

Fouilles de données et Medias sociaux

Master 2 DAC - FDMS

Sylvain Lamprier

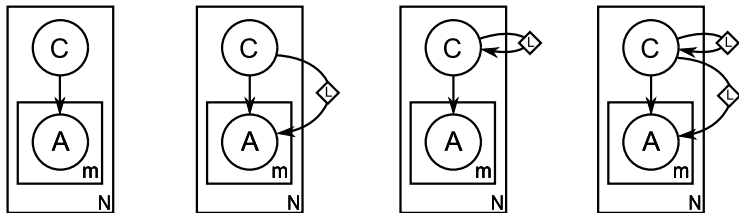
UPMC

Classification / Etiquetage dans les Réseaux

- Classification classique
 - Hypothèse de données i.i.d.
 - Corrélations entre les labels des objets et leurs attributs observés
 - Dans les réseaux:
 - Corrélations entre les labels des objets et leurs attributs observés ;
 - Corrélations entre les labels des objets et les attributs des objets de leur entourage (objets qui leur sont connectés) ;
 - Corrélations entre les labels des objets interconnectés.
- ⇒ Abandon de l'hypothèse i.i.d.

Classification / Etiquetage dans les Réseaux

Modèles relationnels et inférence collective :



• De gauche à droite :

- Modèle intrinsèque : les attributs d'un élément ne dépendent que de sa classe;
- Modèle relationnel : les attributs d'un élément dépendent de la classe de cet élément et de celles des éléments connectés (distants d'au maximum L liens) ;
- Modèle collectif : les attributs d'un élément dépendent de sa classe, elle même dépendante des classes des éléments connectés ;
- Modèle relationnel collectif : les attributs d'un élément dépendent de sa classe et de celles de son entourage, les labels de ces objets étant eux-mêmes interdépendants.

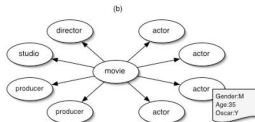
Classification dans les réseaux

- Modèles relationnels

- Initialement prévus pour modéliser des objets relationnels

Receipts >\$2mil	Genre	Rating	Length	Release
+	Drama	PG	114m	Dec
+	Comedy	PG13	98m	May
-	Drama	R	169m	Sept
+	Action	R	105m	July
...

Contenu



Relations

(c)					
Receipts ≥\$2mil	Actor Gender	Actor Age	Actor Oscar	Studio Location	...
+	(F,M,F,F)	(29,35, 32,18)	(N,N,N,Y)	(CA)	...
+	(M,M,F)	(31,45, 24)	(N,Y,N)	(NY)	...
-	(F,F)	(52,50)	(N,N)	(CA)	...
+	(M,M,F,F)	(24,27, 18,21)	(N,N,N,N)	(Canada)	...
...

Attributs des relations

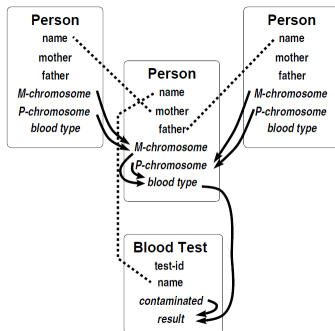
⇒ Comment agréger les attributs relationnels ?

⇒ Comment représenter les dépendances entre les attributs ?

- Modèles relationnels

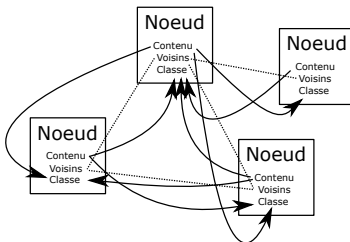
- Dépendances probabilistes entre attributs d'objets relationnels

⇒ Réseau Bayésien



$$P(\mathcal{I} \mid \sigma, \mathcal{S}, \theta_S) = \prod_{X_i} \prod_{A \in \mathcal{A}(X_i)} \prod_{x \in \mathcal{O}^\sigma(X_i)} P(\mathcal{I}_{x.a} \mid \mathcal{I}_{\text{Pa}(x.a)})$$

- Modèles relationnels: application aux réseaux
 - Dépendances probabilistes entre noeuds
- ⇒ Réseau Bayésien



- Modèle relationnel probabiliste (PRM) pour la classification dans les réseaux [McDowell & Aha, 2013]
 - Extension de Naive Bayes aux objets relationnels
 - Hypothèse d'indépendance des attributs \vec{x}_i de i et de ceux de son voisinage $X_{\mathcal{N}_i}$ conditionnellement à sa classe y_i :

$$\begin{aligned} p(y_i | \vec{x}_i, X_{\mathcal{N}_i}) &= p(y_i) \frac{p(\vec{x}_i, X_{\mathcal{N}_i} | y_i)}{p(\vec{x}_i, X_{\mathcal{N}_i})} \\ &= p(y_i) \frac{p(\vec{x}_i | y_i)}{p(\vec{x}_i, X_{\mathcal{N}_i})} \prod_{v_j \in \mathcal{N}_i} p(\vec{x}_j | y_i) \\ &\propto p(y_i) p(\vec{x}_i | y_i) \prod_{v_j \in \mathcal{N}_i} p(\vec{x}_j | y_i) \end{aligned}$$

