 ; Belkin *et al.*, 2006), reposent sur une hypothèse de régularité qui considère que deux nœuds connectés devraient avoir des étiquettes similaires. Cette contrainte est habituellement implémentée par un terme de régularisation. La fonction de coût pour ces modèles a la forme générale :

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

où $\hat{y}_i$ est le score prédit du nœud v_i et α est un paramètre de régularisation. Ici $w_{i,j}^{(1)}$ est le poids de l'arc entre v_i et v_j dans un graphe monorelationnel. *Terme 1* mesure l'erreur entre le score prédit et le score de contenu seul sur les nœuds non étiquetés. Ce terme agit comme une contrainte pour que $\hat{y}_i$ reste proche de $\bar{y_i}$. C'est un changement par rapport à la formulation classique des modèles de régularisation transductive sur les graphes qui ne considèrent que la propagation (*i.e.* terme 2) et pas le classifieur de contenu. *Terme 2* encourage la régularité sur le graphe. Le compromis entre les deux termes est réglé par α et nous permet de contrôler le degré de régularité sur le graphe. Une faible valeur de α encourage le modèle à fournir des scores finaux proches des scores du contenu seul, une forte valeur de α favorise plus de régularité sur le graphe. Les scores finaux des nœuds sont obtenus par la minimisation de cette fonction :

$$\hat{y}^*_{N^\ell+1}, ..., \hat{y}^*_N = \underset{\hat{y}_{N^\ell+1} \cdots \hat{y}_N}{\operatorname{argmin}} L^{reg}(\hat{y}_{N^\ell+1} ... \hat{y}_N) \tag{3}$$

Différentes variantes peuvent être utilisées pour les termes dans l'équation 2.

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 présenterons une technique originale à la section 4, qui nous permettra d'entraîner notre modèle multirelationnel.

4. Propagation d'étiquettes avec des relations multiples

4.1. Modèle multirelationnel

Nous décrivons un nouveau modèle capable d'apprendre à étiqueter dans un contexte multirelationnel. Ce modèle apprend une combinaison linéaire « optimale » des différents poids de relations. Il exploite un mécanisme d'inférence spécifique.

Nous introduisons la fonction de coût à la section 4.1.1, la technique d'inférence à la section 4.2 et l'algorithme d'apprentissage à la section 4.3.

4.1.1. Fonction de coût

Au lieu d'utiliser directement les poids des relations dans la fonction de coût comme dans l'équation 2 au travers du poids $w_{i,j}^{(1)}$, nous utilisons une fonction paramétrisée $\psi_\gamma(i,j) \in [0;1]$ définie sur chaque paire de nœuds v_i, v_j où γ est l'ensemble des paramètres de $\psi_\gamma(.,.)$. $\psi_\gamma(.,.)$ est désignée comme la *fonction sur les arcs* par la suite. Ses paramètres sont appris depuis les données comme décrit dans la section 4.3. Cette fonction fournit des poids normalisés dans [0;1] pour les différents types de relations. La fonction du coût utilisée dans le modèle est $L^{multi}()$:

$$\begin{aligned} L^{multi}(\hat{y}_1 ... \hat{y}_N) = \frac{1}{N^u} \sum_{k=N^\ell+1}^{N} (\hat{y}_k - \bar{y_k})^2 \qquad \text{(terme 1)} \\ + \alpha \sum_{v_i, v_j} \psi_\gamma(i,j)(\hat{y}_i - \hat{y}_j)^2 \quad \text{(terme 2)} \end{aligned} \tag{4}$$

Notons qu'ici encore, la somme dans le terme 1 est faite sur les nœuds non étiquetés ce qui nous permet d'inférer à la fois sur le contenu et les relations.

$L^{multi}(\hat{y}_1, ... \hat{y}_N)$ a la même forme que $L^{reg}(\hat{y}_1, ... \hat{y}_N)$ excepté que $w_{i,j}^{(1)}$ a été remplacé par la fonction $\psi_\gamma(i,j)$. Décrivons désormais cette fonction.

4.1.2. Fonction sur les arcs dans les multigraphes

Soit $\Phi(i,j)$ un vecteur de caractéristiques décrivant les différentes relations entre v_i et v_j :

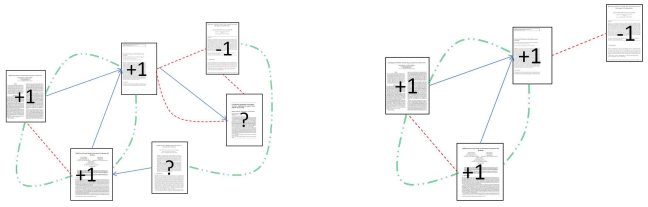
$$\Phi(i,j) = \begin{pmatrix} w_{i,j}^{(1)} \\ \vdots \\ w_{i,j}^{(R)} \end{pmatrix} \tag{5}$$

Nous choisissons pour la fonction sur les arcs une fonction logistique sur le vecteur Φ :

$$\psi_\gamma(i,j) = \text{logit} < \gamma; \Phi(i,j) > \quad = \text{logit} \sum_{r=1}^{R} \gamma_r . w_{i,j}^{(r)} \tag{6}$$

Ce choix considère que la fonction sur les arcs est définie avec un paramètre γ_r pour chaque type de relation. Ce paramètre représente l'importance de la relation r pour propager les scores sur les nœuds non étiquetés. Les paramètres γ sont appris

depuis les données comme expliqué à la section 4.3. Ce modèle est donc capable d'apprendre l'importance relative des différents types de relations pour la propagation sur le graphe. La fonction logistique est utilisée ici pour deux raisons. Tout d'abord elle force le terme de régularisation à être positif, et elle empêche également les coefficients γ de diverger à l'infini et agit en pratique comme une contrainte de régularisation sur les γ. Si une simple combinaison linéaire avait été utilisée au lieu d'une fonction logistique, une des relations serait devenue très dominante à travers son coefficient. Cela arrive rarement en pratique avec notre formulation. Différentes variantes de cette fonction sur les arcs pourraient être définies. Par exemple, le contenu sur les nœuds pourrait également être incorporé dans l'expression de $\psi_\gamma(i,j)$. Ce point n'est pas discuté ici.



