HAL
open science

Trois approches pour classifier les données du web des données

Justine Reynaud, Yannick Toussaint, Amedeo Napoli

To cite this version:

Justine Reynaud, Yannick Toussaint, Amedeo Napoli. Trois approches pour classifier les données du web des données. SFC 2018 - XXVèmes Rencontres de la Société Francophone de Classification, Sep 2018, Paris, France. hal-01887884

HAL Id: hal-01887884

https://inria.hal.science/hal-01887884v1

Submitted on 4 Oct 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Trois approches pour classifier les données du web des données

Justine Reynaud *, Yannick Toussaint *, Amedeo Napoli *

*Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
prénom.nom@loria.fr

Résumé. Nous disposons d'un ensemble d'objets, appartenant à une ou plusieurs classes, entre lesquels il existe des relations. Les classes sont définies en *extension*, et nous cherchons à construire une description en *intension* en s'appuyant sur les relations des objets qui les composent. Pour cela, nous employons trois approches : les règles d'association, les redescriptions et les règles de traduction. À partir d'expérimentations, nous nous intéressons aux spécifictés et à la complémentarité de ces approches.

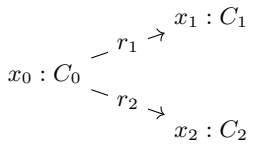
1 Introduction

Nous cherchons à découvrir des définitions dans les données RDF issues du web des données. Nous disposons d'un ensemble d'objets connectés par des relations. L'objectif est de classifier ces objets en fonction des relations dans lesquelles ils sont impliqués. Les objets qui partagent des éléments communs appartiennent à une même classe. Ce partage peut être exact — les éléments sont identiques — ou approximatif — les éléments sont similaires. Nous obtenons un ensemble de classes organisées selon un ordre partiel, et les descriptions associées à ces classes. Ces descriptions sont nécessaires afin de construire les définitions des différentes classes.

L'idée est de construire et d'appliquer des règles d'induction de la forme « $\mathtt{si\ r(x,y)\ et\ y{:}C\ alors\ x{:}\exists r.C}$ ». Cela signifie que, étant données $\mathtt{y}$ une instance de la classe $\mathtt{C}$ et $\mathtt{r}$ une relation telle que $\mathtt{r(x,y)}$, alors $\mathtt{x}$ appartient à une classe, disons $\mathtt{D}$, dont la description comprend $\exists\mathtt{r.C}$; donc les instances de $\mathtt{D}$ sont reliées à au moins une instance de $\mathtt{C}$ par la relation $\mathtt{r}$. Nous utilisons ce type de règles pour construire les définitions des classes. Ces définitions sont de la forme $\mathtt{C_i} \equiv \mathtt{e_1} \sqcap \mathtt{e_2} \sqcap \ldots \sqcap \mathtt{e_n}$ où $\mathtt{e_j}$ est une expression de la forme $\exists\mathtt{r.C_j}$. Nous nous appuyons ici sur trois approches, les règles d'association, la fouille de redescriptions et la découverte de règles de traduction, que nous souhaitons comparer.

2 Représentation des données

Le Web des Données (LOD) peut être vu comme un ensemble de KBs interconnectées. L'unité de base d'une KB est le triplet RDF, noté $\langle s, p, o\rangle$, qui encode une assertion sous la forme sujet–prédicat–objet. Ici, nous nous restreignons aux triplets



(a) Données relationnelles.

```
x0 r1 x1 .
x0 r2 x2 .
x0 a C0 .
x1 a C1 .
x2 a C2 .
```

(b) Données RDF.

	M_C			M_D	
	C_0	C_1	C_2	$\exists r1{:}x_1$	$\exists r2{:}x_2$
x_0	×			×	×
x_1		×			
x_2			×		

(c) Contexte formel.

FIG. 1: Données relationnelles, leur formalisation en RDF et le contexte formel associé.

tels que $\langle s, p, o\rangle \in U \times U \times U$ où U est un ensemble de ressources. Une ressource peut faire référence à n'importe quel objet ou abstraction. La figure 1, présente un exemple de données relationnelles (1a) et l'ensemble de triplets correspondant (1b). Le prédicat `a` correspond à la relation d'instanciation.

À partir des données RDF, nous construisons un contexte formel (G, M, I) où G est l'ensemble des sujets des triplets et M est l'ensemble des des paires (prédicat,objet) issues des triplets. L'ensemble des attributs M est une partition de deux ensembles : $M = M_C \cup M_D$. L'ensemble M_C est composé des paires de la forme $(\mathtt{a}, \mathtt{C})$, qui font d'un sujet une instance de la classe `C`. Ce sont ces classes que nous cherchons à définir. L'ensemble M_D est composé de paires $(\mathtt{p}, \mathtt{o})$ telles que $\mathtt{p} \neq \mathtt{a}$. La figure 1c présente le contexte associé aux triplets de la figure 1b. À partir du contexte construit nous cherchons un ensemble de catégories et un ensemble de descriptions de manière à ce que leurs dérivations soient les mêmes.