- Modèle relationnel probabiliste (PRM) pour la classification dans les réseaux [McDowell & Aha, 2013]
 - Extension de Naive Bayes aux objets relationnels
 - Hypothèse d'indépendance des attributs \vec{x}_i de i et de ceux de son voisinage $X_{\mathcal{N}_i}$ conditionnellement à sa classe y_i :

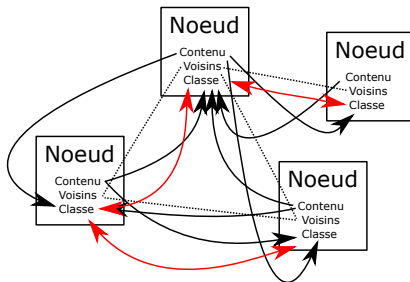
$$\begin{aligned} p(y_i | \vec{x}_i, X_{\mathcal{N}_i}) &= p(y_i) \frac{p(\vec{x}_i, X_{\mathcal{N}_i} | y_i)}{p(\vec{x}_i, X_{\mathcal{N}_i})} \\ &= p(y_i) \frac{p(\vec{x}_i | y_i)}{p(\vec{x}_i, X_{\mathcal{N}_i})} \prod_{v_j \in \mathcal{N}_i} p(\vec{x}_j | y_i) \\ &\propto p(y_i) p(\vec{x}_i | y_i) \prod_{v_j \in \mathcal{N}_i} p(\vec{x}_j | y_i) \end{aligned}$$

⇒ Mais si les attributs ne sont pas indépendants ?

Classification dans les réseaux

Dans les réseaux sociaux

⇒ Et si les classes des noeuds connectés ne sont pas indépendantes ?



Classification dans les réseaux: Modèles relationnels

- Représentation de dépendances non-dirigées

- Modèles graphiques :

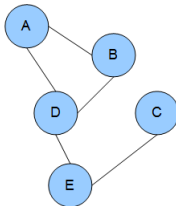
- Réseaux de Markov relationnels
(Relational Markov Random Fields)
 - Probabilité d'une instanciation X :

$$P(X = x) = \frac{1}{Z} \prod_k \phi_k(x_k)$$

Avec trois cliques ABD, DE et EC de potentiel ϕ

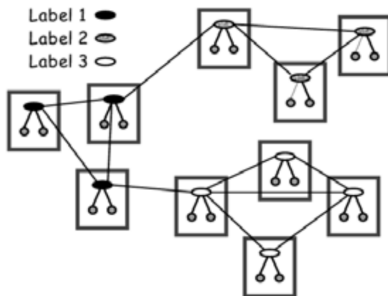
- Réseaux de dépendance relationnels
 - Probabilité d'une instanciation X :

$$\begin{aligned} P(X = x) &= P(A = x_A | x_B, x_D) \times P(B = x_B | x_A, x_D) \\ &\times P(C = x_C | x_E) \times P(D = x_D | x_A, x_B, x_E) \times P(x_E | x_D, x_C) \end{aligned}$$



Inférence Collective

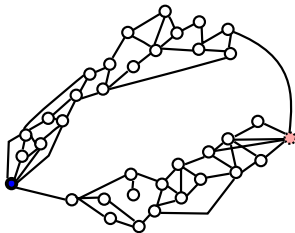
- Inférence Collective [Sen et al., 2008]
 - Processus itératif d'étiquetage de graphe
 - Dépendance entre les labels
 - Modèles transductifs généralement



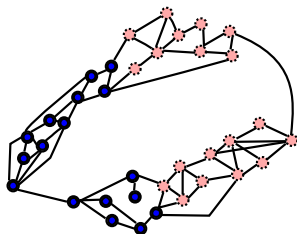
- Modèles d'inférence collective :
 - Iterative Classification (ICA)
 - Gibbs sampling
 - Loopy belief propagation
 - Modèle de Propagation de labels régularisé

Classification dans les Réseaux: Inférence collective

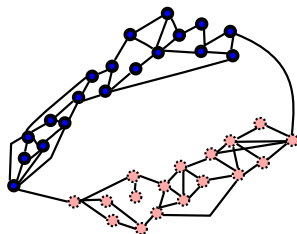
Graphe partiellement étiqueté
(deux étiquettes)



Graphe étiqueté
sans prise en compte de la structure



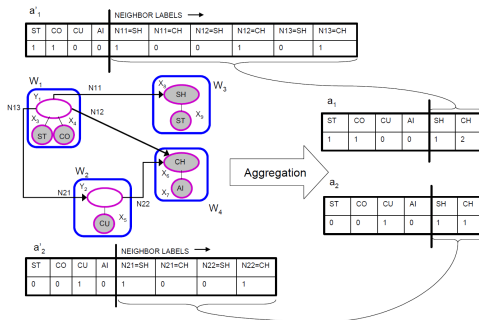
Graphe étiqueté
avec prise en compte de la structure



Inférence Collective:ICA

- Iterative Classification

- Aggrège les labels des noeuds connectés
 - Différents types d'aggrégation: Mode, Count, Proportion, etc...