Problème initial Sous-graphe utilisé pour l'entraînement

Figure 2. Sous-graphe étiqueté utilisé pour entraîner les modèles

4.2. Inférence

L'inférence consiste à calculer les scores prédits $\hat{y}^*_{N^\ell+1}, ...\hat{y}^*_N$ sur les nœuds non étiquetés de manière à minimiser L^{multi} :

$$\hat{y}^*_{N^\ell+1}, ...\hat{y}^*_N = \underset{\hat{y}_{N^\ell+1}, ...\hat{y}_N}{\operatorname{argmin}} L^{multi}(\hat{y}_{N^\ell+1}, ...\hat{y}_N) \tag{7}$$

Notons que cette fonction considère uniquement les nœuds non étiquetés, puisque sur les nœuds étiquetés nous avons posé par définition $\hat{y}_i = y_i$.

Nous proposons de résoudre la minimisation de cette fonction par un algorithme itératif basé sur une méthode de descente de coordonnées. Cette méthode consiste à minimiser la fonction L^{multi} coordonnée par coordonnée. Comme $L^{multi}()$ est une fonction convexe par rapport à $(\hat{y}_{N^\ell+1}, ...\hat{y}_N)$, l'algorithme convergera au minimum de cette fonction. L'algorithme est décrit dans l'algorithme 1 étape 3. À l'itération t, on minimise la fonction sur une coordonnée particulière k sachant les scores prédits sur tous les autres nœuds $\hat{y}^{(t)}_1, ..., \hat{y}^{(t)}_{k-1}, \hat{y}^{(t)}_{k+1}, ..., \hat{y}^{(t)}_N$. Le minimum est obtenu en résolvant l'équation suivante :

$$\hat{y}^{(t+1)}_k = \underset{\hat{y}_k}{\operatorname{argmin}}\, L^{multi}(\hat{y}^{(t)}_1, ..., \hat{y}^{(t)}_{k-1}, \hat{y}_k, \hat{y}^{(t)}_{k+1}, ..., \hat{y}^{(t)}_N) \tag{8}$$

Algorithme 1 Modèle transductif multirelationnel

Etape 1. Entraîner un classifieur sur le contenu {cela peut être un perceptron, un SVM, un modèle génératif, etc.}
Entrée : Contenu et étiquettes des nœuds étiquetés: $V^\ell, \{y_1, .., y_{N^\ell}\}$.
Sortie : Score de contenu pour tous les nœuds du graphe : $\{\bar{y_1}, ..., \bar{y_N}\}$
Pour les nœuds étiquetés $\bar{y} = y$

Etape 2. Apprentissage de la fonction sur les arcs
Entrée : Sous-graphe étiqueté initial (voir figure 2)
Sortie : Paramètres de structure γ

Minimiser $\frac{1}{N^\ell} \sum_{k=1}^{k=N^\ell} \left(\frac{\frac{\bar{y_k}}{N^\ell} + \alpha \left(\sum_{v_i \in V^\ell} (\psi_\gamma(k,i) + \psi_\gamma(i,k)) y_i \right)}{\frac{1}{N^\ell} + \alpha \left(\sum_{v_i \in V^\ell} (\psi_\gamma(k,i) + \psi_\gamma(i,k)) \right)} - y_k \right)^2$ sur γ.

Etape 3. Inférence
Entrée : Graphe complet $G = (V, E)$
Sortie : Scores finaux $\{\hat{y_1}, ..., \hat{y_N}\}$
pour tout nœud étiqueté $v_i \in V^\ell$ **faire**
  $\hat{y}_i \leftarrow y_i$ {Initialisation des nœuds étiquetés}
fin pour

pour tout nœud non étiqueté $v_i \in V^u$ **faire**
  $\hat{y}_i \leftarrow \bar{y_i}$ {Initialisation avec le modèle de contenu seul}
fin pour

répéter
  pour tout nœud non étiqueté $v_k \in V^u$ **faire**

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

  fin pour
jusqu'à convergence

Cette minimisation est faite uniquement sur les nœuds non étiquetés et obtenue quand :

$$\frac{\partial L^{multi}(\hat{y}^{(t)}_{N^\ell+1}, ..., \hat{y}^{(t)}_{k-1}, \hat{y}_k, \hat{y}^{(t)}_{k+1}, ..., \hat{y}^{(t)}_N)}{\partial \hat{y}_k} = 0 \quad (9)$$

La solution est donc :

$$\hat{y}_k^{(t+1)} = \frac{\frac{1}{N^u}\bar{y_k} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\hat{y}_i\right)}{\frac{1}{N^u} + \alpha\left(\sum_{v_i}(\psi_\gamma(k,i) + \psi_\gamma(k,i))\right)} \tag{10}$$

À chaque étape, la nouvelle valeur $\hat{y}_k^{(t+1)}$ est une combinaison pondérée du score sur le contenu $\bar{y_k}$ et des valeurs $\hat{y}_i$ des voisins. Ainsi, les étiquettes se propagent à travers les relations avec prise en compte du score sur le contenu. Notons que l'algorithme utilise une formulation de graphe dirigé (les deux arcs (k,i) et (i,k) apparaissent dans l'expression), de manière à ce que des graphes dirigés puissent être utilisés. L'algorithme consiste à itérer l'équation 10 pour chaque nœud non étiqueté v_k jusqu'à convergence. Il est décrit dans l'algorithme 1.