3 Algorithmes de fouille de règles

3.1 Règles d'association

Une règle d'association (Han et al., 2011) entre deux ensembles d'attributs X et Y, notée $X \to Y$, signifie « si un objet a tous les attributs de X, alors il a tous les attributs de Y ». La qualité d'une règle d'association est mesurée avec la confiance : $\mathrm{conf}(X \to Y) = \frac{|X' \cap Y'|}{|X'|}$. Pour un seuil θ défini par l'utilisateur, une règle d'association est valide si $\mathrm{conf}(X \to Y) \geqslant \theta$. Si $\theta = 1$, il s'agit d'une implication et l'on note $X \Rightarrow Y$. Si on a également $Y \Rightarrow X$, alors X et Y forment une définition, notée $X \equiv Y$. Ici, nous considérons conjointement $X \to Y$ et $Y \to X$ et cherchons à estimer à quel point chacune de ces règles s'approche d'une implication. Pour cela, nous introduisons la notion de quasi-définition qui est à la définition ce que la règle d'association est à l'implication. Étant donné deux ensembles d'attributs X et Y, une *quasi-définition* $X \leftrightarrow Y$ est valide ssi $\min(\mathrm{conf}(X \to Y), \mathrm{conf}(Y \to X)) \geqslant \theta$ pour un seuil θ donné. Nous utilisons l'algorithme `Eclat` (Zaki, 2000). Un post-traitement nous permet d'extraire les quasi-définitions à partir des règles d'association obtenues

3.2 Redescriptions

La fouille de redescriptions (Galbrun et Miettinen, 2018) cherche des caractérisations multiples d'un même ensemble d'entités. Les redescriptions s'appuient sur une

séparation de l'ensemble des attributs en *vues*, l'ensemble des vues formant une partition des attributs. Nous utilisons ici deux vues, qui correspondent aux ensembles d'attributs M_C et M_D. La similarité entre deux ensembles d'attributs provenant de deux vues différentes, est mesurée avec le coefficient de Jaccard : $\text{jacc}(A, B) = \frac{|A' \cap B'|}{|A' \cup B'|}$. Si, pour un seuil θ défini par l'utilisateur, $\text{jacc}(A, B) \geqslant \theta$ alors (A, B) est une redescription. Une redescription dont le coefficient de Jaccard vaut 1 est une définition. Toutes les redescriptions sont nécessairement des quasi-définitions : $\min(\text{conf}(A \rightarrow B), \text{conf}(B \rightarrow A)) \geqslant \text{jacc}(A, B)$. L'algorithme `ReReMi` (Galbrun et Miettinen, 2012) est utilisé ici pour extraire les redescriptions.

3.3 Règles de traduction

L'algorithme `Translator` considère des données divisées en deux vues, et recherche un ensemble de règles qui permette de reconstruire une vue à partir de l'autre. Cet ensemble doit à la fois couvrir la majorité des données et avoir les plus petites règles possible en terme d'attributs. Afin de trouver un équilibre entre ces deux contraintes, `Translator` se base sur le concept de Longueur de Description Minimum (MDL).

Pour construire l'ensemble des règles, les auteurs font l'analogie avec la notion de traduction. Une règle est une traduction d'une vue vers une autre. L'idée sous-jacente est la suivante : on souhaite construire un ensemble de règles qui permettent, connaissant une des deux vues, de reconstruire la seconde et *vice versa*. Les erreurs entre la vue cible et la vue reconstruite sont corrigées à l'aide d'un masque. La taille de ce masque indique donc le nombre d'erreurs introduites. À chaque ensemble d'attribut X est associé une taille $L(X)$ qui dépend de la fréquence de X dans les données. L'ensemble de règles est construit de manière itérative, en prenant, à chaque étape, la règle $X \rightarrow Y$ qui maximise $\Delta(X \rightarrow Y) = L(Mask^-) - L(Mask^+) - L(X \cup Y)$ où $Mask^+$ correspond aux éléments ajoutés au masque (erreurs introduites par la règle), $Mask^-$ correspond aux éléments retirés du masque (erreurs corrigées par la règle) et $L(X \cup Y)$ correspond à la longueur de la règle.