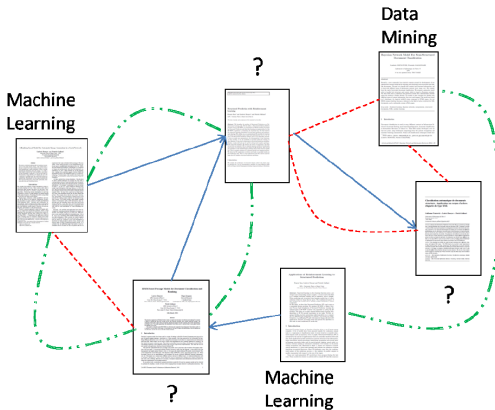


- Apprend un classifieur selon les labels connus
- Etiquetage des noeuds i selon leurs attributs et leur voisinage \mathcal{N}_i

Algorithm 1 Iterative Classification Algorithm (ICA)

```
for each node  $Y_i \in \mathcal{Y}$  do // bootstrapping
  // compute label using only observed nodes in  $\mathcal{N}_i$ 
  compute  $\vec{a}_i$  using only  $\mathcal{X} \cap \mathcal{N}_i$ 
   $y_i \leftarrow f(\vec{a}_i)$ 
end for
repeat // iterative classification
  generate ordering  $\mathcal{O}$  over nodes in  $\mathcal{Y}$ 
  for each node  $Y_i \in \mathcal{O}$  do
    // compute new estimate of  $y_i$ 
    compute  $\vec{a}_i$  using current assignments to  $\mathcal{N}_i$ 
     $y_i \leftarrow f(\vec{a}_i)$ 
  end for
until all class labels have stabilized or a threshold number of iterations have elapsed
```

Cas Multi-Relations



IMMCA [Peters et al., 2012] = Iterative Multi-Labels Multi-Relational Classification

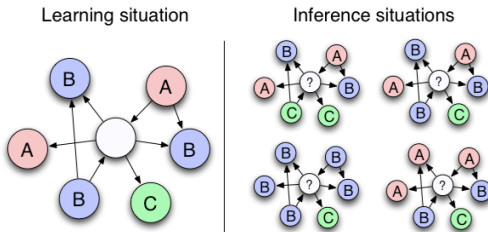
Algorithm 1 IMMCA Inference algorithm

```
1: procedure INFERENCE( $\mathcal{G}, \mathcal{L}$ )
2:   for all  $n_i \in \mathcal{N}_u, l \in \mathcal{L}$  do                                 $\triangleright$  Initialization
3:      $y_i^{l,(t)} \leftarrow \text{random}$ 
4:   end for
5:    $t \leftarrow 1$ 
6:   while  $t \leq \text{maxt}$  do                                          $\triangleright$  maxt iterations
7:     Choose  $n_i \in \mathcal{N}_u$  at random
8:     for all  $l \in \mathcal{L}$  do       $\triangleright$  Compute scores for all labels
9:        $y_i^{l,(t)} \leftarrow f_{\theta}(n_i, l, \mathcal{S}^{(t-1)}(n_i))$ 
10:    end for
11:    for all  $n_j \neq n_i, l \in \mathcal{L}$  do
12:       $y_j^{l,(t)} \leftarrow y_j^{l,(t-1)}$ 
13:    end for
14:     $t \leftarrow t + 1$ 
15:  end while
16:  return  $\{y_i^{l,(\text{maxt})}\}_{n_i \in \mathcal{N}}$ 
17: end procedure
```

Avec $\mathcal{S}^{(t)}(n_i)$ l'ensemble des scores pour chaque label dans le voisinage de n_i au temps t .

Inférence Collective: ICA

- Inférence collective par ICA
 - Étiquetage du graphe avec prise en compte locale de dépendances connues
 - Basé sur un modèle de dépendances f entre éléments connectés
- Comment apprendre le modèle de dépendances?
 - Apprentissage a priori
 - Modèle fixe
 - Requiert une large part de labels connus
 - Biais d'apprentissage: pas représentatif des situations d'inférence



- Inférence collective par ICA
 - Étiquetage du graphe avec prise en compte locale de dépendances connues
 - Basé sur un modèle de dépendances f entre éléments connectés
- Comment apprendre le modèle de dépendances?
 - Apprentissage itératif ?
 - Prise en compte des étiquettes inférées
 - ⇒ Propagation des connaissances
 - ⇒ Réduction du biais d'apprentissage: situations en apprentissage et en inférence mieux connectées
 - .. Mais risque de divergence

Simulated ICA Learning algorithm [Maes et al., 2009]

```
Input: A labeled graph  $(G, Y)$   
Input: A learning algorithm  $\mathcal{A}$   
Output: A classifier  $f$   
repeat  
  // Bootstrapping  
  foreach  $x_i \in X$  do  
    compute  $\tilde{a}_i$  using current assignments to  $\mathcal{N}_i$   
     $y_i \leftarrow f(\tilde{a}_i)$   
  end  
  // Iterative Classification  
  repeat  
    Generate ordering  $\mathcal{O}$  over nodes of  $G$ .  
    foreach  $i \in \mathcal{O}$  do  
      compute  $\tilde{a}_i$  using current assignments to  $\mathcal{N}_i$   
       $y_i \leftarrow f(\tilde{a}_i)$   
      submit training example  $(\tilde{a}_i, y_i)$  to  $\mathcal{A}$   
    end  
  until iterative classification terminates  
until learning has converged or a threshold number of iterations have elapsed  
return the classifier trained by  $\mathcal{A}$ 
```