4.3. Apprentissage des paramètres de structure

Le but est ici d'apprendre les paramètres γ de la fonction sur les arcs. Nous considérons que les paramètres sur le contenu seul ont été précédemment appris comme expliqué à la section 3.2. Définissons $\Delta(\hat{y}_i, y_i)$, le coût de prédiction du score $\hat{y}_i$ au lieu de y_i en utilisant la procédure d'inférence proposée. Le coût empirique est défini sur les nœuds étiquetés comme étant :

$$\forall v_k \in V^\ell, E(\gamma) = \frac{1}{N^\ell}\sum_{i=1}^{N^\ell}\Delta_\gamma(\hat{y}_k^*; y_k) \tag{11}$$

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

$$\hat{y}_1^*, ...\hat{y}_{N^\ell}^* = \underset{\hat{y}_1,...\hat{y}_{N^\ell}}{\operatorname{argmin}} L^{multi}(\hat{y}_1, ...\hat{y}_{N^\ell}) \tag{12}$$

Les paramètres de structure γ sont appris sur les données étiquetées. La figure 4.1.2 illustre le sous-graphe étiqueté extrait du graphe (étiqueté + non étiqueté). L'idée sous-jacente est de découvrir, pour chaque nœud, comment les étiquettes des voisins sont propagées sur les différents types de relations. Pour le sous-graphe étiqueté, nous générons un exemple d'apprentissage pour chaque nœud : il est composé de ce nœud et de ses voisins comme illustré à la figure 4.3. Nous calculons ensuite les paramètres γ sur ces exemples. Le principe de l'algorithme est de trouver pour les exemples d'apprentissage, les paramètres γ qui propageront correctement les étiquettes ($+1$ et -1) sur les nœuds non étiquetés comme expliqué dans l'équation 13.

Pour les nœuds étiquetés, nous voulons prédire des scores $\hat{y}_k^{(t+1)}$ aussi proches que possible du score réel y_k de v_k. $\hat{y}_k^{(t+1)}$, calculé par la procédure d'inférence, qui

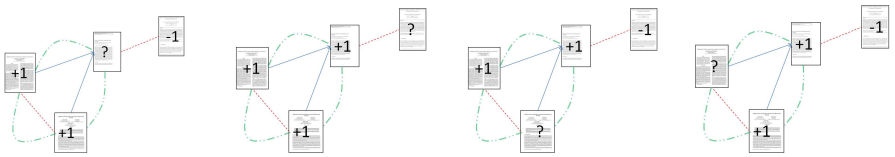


Figure 3. Différents exemples d'apprentissage extraits du sous-graphe étiqueté de la figure 2, utilisés pour apprendre les paramètres de structure

dépend des paramètres γ (équation 10). Nous optimisons les valeurs γ pour obtenir des scores prédits aussi proches que possible des scores réels. Apprendre γ consiste ensuite à minimiser le coût de prédiction Δ défini pour chaque nœud étiqueté v_k :

$$\Delta_\gamma(\hat{y}_k^*; y_k) \approx \Delta\left(\frac{\frac{1}{N^\ell}\bar{y_k} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))y_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\right)}, y_k\right) \tag{13}$$

Nous utilisons ici un coût d'erreur au carré classique :

$$\begin{aligned}\gamma^* &= \underset{\gamma}{\operatorname{argmin}} \frac{1}{N^\ell} \sum_{k=1}^{k=N^\ell} \Delta_\gamma(\hat{y}_k^*; y_k) \\ &= \underset{\gamma}{\operatorname{argmin}} \frac{1}{N^\ell} \sum_{k=1}^{k=N^\ell} \left(\frac{\frac{\bar{y_k}}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))y_i\right)}{\frac{1}{N^\ell} + \alpha\left(\sum_{v_i \in V^\ell}(\psi_\gamma(k,i) + \psi_\gamma(i,k))\right)} - y_k \right)^2\end{aligned} \tag{14}$$

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

5. Expériences

5.1. Corpora

Nous avons fait des expériences sur quatre jeux de données :

– Le corpus **Cora** est une base d'articles scientifiques avec 10 thématiques possibles. Le contenu de chaque nœud correspond au vecteur TF-IDF normalisé sur le résumé de l'article. Il y a 5 types de relations dans ce corpus :

- la relation *Cite* entre v_i et v_j vaut 1 si l'article i cite l'article j ;
- la relation *isCited* entre v_i et v_j vaut 1 si l'article j cite l'article i ;
- la relation *sameAuthor* vaut 1 si les deux articles ont un auteur en commun ;
- la relation *Title* est définie comme le produit scalaire entre les vecteurs TF-IDF normalisés représentant les titres des deux articles ;

- la relation *Abstract* est définie comme le produit scalaire entre les vecteurs TF-IDF normalisés représentant les résumés des deux articles.

Pour ce corpus, la tâche est de la classification multiclasse, mono étiquette.

– Le corpus **Flickr** (Peters *et al.*, 2010) correspond aux images et à l'information textuelle extraite de Flickr. Chaque image est représentée par une description textuelle sur Flickr et a été annotée avec différents tags. La tâche est ici l'étiquetage automatique, *i.e.* assigner un ou plusieurs tags à chaque image. Les images sont connectées par 3 types de relations différentes :

- la relation *Friendship* : deux images sont connectées avec le poids 1 si les auteurs sont amis ;

- la relation *Comment* : deux images sont connectées avec le poids 1 si le même utilisateur a commenté les deux images ;

- la relation *SameMonth* : deux images sont connectées avec le poids 1 si elles ont été publiées pendant la même période de 30 jours.

La tâche est ici de la classification multiclasse, multi étiquette. Des expériences ont été lancées sur une version avec 10 tags possibles et une autre version avec 50 tags possibles.

– Le corpus **Email** est une base d'emails extraits des emails des auteurs de cet article. Ces emails ont été classés manuellement dans 16 dossiers différents. Chaque email est décrit par son contenu textuel et les emails sont connectés par 4 types de relations :

- la relation *Authorship* : deux emails sont connectés avec le poids 1 s'ils ont le même expéditeur ;

- la relation *Recipient* : deux emails sont connectés avec le poids 1 s'ils ont le même destinataire ;

- la relation *Copy* : deux emails sont connectés avec le poids 1 s'ils ont le même destinataire en copie ;

- la relation *SameDay* : deux emails sont connectés avec le poids 1 s'ils ont été envoyés le même jour.