4 Expérimentations

Nous avons réalisé nos expérimentations sur des données issues de *DBpedia*, qui est une KB construite à partir de *Wikipédia*. Nous avons extrait quatre sous-ensembles de triplets de *DBpedia*, de différents domaines. Chaque algorithme X retourne un ensemble ordonné de quasi-définitions $\mathcal{B}^X_{cand}$ de la forme $C_0, \ldots, C_n \leftrightarrow D_0, \ldots, D_m$. Chacune des quasi-définitions est évaluée manuellement par des experts. Si la règle est acceptée, elle rejoint l'ensemble des définitions obtenues $\mathcal{B}^X_{def}$ (cf figure 2).

Les algorithmes sont comparés sur la base des définitions extraites et des catégories définies. Ils se distinguent sur la quantité et la forme des règles retournées, mais aussi sur leur précision. Le nombre de règles retournées varie entre `Eclat` et les autres algorithmes. `Eclat` retourne entre 4 et 20 fois plus de règles que `ReReMi` ou `Translator`. En moyenne, `Translator` retourne autant de règles que `ReReMi`. `Eclat` et `Translator` retournent tous deux des règles assez longues : entre 6 et 8 attributs par règle, contre 2 à 3 attributs pour `ReReMi`. Cette différence s'explique par l'heuristique employée par

`ReReMi`. Cela a également pour conséquence de rendre les règles plus faciles à interpréter par l'utilisateur. En revanche, la précision de `ReReMi` a une très forte variabilité (entre 33 et 75%) et est globalement la plus faible. La précision d'`Eclat` est stable (entre 64 et 72%). `Translator` a la meilleure précision, qui est toujours supérieure à 74% quel que soit le jeu de données.

`Sports_cars`

R **McLaren_vehicles** ↔ (manufacturer McLaren_Automotive)

R **McLaren_vehicles** ↔ (assembly Surrey)

ET **McLaren_vehicles, Sports_cars** ↔ (a Automobile), (a MeanOfTransportation), (assembly Woking), (assembly Surrey), (assembly England), (bodyStyle Coupé), (manufacturer McLaren_Automotive)

E **McLaren_vehicles, Sports_cars** ↔ (a Automobile), (a MeanOfTransportation), (assembly England), (assembly Surrey), (bodyStyle Coupé)

E **McLaren_vehicles, Sports_cars** ↔ (a Automobile), (a MeanOfTransportation), (assembly Surrey), (bodyStyle Coupé)

`Turing_Award_laureates`

R **Harvard_University_alumni** ↔ (almaMater Harvard_University)

ET **Harvard_University_alumni, Turing_Award_laureates** ↔ (a Agent), (a Person), (a Scientist), (almaMater Harvard_University)

E **Turing_Award_laureates** ↔ (a Agent), (a Person), (award Turing_Award)

ET **Turing_Award_laureates** ↔ (a Agent), (a Person), (a Scientist), (award Turing_Award)

E **Modern_cryptographers** ↔ (field Cryptography)

`Smartphones`

ET **Firefox_OS_devices, Open-source_mobile_phones, Smartphones, Touchscreen_mobile_phones** ↔ (a Device), (operatingSystem Firefox_OS)

R **Nokia_mobile_phones** ↔ (manufacturer Nokia)

ET **Nokia_mobile_phones, Smartphones** ↔ (a Device), (manufacturer Nokia)

R **Samsung_Galaxy** ↔ (manufacturer Samsung_Electronics), (operatingSystem Android_(operating_system))

ET **Samsung_Galaxy, Samsung_mobile_phones, Smartphones** ↔ (a Device), (manufacturer Samsung_Electronics), (operatingSystem Android_(operating_system))

`French_films`

R **Pathé_films** ↔ (distributor Pathé)

R **Films_directed_by_Georges_Méliès** ↔ (director Georges_Méliès)

ET **Films_directed_by_Georges_Méliès, French_films, French_silent_short_films** ↔ (a Film), (a Work), (director Georges_Méliès)

ET **Films_directed_by_Jean_Rollin, French_films** ↔ (a Film), (a Work), (director Jean_Rollin)

ET **Film_scores_by_Gabriel_Yared, French_films** ↔ (a Film), (a Work), (musicComposer Gabriel_Yared)

FIG. 2: Exemples de définitions extraites pour chaque jeu de données.

Références

Galbrun, E. et P. Miettinen (2012). From Black and White to Full Color : Extending Redescription Mining Outside the Boolean World. *SADM 5*(4), 284–303.

Galbrun, E. et P. Miettinen (2018). *Redescription mining.* Springer.

Han, J., J. Pei, et M. Kamber (2011). *Data mining : concepts and techniques.* Elsevier.

Zaki, M. J. (2000). Scalable algorithms for association mining. *TKDE 12*(3), 372–390.