Inférence Collective: Gibbs Sampling

- Gibbs Sampling

- Processus d'échantillonnage itératif permettant de tirer aléatoirement un élément de Ω selon une loi jointe multivariée π
- Couramment utilisé lorsqu'un tirage direct est trop complexe
- Idée : Il est plus simple d'échantillonner à partir d'une distribution conditionnelle que considérer l'intégration de la loi jointe

$$p(x_j | x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_n) = \frac{p(x_1, \dots, x_n)}{p(x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_n)} \propto p(x_1, \dots, x_n)$$

⇒ A partir d'un état courant, \vec{x} :

- 1 On choisit une variable x_j aléatoirement
- 2 On tire une nouvelle valeur v pour x_j proportionnellement à la probabilité conditionnelle : $p(x_j = v | x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_n)$

Application à un réseau de dépendances locales

- Estimation de probabilité postérieure par Gibbs Sampling
- Deux phases
 - Période de *burn-in* durant laquelle on tire itérativement de nouvelles valeurs pour les variables de \vec{x}
 - Objectif: Atteindre un échantillon probable selon la loi jointe
 - Période de comptage durant laquelle on compte les affectations successives des valeurs des variables de \vec{x}
 - Les compteurs d'affectation permettent finalement d'estimer les probabilité d'observation des différentes valeurs possibles pour chaque variable de \vec{x}
- Inférence collective par Gibbs Sampling
 - On cherche à estimer les probabilités des différents labels pour les noeuds du graphe
 - Sampling des valeurs par échantillonnages successifs des labels conditionnellement à tous les voisins dans le graphe

Inférence Collective: Gibbs Sampling

Algorithm 1: Collective Gibbs Sampling Algorithm

```
1 // Bootstrap
2 for each node  $x_i \in \mathcal{X}$  do
3    $y_i \leftarrow$  Sample uniformly  $l$  from  $\mathcal{L}$ 
4 end
5 // Burn-in
6 for  $it = 1$  to  $nbItBurnIn$  do
7   for each node  $x_i \in \mathcal{X}$  do
8      $y_i \leftarrow$  Sample  $l$  from  $\mathcal{L}$  w.r.t.  $P_\theta(y_i = l \mid x_i, Y \setminus y_i)$ 
9   end
10 end
11 // Init counts
12 for each node  $x_i \in \mathcal{X}$  do
13   for each label  $l \in \mathcal{L}$  do
14      $c[i, l] \leftarrow 0$ 
15   end
16 end
17 // Collect Samples
18 for  $it = 1$  to  $nbIt$  do
19   for each node  $x_i \in \mathcal{X}$  do
20      $y_i \leftarrow$  Sample  $l$  from  $\mathcal{L}$  w.r.t.  $P_\theta(y_i = l \mid x_i, Y \setminus y_i)$ 
21      $c[i, y_i] \leftarrow c[i, y_i] + 1$ 
22   end
23 end
24 // Compute Posteriors
25 for each node  $x_i \in \mathcal{X}$  do
26   for each label  $l \in \mathcal{L}$  do
27      $c[i, y_i] \leftarrow c[i, y_i] / nbIt$ 
28   end
29 end
```

- Algorithme très proche de ICA mais :
 - Non déterministe (sampling)
 - Prise en compte de la vraisemblance des voisins:

$$P_{\theta}(y_i = l \mid x_i, Y \setminus y_i) \propto \frac{P_{\theta}(y_i = l, Y \setminus y_i \mid X)}{\sum_{l' \in \mathcal{L}} P_{\theta}(y_i = l', Y \setminus y_i \mid X)}$$
$$\propto \frac{f_{\theta}(l, x_i, Y(\mathcal{N}_i)) \prod_{j \in \mathcal{N}_i} f_{\theta}(y_j, x_j, Y(\mathcal{N}_j \setminus i) \cup \{y_i = l\})}{\sum_{l' \in \mathcal{L}} f_{\theta}(l', x_i, Y(\mathcal{N}_i)) \prod_{j \in \mathcal{N}_i} f_{\theta}(y_j, x_j, Y(\mathcal{N}_j \setminus i) \cup \{y_i = l'\})}$$

Avec par exemple $f_{\theta}(l, x_i, Y) = \frac{1}{1 + \exp^{-\langle \theta, \tilde{a}_i \rangle}}$ où \tilde{a}_i est une représentation vectorielle du noeud i et de l'étiquetage de son voisinage (comme dans ICA).

Inférence Collective: Gibbs Sampling

Algorithm 2: Collective Gibbs Sampling with iterative learning

```
1 // Bootstrap
2 for each node  $x_i \in \mathcal{X}$  do
3    $y_i \leftarrow$  Sample uniformly  $l$  from  $\mathcal{L}$ 
4 end
5 // Burn-in
6 for  $it = 1$  to  $nbItBurnIn$  do
7   for each node  $x_i \in \mathcal{X}$  do
8      $y_i \leftarrow$  Sample  $l$  from  $\mathcal{L}$  w.r.t.  $P_\theta(y_i = l \mid x_i, Y \setminus y_i)$ 
9   end
10  // Likelihood maximization
11   $\theta = \arg\max_{\theta'} P_{\theta'}(Y \mid X)$ 
12 end
13 // Init counts
14 for each node  $x_i \in \mathcal{X}$  do
15   for each label  $l \in \mathcal{L}$  do
16      $c[i, l] \leftarrow 0$ 
17   end
18 end
19 // Collect Samples
20 for  $it = 1$  to  $nbIt$  do
21   for each node  $x_i \in \mathcal{X}$  do
22      $y_i \leftarrow$  Sample  $l$  from  $\mathcal{L}$  w.r.t.  $P_\theta(y_i = l \mid x_i, Y \setminus y_i)$ 
23      $c[i, y_i] \leftarrow c[i, y_i] + 1$ 
24   end
25 end
26 // Compute Posteriors
27 for each node  $x_i \in \mathcal{X}$  do
28   for each label  $l \in \mathcal{L}$  do
29      $c[i, y_i] \leftarrow c[i, y_i] / nbIt$ 
30   end
31 end
```