La tâche est ici de la classification multiclasse, mono étiquette.

Le tableau 1 donne quelques statistiques sur les corpora.

Tableau 1. Statistiques sur les corpora

Base	Nb. nœuds	Nb. d'arcs	Nb. relations	Nb. categories	Tâche
Cora	24 245	712 440	5	10	mono étiquette
Flickr10	1 995	316 002	3	10	multi étiquette
Flickr50	4 338	1 186 190	3	50	multi étiquette
Emails	4 408	$\approx$ 1 M	4	15	mono étiquette

Le modèle présenté calculant des scores sur les multigraphes binaires, nous apprenons un modèle pour chaque catégorie possible. Dans le cas de la classification mono étiquette, la catégorie avec le plus gros score est choisie. Dans le cas multi étiquette,

les scores de toutes les catégories sont utilisés pour calculer une précision @ n comme expliqué ci-dessous.

5.2. Modèles

Pour comparer notre approche avec les méthodes multi relationnelles existantes qui ont été développées uniquement pour les réseaux avec structure seule [1], nous avons lancé deux ensembles d'expériences :

– Les *Expériences sur la structure seule (SO)* ne font usage que de la structure sur les multigraphes, sans information de contenu. Pour ces expériences, nous avons utilisé trois modèles :

- **Le modèle monorelationnel (SO-Mono)** qui correspond au modèle régularisé décrit à la section 3.3 où $\alpha = +\infty$. Ce modèle ne gère qu'un type de relation à la fois. Les expériences ont été lancées pour chaque type de relation, un à la fois.

- **Le modèle multirelationnel (SO-Multi)** qui a été présenté à la section 4 avec $\alpha = +\infty$ et qui nous permet d'utiliser simultanément les différents types de relations.

- **Le modèle de l'état de l'art (Kato)** proposé par Kato *et al.* (2008) qui trouve automatiquement un schéma de pondération pour les différents types de relations par un algorithme d'optimisation alternée (voir la section 6 pour plus de détails sur les différences entre cette approche et la notre).

– Les *Expériences sur le contenu et la structure (SC)* où les modèles peuvent utiliser à la fois le contenu et la structure des multigraphes. Ici, nous avons comparé trois différents modèles :

- **Le modèle monorelationnel (SC-Mono)** qui correspond au modèle régularisé décrit à la section 3.3.

- **Le modèle multirelationnel (SC-Multi)** qui a été présenté à la section 4.

- **Le modèle sur le contenu seul (SC-CO)** qui correspond au modèle présenté dans la partie 3.2. Notons qu'il est équivalent aux deux modèles précédents avec $\alpha = 0$.

5.3. Mesures d'évaluation

Pour les corpora multiclasse mono étiquette, les résultats sont évalués en utilisant les mesures classiques micro-F_1 (miF_1) et macro-F_1 (maF_1). La mesure F_1 pour une catégorie est la moyenne harmonique de la précision et du rappel. La macro-F_1 est la moyenne de la F_1 sur toutes les catégories du problème. Le micro-F_1 correspond à la moyenne pondérée par la taille de chaque catégorie. Pour les problèmes multiétiquettes, nous avons utilisé également les micro-F_1 et macro-F_1 en assignant

1. Ces méthodes utilisent une information éventuelle de contenu en calculant des relations de similarité.

une catégorie à un nœud si son score est positif. Pour évaluer la performance en ordonnancement, nous avons aussi calculé la précison à 1 (P@1) et à 3 (P@3) sur la liste ordonnée par score des étiquettes. Nous avons lancé les expériences sur diverses tailles en entraînement - 5 %, 10 %, 25 % et 50 % de nœuds étiquettés - et valeur de α - .01, 0.1, 1, 10, 100. Trois processus ont été lancés pour chaque expérience et les résultats présentés sont leur moyenne. Nous avons rapporté les meilleurs résultats en fonction de α.

5.4. Expériences sur la structure seule

Les expériences sur la structure seule ont été faites principalement pour pouvoir comparer notre approche avec les approches multirelationnelles de l'état de l'art. Les résultats sur les différents modèles sont présentés dans les tableaux 2 et 3. Nous fournissons également des courbes de performance montrant les résultats des différents modèles - cf. figures 4 et 5.

On peut observer que la performance sur chaque modèle monorelationnel dépend principalement de la relation utilisée. Par exemple, l'utilisation de la relation 3 sur Cora avec 10 % de nœuds en entraînement permet d'obtenir 33 % en micro-F_1 alors que la relation 1 n'atteint que 12 %. Cela est dû principalement au fait que l'hypothèse de régularité faite par les modèles monorelationnels n'améliore les performances que si le type de relation considéré respecte cette hypothèse - *i.e.* certains types de relations sont réguliers et d'autres pas. Le modèle **Kato** améliore clairement les performances par rapport au monorelationnel en obtenant par exemple 60 % en micro-F_1 pour la même taille en entraînement. En fonction du corpus et de la taille en entraînement, notre approche obtient en général des performances équivalentes ou meilleures que le modèle Kato, avec une amélioration nette quand le nombre de nœuds en entraînement n'est pas trop petit. Les figures 4 illustrent la précision à 1 des différents modèles sur les bases Flickr10 et Flickr50 démontrant la capacité de notre modèle à améliorer les performances par rapport aux algorithmes existants. L'exemple de Flickr démontre que les différentes relations sont réellement complémentaires et nous permettent d'éviter la variabilité et l'incertitude associés aux étiquettes pour ce corpus. Ces expériences démontrent que l'algorithme proposé apprend bien comment les étiquettes se propagent. Dans ce cas, le modèle multirelationnel est capable de tirer parti de l'information additionnelle apportée par les relations multiples et d'obtenir des performances significativement supérieures aux autres modèles.