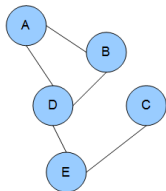
Inférence Collective: Loopy Belief Propagation

Inférence collective: Loopy Belief Propagation

- Inférence dans Markov Random Field

- Réseau de Markov = Graphe de dépendances non dirigées
- Dépendances plus complexes que les modèles précédents

- Découpage en cliques : ABD, DE et EC



- Probabilité d'une instantiation X :

$$P(X = x) = \frac{1}{Z} \prod_k \phi_k(x_k)$$

- x_k est l'instantiation de la k -ième clique du graphe
- ϕ_k est la fonction potentiel définie pour cette clique k (compatibilité des éléments d'une instantiation de cette clique)
- $Z = \sum_{X \in \mathcal{X}} \prod_k \phi_k(x_k)$ est une constante de normalisation calculée sur toutes les configurations possibles X

- Application aux données relationnelles [Taskar et al., 2007]

- On définit un ensemble de templates de cliques \mathcal{C}

Exemple :

```
SELECT d1.label, d2.label
FROM doc d1, doc d2
WHERE link.from = d1
      AND
      link.to = d2;
```

- Pour chaque template de clique $C \in \mathcal{C}$
 - On calcule les cliques $c \in C$
 - On définit le potentiel $\phi_C(x_c)$ qui retourne la compatibilité des valeurs attribuées pour les noeuds d'une clique c
- On pose alors la probabilité d'un étiquetage y par :

$$P(X = x) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \prod_{c \in C} \phi_C(x_c)$$

- $Z = \sum_{X \in \mathcal{X}} \prod_{C \in \mathcal{C}} \prod_{c \in C} \phi_C(x_c)$ est la constante de normalisation calculée sur toutes les configurations possibles X
- $\phi_C(x_c)$ peut être de la forme $\exp(\langle w, g(x_c) \rangle)$, afin de ré-écrire $P(X = x)$ sous forme d'un modèle log-linéaire

- pairwise Markov Random Field

- On connaît des valeurs de potentiel (apprenues a priori) pour:
 - Les priors de étiquettes
 - Les adéquations du contenu avec les labels
 - Des compatibilités label-label pour les noeuds connectés
- On pose $p(y|x)$ la probabilité d'un étiquetage y parmi l'ensemble des étiquetages possibles pour les labels non observés des noeuds de x :

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathcal{Z}(\mathbf{x})} \prod_{Y_i \in \mathcal{Y}} \phi_i(y_i) \prod_{(Y_i, Y_j) \in E} \psi_{ij}(y_i, y_j)$$

Labels Non-Observés

Compatibilité
Label - Label

$$\phi_i(y_i) = \psi_i(y_i) \prod_{(Y_i, X_j) \in E} \psi_{ij}(y_i)$$

Prior

Adéquation
Contenu - Label

- Belief Propagation

- Inventé dans [Pearl, 1982] pour le calcul de probabilités marginales dans les arbres
- Algorithme de passage de messages entre variables suivant une loi jointe
 - Croyance d'une variable de l'état dans lequel doit être une variable liée

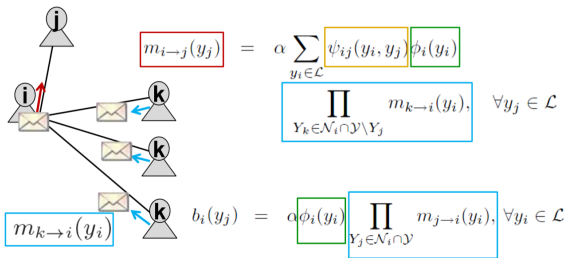
- Loopy Belief Propagation

- Application de Belief Propagation dans les graphes généraux
- Approximation de marginales dans les Bayesian Networks ou Markov Random Fields

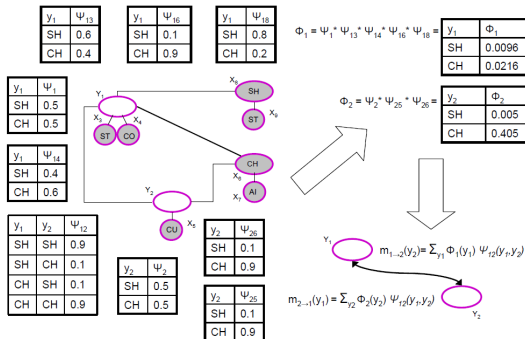
Inférence collective: Loopy Belief Propagation

Application au pairwise Markov Random Field :