Par ailleurs, comme le modèle Kato est basé sur un algorithme d'optimisation alternée [2], le temps passé en entraînement par notre méthode est environ 3 fois inférieur. Sur le corpus CORA avec 5% de nœuds en entraînement, notre approche met 2 minutes et 42 secondes pour inférer les étiquettes pour les nœuds non étiquetés alors que le modèle monorelationnel met 1 minute et 2 secondes pour une seule relation et le modèle Kato demande environ 10 minutes. Pour toutes les expériences, la complexité

2. Demande 3 itérations pour converger

*Tableau 2. Performance des différents modèles pour les expériences sur la structure seule sur Cora et Email. **SO-Mono** n correspond au modèle SO-Mono utilisant la relation n. Les meilleures performances figurent en gras. '-' correspond aux relations qui n'existent pas dans la base concernée*

Modèle	Entraînement	Cora Corpus		Email Corpus	
		miF_1	maF_1	miF_1	maF_1
SO-Mono 1	5 %	10.71	4.96	45.22	19.25
	10 %	12.02	5.62	51.52	25.38
	25 %	13.44	7.50	59.48	32.67
	50 %	14.12	8.14	63.60	35.41
SO-Mono 2	5 %	8.40	3.09	44.02	18.08
	10 %	9.18	4.57	46.01	20.05
	25 %	10.61	6.71	51.55	22.85
	50 %	12.64	9.65	54.57	25.57
SO-Mono 3	5 %	29.44	25.50	9.14	5.94
	10 %	33.41	30.66	10.15	6.94
	25 %	40.56	38.28	11.80	8.73
	50 %	50.93	48.61	12.29	8.83
SO-Mono 4	5 %	19.01	18.21	8.46	6.10
	10 %	22.76	23.47	11.49	11.72
	25 %	28.09	29.82	19.65	17.27
	50 %	25.08	36.47	28.49	22.94
SO-Mono 5	5 %	24.10	19.23	-	-
	10 %	26.92	22.14	-	-
	25 %	31.60	28.04	-	-
	50 %	35.17	32.03	-	-
Kato	5 %	57.02	46.12	63.10	25.14
	10 %	60.72	53.50	64.74	28.42
	25 %	**65.33**	59.99	70.85	**35.66**
	50 %	**68.39**	**64.61**	73.71	38.78
SO-Multi	5 %	**57.77**	**46.57**	**63.58**	**26.26**
	10 %	**61.51**	**53.88**	**68.45**	**32.42**
	25 %	64.51	**60.91**	**72.45**	34.20
	50 %	65.64	60.75	**77.89**	**43.63**

*Tableau 3. Performance des différents modèles pour les expériences sur la structure seule sur Flickr10 et Flickr50. **SO-Mono** n correspond au modèle SO-Mono utilisant la relation n. Les meilleures performances figurent en gras. '-' correspond aux relations qui n'existent pas dans la base concernée*

Modèle	Entraînement	Flickr 10 Corpus				Flickr 50 Corpus			
		miF_1	maF_1	P@1	P@3	miF_1	maF_1	P@1	P@3
SO-Mono 1	5 %	2.98	2.77	19.59	15.54	1.34	1.29	6.92	6.98
	10 %	2.48	2.32	21.40	16.09	0.86	0.81	7.61	7.57
	25 %	4.92	4.47	22.44	16.44	1.94	1.92	8.27	7.86
	50 %	6.78	6.31	24.70	17.85	3.87	3.70	10.00	8.94
SO-Mono 2	5 %	14.79	**14.24**	22.16	16.86	3.96	2.26	10.47	8.41
	10 %	14.61	12.11	21.34	16.93	2.27	1.54	12.66	9.68
	25 %	10.79	8.73	25.59	17.80	1.71	1.09	15.68	11.49
	50 %	12.60	7.70	26.66	18.45	0.27	0.31	17.03	12.74
SO-Mono 3	5 %	**16.45**	12.69	22.75	17.89	4.86	2.46	13.82	10.27
	10 %	15.74	11.80	25.89	19.47	1.40	1.27	13.99	11.81
	25 %	17.63	13.73	27.47	20.60	2.62	2.37	14.81	11.24
	50 %	19.32	16.27	29.77	20.69	8.84	6.39	14.72	10.36
Kato	5 %	12.06	7.39	26.89	18.46	2.45	1.24	26.02	**18.26**
	10 %	8.85	5.70	28.22	20.48	1.17	1.06	27.96	18.86
	25 %	11.72	8.03	31.32	21.91	2.53	1.79	35.18	**24.23**
	50 %	13.25	9.18	33.91	23.03	8.90	5.94	**44.77**	**27.73**
SO-Multi	5 %	8.02	6.30	**30.28**	**21.88**	**6.51**	**5.03**	**26.29**	17.52
	10 %	**20.04**	**15.23**	**37.06**	**24.43**	**15.17**	**11.49**	**33.60**	**20.04**
	25 %	**29.14**	**24.89**	**43.13**	**24.19**	**29.80**	**22.16**	**40.82**	23.88
	50 %	**36.93**	**33.64**	**49.75**	**26.84**	**34.72**	**27.40**	43.58	24.54

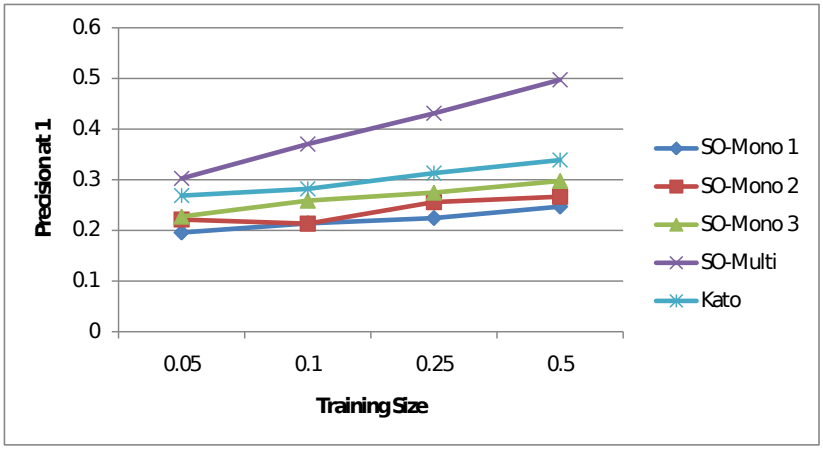


Figure 4. Flickr 10

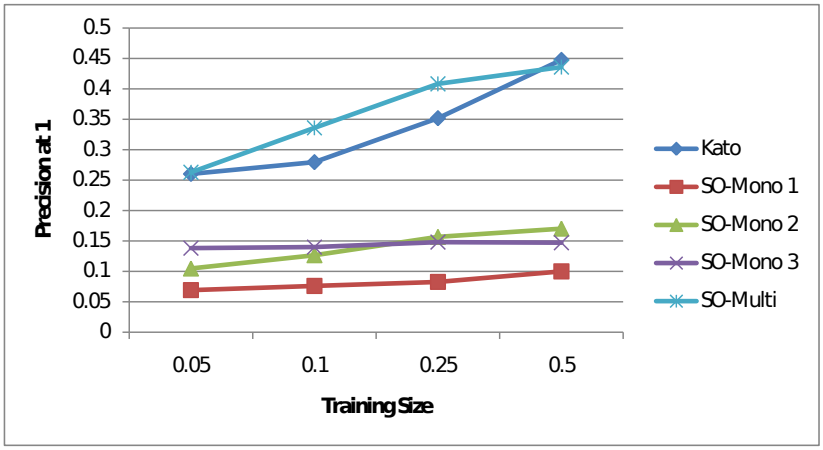


Figure 5. Flickr 50

des modèles mono et multirelationnels respecte la même proportion. Pour CORA, l'entraînement d'un seul modèle multirelationnel est plus rapide que l'entraînement séparé de 5 modèles monorelationnels et que la méthode Kato de l'état de l'art.

5.5. Expériences sur le contenu et la structure

Les résultats des expériences sur le contenu et la structure sont rapportés dans le tableau 4 pour les bases Cora et Email. Les résultats pour Flickr ne sont pas rapportés car le contenu de cette base n'est pas assez informatif pour faire de la classification

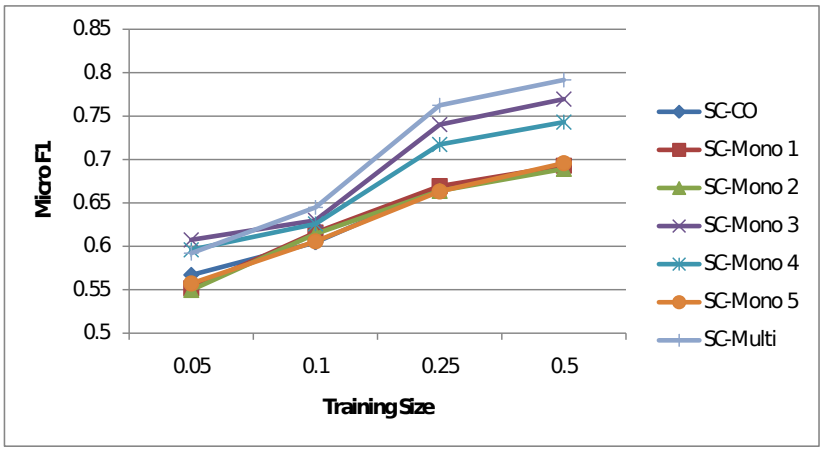


Figure 6. Cora

*Tableau 4. Performance des différents modèles sur les expériences sur le contenu et la structure. Les modèles **SC-Mono** n correspondent au modèle SC-Mono utilisant la relation n. Les meilleures performances sont en gras*

Corpus	Entraînement	Cora Corpus		Email Corpus	
		miF_1	maF_1	miF_1	maF_1
SC-Mono 1	5 %	55.17	44.30	**64.31**	**29.37**
	10 %	61.59	52.49	**70.74**	**33.13**
	25 %	66.94	59.09	74.21	**37.73**
	50 %	69.29	62.06	77.92	43.43
SC-Mono 2	5 %	54.97	44.32	57.07	23.47
	10 %	61.43	52.36	62.31	28.16
	25 %	66.38	58.56	67.99	33.66
	50 %	68.91	61.34	71.01	36.38
SC-Mono 3	5 %	**60.74**	47.94	49.28	20.45
	10 %	62.99	52.93	55.99	24.67
	25 %	74.02	66.64	61.49	29.66
	50 %	76.96	70.77	65.72	32.92
SC-Mono 4	5 %	59.60	**48.67**	54.09	23.99
	10 %	62.58	52.70	58.88	26.68
	25 %	71.73	64.70	64.29	32.06
	50 %	74.73	67.91	67.15	35.85
SC-Mono 5	5 %	55.72	43.97	-	-
	10 %	60.57	50.56	-	-
	25 %	66.32	58.64	-	-
	50 %	69.58	62.33	-	-
SC-CO	5 %	56.70	44.18	50.32	22.48
	10 %	60.44	51.22	56.03	24.19
	25 %	66.70	59.41	60.90	28.75
	50 %	69.26	62.52	64.91	33.64
SC-Multi	5 %	59.21	46.55	61.82	23.95
	10 %	**64.49**	**55.30**	66.92	25.52
	25 %	**76.24**	**69.44**	**75.90**	35.25
	50 %	**79.16**	**74.05**	**79.79**	**43.58**