- $m_{i \rightarrow j}(y_j)$: message de Y_i à Y_j sur sa croyance de la classe y_j pour le noeud Y_j
- $b_i(y_i)$: "belief" (croyance) de la classe y_i pour le noeud Y_i
= probabilité marginale $p(Y_i = y_i)$



Inférence collective: Loopy Belief Propagation



Algorithm 3 Loopy belief propagation (LBP)

```
for each  $(Y_i, Y_j) \in E(G)$  s.t.  $Y_i, Y_j \in \mathcal{Y}$  do
  for each  $y_j \in \mathcal{L}$  do
     $m_{i \rightarrow j}(y_j) \leftarrow 1$ 
  end for
end for
repeat // perform message passing
  for each  $(Y_i, Y_j) \in E(G)$  s.t.  $Y_i, Y_j \in \mathcal{Y}$  do
    for each  $y_j \in \mathcal{L}$  do
       $m_{i \rightarrow j}(y_j) \leftarrow \alpha \sum_{y_i} \psi_{ij}(y_i, y_j) \phi_i(y_i)$ 
       $\prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k \rightarrow i}(y_i)$ 
    end for
  end for
until all  $m_{i \rightarrow j}(y_j)$  stop showing any change
for each  $Y_i \in \mathcal{Y}$  do // compute beliefs
  for each  $y_i \in \mathcal{L}$  do
     $b_i(y_i) \leftarrow \alpha \phi_i(y_i) \prod_{Y_j \in \mathcal{N}_i \cap \mathcal{Y}} m_{j \rightarrow i}(y_i)$ 
  end for
end for
```

Relationnal Markov Random Field

- Relationnal Markov Random Field : Apprentissage

- $\phi_C(x_c)$ peut être de la forme $\exp(\langle w, g(x_c) \rangle)$:

$$\log P(X = x) = \sum_{C \in \mathcal{C}} \sum_{c \in C} \langle w, g(x_c) \rangle - \log Z$$

- Avec un prior sur les paramètres

$$p(w_i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp -w_i^2/2\sigma^2, \text{ le MAP est donné par :}$$

$$L = \sum_{C \in \mathcal{C}} \sum_{c \in C} \langle w, g(x_c) \rangle - \log Z - \frac{\|w\|_2^2}{2\sigma^2} + C$$

- Modèle log-linéaire dont le gradient est donné par:

$$\nabla_w L = \sum_{C \in \mathcal{C}} \sum_{c \in C} g(x_c) - E_{P_w}[g(X)] - \frac{w}{\sigma^2}$$

Avec $E_{P_w}[g(X)] = \sum_{x \in X} P_w(x) \sum_{C \in \mathcal{C}} \sum_{c \in C} g(x_c)$ l'espérance de potentiel pour les étiquetages possibles du graphe.

- Calcul de $E_{P_w}[g(X)]$ impossible (trop complexe)

⇒ Approximation (e.g. par Gibbs Sampling)

Inférence Collective: Modèle régularisé

Apprentissage régularisé selon graphe

- Modèle régularisé selon un graphe
- Problème d'apprentissage transductif :
 - Ensemble de données $X = (x_1, \dots, x_n)$ avec $X = X_l \cup X_u$.
 - Arcs $E : X^2 \rightarrow \mathbb{R}^+$ tel que: $E(i, j)$ est le poids de l'arc (i, j)
 - Ensemble d'étiquettes $Y = (y_1, \dots, y_l)$ connues pour les données de X_l
 - On cherche les étiquettes restantes (y_{l+1}, \dots, y_n)
- ⇒ La structure du graphe correspond à une régularité sur la fonction de prédiction $f_\theta : X \rightarrow Y$ à définir.
- Fonction f_θ :
 - Comportements similaires pour données au contenu proche
 - Comportements similaires pour données fortement connectées
- ⇒ Propagation de labels

- Risque régularisé classique:

$$L = \sum_{x_i \in X_I} \Delta(f_\theta(x_i), y_i) + \lambda \|\theta\|^2$$

- Modèle transductif pour les graphes [Belkin et al., 2004]:

$$\operatorname{argmin}_f \sum_{x_i \in X_I} (y_i - f_i)^2 + \beta \sum_{i,j \in E} E(i,j) (f_i - f_j)^2$$

- Modèle résultant [Denoyer & Gallinari, 2010]:

$$\operatorname{argmin}_\theta \sum_{x_i \in X_I} \Delta(f_\theta(x_i), y_i) + \beta \sum_{i,j \in E} E(i,j) (f_\theta(x_i) - f_\theta(x_j))^2 + \lambda \|\theta\|^2$$

- Risque régularisé classique:

$$L = \sum_{x_i \in X_I} \Delta(f_\theta(x_i), y_i) + \lambda \|\theta\|^2$$

- Modèle transductif pour les graphes [Belkin et al., 2004]:

$$\operatorname{argmin}_f \sum_{x_i \in X_I} (y_i - f_i)^2 + \beta \sum_{i,j \in E} E(i,j) (f_i - f_j)^2$$

- Modèle résultant [Denoyer & Gallinari, 2010]:

$$\operatorname{argmin}_\theta \sum_{x_i \in X_I} \Delta(f_\theta(x_i), y_i) + \beta \sum_{i,j \in E} E(i,j) (f_\theta(x_i) - f_\theta(x_j))^2 + \lambda \|\theta\|^2$$

⇒ Mais comment déterminer les poids $E(i,j)$?