sur le contenu seul [3] et les techniques de classification d'images ont donné des performances proches de l'aléatoire. À partir de ce tableau, on peut voir que l'utilisation simultanée du contenu et de la structure améliore grandement la performance pour toutes les approches. La figure 6 montre que sur Cora, SC-Multi est meilleur que tous les modèles SC-Mono à l'exception d'une taille en entraînement de 5 % où il est légèrement moins bon que le meilleur des modèles SC-Mono (SC-Mono 3). Par exemple, le modèle SC-Mono 3 sur Cora - taille en entraînement = 25 % - a une micro-F_1 de 74 % alors que la version en structure seule n'obtient que 40 %. Dans ce cadre, notre méthode obtient de meilleures performances que le meilleur modèle monorelationnel démontrant la capacité du modèle SC-Multi à associer l'information contenue dans les différents types de relations, tout en agrégeant cette information avec celle du contenu des nœuds. L'augmentation de l'écart de performances avec la taille en entraînement est assez intuitive, le modèle multirelationnel étant plus général il demande plus de données pour trouver les meilleurs paramètres et éviter le surapprentissage. La performance varie grandement selon les corpus. Pour le corpus Email, le modèle multirelationnel est équivalent au meilleur modèle monorelationnel. Cela est dû au fait que la relation *Authorship* (SC-Mono 1) porte vraisemblablement la totalité de l'information relationnelle, ce qui en fait la borne supérieure potentiellement atteignable par notre algorithme.

6. État de l'art

6.1. Classification sur les graphes

La plupart des travaux dans ce domaine concernent l'apprentissage dans les graphes monorelationnels. Il y a deux perspectives principales sur ce problème.

L'approche transductive, qui est celle suivie dans cet article, a été motivée initialement par l'apprentissage semi-supervisé et initiée par Zhou (Zhou *et al.*, 2004) et Belkin (Belkin *et al.*, 2006). Tous ces modèles reposent sur la minimisation d'une fonction objectif combinant une fonction de coût d'erreur de prédiction classique sur les données étiquetées et un terme de régularisation sur les arcs du type $L = \sum_{i,j} \omega_{i,j}(f_i - f_j)^2$ où f_i est le score du nœud i et $\omega_{i,j}$ est le poids de l'arc entre i et j. Plusieurs variantes différentes de ces modèles existent et ce cadre a été utilisé dans beaucoup d'applications. Ces modèles font habituellement de la propagation sur un graphe de similarités. Il n'y a pas de « classifieur de contenu » comme dans le modèle introduit ici, le contenu n'apparaît qu'à travers le calcul de la fonction de similarité utilisée pour construire le graphe. Notons que (Zhou *et al.*, 2004) présente une relation intéressante entre les méthodes régularisées et une formulation de marche aléatoire dans le cas de la classification binaire. Ces modèles sont généralement développés pour des graphes non dirigés, mais des extensions pour les graphes dirigés ont été proposées (cf. (Zhou *et al.*,

3. Des tags qui apparaissent souvent sur Flickr sont le type d'appareil photo, ou des observations subjectives comme *beau*, *étrange*, etc. qui rendent la classification sur le contenu seul très difficile.

2005)). Le premier modèle général qui permet de combiner directement le contenu et l'information relationnelle est un article récent (Abernethy *et al.*, 2008). Au-delà de la classification, les modèles de graphe ont été utilisés pour l'ordonnancement (Agarwal, 2006) ou le réordonnancement (Qin *et al.*, 2008).

Dans le cas de l'approche inductive, les modèles de classification collective sont généralement utilisés. Ils peuvent utiliser à la fois l'information de contenu et de structure. La plupart comportent deux étapes. La première consiste à apprendre un classifieur de contenu sur les données étiquetées, sans considérer l'information relationnelle. Les scores obtenus sont utilisés pour fournir des étiquettes initiales aux nœuds. La seconde étape consiste à apprendre un classifieur relationnel opérant sur une description agrégée du voisinage du nœud dans le graphe. Pour cette seconde étape, l'apprentissage est également inductif, i.e. n'utilisant que l'information fournie par les nœuds étiquetés. Une fois le classifieur relationnel appris, il est utilisé pour étiqueter de nouveaux nœuds. Différents algorithmes ont été proposés pour ce problème d'étiquetage. Ces modèles incluent *Iterative Classification* (Sen *et al.*, 2008), *Gibbs Sampling* (Macskassy, Provost, 2003), *Stacked Learning* (Kou, Cohen, 2007) et SICA (Maes *et al.*, 2009). Ils utilisent une fonction de réestimation pour mettre à jour à chaque itération la probabilité d'étiqueter le nœud i avec l'étiquette j sachant les étiquettes des voisins. Ces modèles sont plus rapides que ceux de la famille régularisée et permettent de traiter de très grands graphes.

6.2. Multigraphes

Peu de travaux traitent de données multirelationnelles (ou multigraphes). (Chen *et al.*, 2009) utilise une combinaison pondérée de noyaux sur les relations. Cette approche produit un graphe fortement connexe, i.e. une complexité de l'algorithme d'apprentissage en n^2 qui est trop importante pour un usage sur des données réelles. (Bhagat *et al.*, 2007) considère l'étiquetage de blogs dans un contexte multigraphe où les poids sont uniformes et pas appris. Zhou *et al.* (2007) étend les idées introduites dans (Zhou *et al.*, 2005) aux multigraphes. Ils développent une méthode de clustering dans un hypergraphe en utilisant une forme de clustering spectral et en en dérivant une méthode de classification. Ici encore, la complexité pose problème. Dans un contexte inductif, différent de celui considéré ici, Peters *et al.* (2010) proposent une méthode de classification collective pour l'annotation dans des réseaux multirelationnels.

Kato *et al.* (2008) proposent un algorithme simple du type Expectation-Maximization(EM) auquel nous avons comparé notre méthode dans nos expériences. Ce modèle apprend à pondérer les différents types de relations en alternant l'optimisation sur les scores des nœuds et le poids des types de relation. Cet algorithme consiste, en prenant des poids aléatoires au départ, à utiliser ces poids pour propager les étiquettes. Une fois les étiquettes propagées, le modèle mesure la régularité obtenue par chaque relation sur le graphe et met à jour les poids en augmentant les poids des relations régulières et en diminuant les poids des relations non régulières. Ils fournissent également une interprétation probabiliste bayésienne de leur modèle.