- Et si l'hypothèse de smoothness (sur le graphe, où sur la similarité utilisée) est mauvaise ?
 - \Rightarrow Alors le modèle marchera mal.....
- Et si la smoothness n'est pas garantie sur tout le graphe ?
 - \Rightarrow Alors le modèle marchera mal.....
- Et si je ne connais pas les pondérations à appliquer pour la smoothness ?
 - \Rightarrow Alors le modèle marchera mal.....

La smoothness peut être apprise directement à partir des données étiquetés.

...et donc déduire où/pourquoi les données respectent cette hypothèse.

- On définit un sous ensemble E_l de E tel que:

$$E_l = \{(i, j) \in E / x_i \in X_l \text{ et } x_j \in X_l\}$$

- On définit un ensemble d'étiquettes sur les liens

$$\forall (i, j) \in E_l, y_{(i, j)} = \begin{cases} 1 & \text{si } y_i = y_j \\ 0 & \text{sinon} \end{cases}$$

On est alors capable d'apprendre un classifieur sur les liens
 $g_w : (i, j) \in E \rightarrow [0; 1]$

On définit une nouvelle fonction objective comme:

$$\begin{aligned} L = & \sum_{x_i \in X_I} \Delta(f_\theta(x_i), y_i) \\ & + \alpha \sum_{(i,j) \in E_I} \Delta'(g_\omega(i,j), y_{(i,j)}) \\ & + \beta \sum_{(i,j) \in E} g_\omega(i,j) (f(x_i) - f(x_j))^2 \\ & + \lambda \|\theta\|^2 \\ & + \lambda' \|\omega\|^2 \end{aligned}$$

Cette fonction se minimise en fonction de θ et ω par une descente de gradient classique.

Fonction g_ω :

$$g_\omega(i,j) = g'(< \omega; \Phi(i,j) >)$$

où $\Phi(i,j)$ est une représentation vectorielle.

$\Phi(i,j)$ peut prendre plusieurs formes.

- Cas mono-relationnel:

- $\Phi(i,j) \in \mathbb{R}^1$

$$\Phi(i,j) = \begin{cases} 1 & \text{si } (i,j) \in E \\ 0 & \text{sinon} \end{cases}$$

Fonction g_ω :

$$g_\omega(i,j) = g'(< \omega; \Phi(i,j) >)$$

où $\Phi(i,j)$ est une représentation vectorielle.

$\Phi(i,j)$ peut prendre plusieurs formes.

- Cas mono-relationnel:

- $\Phi(i,j) \in \mathbb{R}^1$

$$\Phi(i,j) = \begin{cases} 1 & \text{si } (i,j) \in E \\ 0 & \text{sinon} \end{cases}$$

- ⇒ Le modèle est capable d'apprendre l'importance des étiquettes voisines à inclure dans la fonction à optimiser
- ⇒ Pondération uniforme permettant d'ajuster l'importance de la smoothness sur l'ensemble du graphe

Fonction g_{ω} :

$$g_{\omega}(i,j) = g'(<\omega; \Phi(i,j) >)$$

où $\Phi(i,j)$ est une représentation vectorielle.

$\Phi(i,j)$ peut prendre plusieurs formes.

- Cas multi-relationnel:

- $\Phi(i,j) \in \mathbb{R}^r$ où r est le nombre de relations entre les données
- $E^j, j \in [1; r]$ est l'ensemble des arcs pour la relation j

$$\Phi^k(i,j) = \begin{cases} 1 & \text{si } (i,j) \in E^k \\ 0 & \text{sinon} \end{cases}$$

Fonction g_ω :

$$g_\omega(i,j) = g'(<\omega; \Phi(i,j) >)$$

où $\Phi(i,j)$ est une représentation vectorielle.

$\Phi(i,j)$ peut prendre plusieurs formes.

- Cas multi-relationnel:

- $\Phi(i,j) \in \mathbb{R}^r$ où r est le nombre de relations entre les données
- $E^j, j \in [1; r]$ est l'ensemble des arcs pour la relation j

$$\Phi^k(i,j) = \begin{cases} 1 & \text{si } (i,j) \in E^k \\ 0 & \text{sinon} \end{cases}$$

- ⇒ Le modèle obtenu est un modèle capable d'apprendre par relation, à quel point un arc entraine de la smoothness. Le modèle découvre donc les relations pertinentes (au sens de la smoothness).
- ⇒ Pondération uniforme sur les différents arcs de même type (mais un poids différent par type de relation)

Fonction g_ω :

$$g_\omega(i, j) = g'(\langle \omega; \Phi(i, j) \rangle)$$

où $\Phi(i, j)$ est une représentation vectorielle.

$\Phi(i, j)$ peut prendre plusieurs formes.

- Fonction du contenu
- On peut imaginer plusieurs représentations:
 - $\Phi(i, j) = \frac{x_i + x_j}{2}$
 - $\Phi(i, j) = x_i - x_j$
 - $\Phi(i, j) = \begin{pmatrix} x_i \\ x_j \end{pmatrix}$

Inférence Collective: Modèle régularisé

Fonction g_{ω} :

$$g_{\omega}(i,j) = g'(<\omega; \Phi(i,j) >)$$

où $\Phi(i,j)$ est une représentation vectorielle.

$\Phi(i,j)$ peut prendre plusieurs formes.