Wang *et al.* (2009) proposent un modèle très proche de la régularisation sur les multigraphes dans un contexte différent (annotation de vidéo). Ici encore, une méthode alternée du type EM est utilisée pour apprendre une combinaison linéaire de laplaciens de graphes.

L'hypothèse faite et l'algorithme que nous proposons dans cet article diffèrent de ceux de ces deux méthodes. Une différence importante est que notre modèle emploie directement durant l'entraînement et l'inférence à la fois un classifieur sur le contenu et la propagation à travers les relations. Par ailleurs nous n'avons pas besoin d'avoir une connaissance a priori sur la forme de la distribution des poids des types de relations, connaissance qui est nécessaire dans les modèles de type EM. Nous n'avons donc pas besoin de faire une recherche coûteuse d'hyperparamètres (avec de la validation croisée par exemple). Par ailleurs l'absence d'optimisation alternée fait que notre méthode est plus rapide que celles de l'état de l'art. Une troisième différence est que notre modèle est capable de traiter à la fois les graphes orientés et non orientés alors que les modèles existants se limitent au cas non orienté. Les expériences démontrent sur nos corpora que la plupart du temps, la méthode proposée est aussi bonne ou meilleure que les méthodes de l'état de l'art basées sur de l'optimisation de type EM (c'est particulièrement visible sur Flickr10).

7. Conclusion

Nous avons proposé un modèle pour la classification multiclasse, multiétiquette dans les multigraphes. Ce modèle repose sur de nouveaux algorithmes d'apprentissage et procédure d'inférence. Il inclut comme cas particuliers différents modèles monorelationnels existants. Par ailleurs, en comparaison des modèles multirelationnels existants, il considère à la fois le contenu et la structure des multigraphes et l'apprentissage des poids des différents types de relations est fait en une seule étape. Les expériences faites sur différents jeux de données démontrent la capacité du modèle à extraire l'information pertinente de l'information relationnelle riche et obtient des performances supérieures par rapport aux modèles existants - à la fois monorelationnels et multirelationnels. Les perspectives résident dans différentes extensions de ce modèle et une analyse en profondeur de son comportement. En particulier, le modèle doit être amélioré pour une utilisation sur les très grands graphes, avec un très grand nombre de classes. Une autre direction de recherche consiste en l'application de cet algorithme aux réseaux hétérogènes et dynamiques.

Bibliographie

Abernethy J., Chapelle O., Castillo C. (2008). Witch: A new approach to web spam detection. In *Adversarial information retrieval on the web workshop*. Citeseer.

Agarwal S. (2006). Ranking on graph data. In *International conference on machine learning*, p. 25–32. New York, USA, ACM.

Belkin M., Niyogi P., Sindhwani V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, vol. 7, p. 2399–2434.

Bhagat S., Cormode G., Rozenbaum I. (2007). Applying link-based classification to label blogs. In *Webkdd/sna-kdd*, p. 97-117.

Bilgic M., Namata G., Getoor L. (2007). Combining collective classification and link prediction. In *International conference on data mining workshops*, p. 381-386.

Cao L., Luo J., Huang T. (2008). Annotating photo collections by label propagation according to multiple similarity cues. In *Multimedia*, p. 121–130. New York, USA, ACM.

Chen H., Li L., Peng J. (2009, mai). Error bounds of multi-graph regularized semi-supervised classification. *Information Sciences*, vol. 179, n° 12, p. 1960–1969.

Kato T., Kashima H., Sugiyama M. (2008). Integration of multiple networks for robust label propagation. In *Siam conference on data mining*, p. 716–726. Citeseer.

Kou Z., Cohen W. (2007). Stacked graphical models for efficient inference in markov random fields. In *Siam conference on data mining*, p. 533–538. Citeseer.

Macskassy S., Provost F. (2003). A simple relational classifier. In *Proc. of the 2nd workshop on multi-relational data mining*, p. 64–76. Citeseer.

Maes F., Peters S., Denoyer L., Gallinari P. (2009). Simulated iterative classification a new learning procedure for graph labeling. In *European conference on machine learning and principles and practice of knowledge discovery in databases*, p. 47-62.

Peters S., Denoyer L., Gallinari P. (2010). Iterative Annotation of Multi-relational Social Networks. In *Advances in social network analysis and mining*, p. 96–103. IEEE.

Qin T., Liu T.-Y., Zhang X.-D., Wang D.-S., Xiong W.-Y., Li H. (2008). Learning to rank relational objects and its application to web search. In *World wide web*, p. 407-416.

Sen P., Namata G., Bilgic M., Getoor L., Gallagher B., Eliassi-Rad T. (2008). Collective classification in network data. *AI Magazine*, vol. 29, n° 3, p. 93-106.

Tsuda K., Shin H., Schölkopf B. (2005, septembre). Fast protein classification with multiple networks. *Bioinformatics*, vol. 21, n° suppl 2, p. ii59.

Wang M., Hua X.-S., Hong R., Tang J., Qi G.-J., Song Y. (2009, may). Unified video annotation via multigraph learning. *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 19, n° 5, p. 733 -746.

Zhang T., Popescul A., Dom B. (2006). Linear prediction models with graph regularization for web-page categorization. In *Knowledge discovery and data mining*, p. 821–826. ACM.

Zhou D., Bousquet O., Lal T., Weston J., Schölkopf B. (2004). Learning with local and global consistency. In *Neural information processing systems*, vol. 1, p. 595–602.

Zhou D., Huang J., Schölkopf B. (2005). Learning from labeled and unlabeled data on a directed graph. In *International conference on machine learning*, p. 1036–1043. ACM.

Zhou D., Huang J., Scholkopf B. (2007). Learning with hypergraphs: Clustering, classification, and embedding. In *Neural information processing systems*, p. 1601-1608.