- Fonction du contenu
- On peut imaginer plusieurs représentations:

- $\Phi(i,j) = \frac{x_i + x_j}{2}$
- $\Phi(i,j) = x_i - x_j$
- $\Phi(i,j) = \begin{pmatrix} x_i \\ x_j \end{pmatrix}$

- ⇒ Le modèle obtenu est un modèle capable d'apprendre l'importance des étiquettes voisines en fonction de la ressemblance de leur contenu
- ⇒ Pondération différente par relation en fonction du contenu des noeuds connectés

Inférence Collective: Modèle régularisé

Fonction g_ω :

$$g_\omega(i, j) = g'(< \omega; \Phi(i, j) >)$$

où $\Phi(i, j)$ est une représentation vectorielle.

$\Phi(i, j)$ peut prendre plusieurs formes.

- Fonction du contenu
- On peut imaginer plusieurs représentations:

- $\Phi(i, j) = \frac{x_i + x_j}{2}$
- $\Phi(i, j) = x_i - x_j$
- $\Phi(i, j) = \begin{pmatrix} x_i \\ x_j \end{pmatrix}$

- ⇒ Le modèle obtenu est un modèle capable d'apprendre l'importance des étiquettes voisines en fonction de la ressemblance de leur contenu
- ⇒ Pondération différente par relation en fonction du contenu des noeuds connectés
- ⇒ Si pas de graphe: Modification de la frontière f_θ pour que celle-ci passe par des regions peu "denses" de l'espace;

Fonction g_ω :

$$g_\omega(i, j) = g'(< \omega; \Phi(i, j) >)$$

où $\Phi(i, j)$ est une représentation vectorielle.

$\Phi(i, j)$ peut prendre plusieurs formes.

- Fonction noyau $K(x_i, x_j)$
 - $\Phi(i, j) = \left(\phi^k_{(i, j)} \right)$ où $\phi^k(i, j) = K^k(x_i, x_j)$
 - $K^k(x_i, x_j)$ peut prendre diverses formes (noyau Gaussien, polynomial, etc...)
 - ... et intégrer diverses informations (contenu, profil, nombre de voisins communs, existence d'un arc pendant une période donnée, etc...)

Fonction g_ω :

$$g_\omega(i, j) = g'(< \omega; \Phi(i, j) >)$$

où $\Phi(i, j)$ est une représentation vectorielle.

$\Phi(i, j)$ peut prendre plusieurs formes.

- Fonction noyau $K(x_i, x_j)$

- $\Phi(i, j) = \left(\phi_{(i,j)}^k \right)$ où $\phi^k(i, j) = K^k(x_i, x_j)$
- $K^k(x_i, x_j)$ peut prendre diverses formes (noyau Gaussien, polynomial, etc...)
- ... et intégrer diverses informations (contenu, profil, nombre de voisins communs, existence d'un arc pendant une période donnée, etc...)

- ⇒ Pondération différente par relation en fonction de la valeur des fonction noyaux des noeuds connectés
- ⇒ Smoothness dépendente des noyaux définis sur les paires de noeuds

- [Belkin et al., 2004] Mikhail Belkin, Irina Matveeva, Partha Niyogi: Regularization and Semi-supervised Learning on Large Graphs. COLT 2004: 624–638
- [Friedman et al., 1999] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In Thomas Dean, editor, IJCAI, pages 1300–1309. Morgan Kaufmann, 1999.
- [Ludovic Denoyer & Patrick Gallinari, 2010] Ludovic Denoyer and Patrick Gallinari. A Ranking Based Model for Automatic Image Annotation in a Social Network. ICWSM 2010.
- [McDowell & Aha, 2013] Luke K. McDowell and David W. Aha. 2013. Labels or attributes?: rethinking the neighbors for collective classification in sparsely-labeled networks. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM '13). ACM, New York, NY, USA, 847–852
- [Maes et al., 2009] Francis Maes, Stéphane Peters, Ludovic Denoyer, Patrick Gallinari: Simulated Iterative Classification A New Learning Procedure for Graph Labeling. ECML/PKDD (2) 2009: 47-62
- [Neville & Jensen, 2002] Jennifer Neville and David Jensen. Relational dependency networks. Journal of Machine Learning Research, 8 :653–692, 2007.
- [Pearl, 1982] Pearl, Judea (1982). "Proceedings of the Second National Conference on Artificial Intelligence". AAAI-82: Pittsburgh, PA. Menlo Park, California: AAAI Press. pp. 133–136. Retrieved 2009-03-28.
- [Peters et al., 2012] Stéphane Peters, Yann Jacob, Ludovic Denoyer, Patrick Gallinari: Iterative Multi-label Multi-relational Classification Algorithm for complex social networks. Social Netw. Analys. Mining 2(1): 17-29 (2012)
- [Sen et al., 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, Tina Eliassi-Rad: Collective Classification in Network Data. AI Magazine 29(3): 93–106 (2008)
- [Taskar et al., 2002] Benjamin Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In Adnan Darwiche and Nir Friedman, pages 485–492. Morgan Kaufmann, 2002.
- [Taskar et al., 2007] Taskar, B., Abbeel, P., Wong, M.-F., and Koller, D. Relational markov networks. In Getoor, L. and Taskar, B. (eds.), Introduction to Statistical Relational Learning. MIT Press, 2007.