UNIVERSITE SAIDA - Dr. MOULAY Tahar

FACULTE : TECHNOLOGIE
DEPARTEMENT : INFORMATIQUE

MEMOIRE DE MASTER

OPTION : Réseaux Informatiques et Systèmes Répartis

Thème :

La qualité des données :

Résolution des entités

Présenté par :

BRAHMI Sara

BELLOUT Lina Manel

Encadré par :

BENYAHIA Kadda

BENYAHIA Miloud

Promotion :2019/2020

REMERCIEMENTS

Au terme de ce travail , je tiens à remercier toutes les personnes ayant contribué à la réalisation de ce mémoire.

Un remerciement particulier à est adressé à Monsieur **K.Benyahia** et **M.Benyahia** pour avoir proposé, suivi et dirigé le présent travail.je tiens à lui exprimer ma sincère gratitude pour ses conseils pertinents, pour ses orientations constructives et pour le temps si précieux qu'il m'a consacré.

Mes vifs remerciements s'adressent aussi aux membres du jury Monsieur **A.Zahaf** et monsieur **H.A Bouarara**

pour leur disponibilité et acceptation d’examiner et de rapporter mon travail.

Dédicaces

Je dédie ce modeste travail à mon père qui ma tellement soutenue durant ma vie et ses sacrifices il est l'espoir de mon existence pour réussir dans mes études.

A ma très chère mère pour le soutien moral et affectif

A

Mes frères omar ,Ali, et Sedik

Ma sœur Fatima.

A

Toute ma famille(Brahmi,Bouchikhi et Allali).

A

Tous mes amis.

A tous mes enseignants qui ont fait leurs possibles pour me donner le maximum d'information concernant mes études.

A tous mes collègues de l'université surtout la promotion de l'informatique.

A toutes les personnes qui nous ont aidés de prés ou de loin pour la réalisation de ce mémoire.

Brahmi sara

Dédicaces

Je dédie ce modeste travail à mon père qui ma tellement
soutenue durant ma vie et ses sacrifices il est l'espoir de mon
existence pour réussir dans mes études.
A ma très chère mère pour le soutien moral et affectif

A

Mes frères Mourad , et Hichem
Ma sœur Lamia.

A

L'âme du défunt regretté et cher frère ARABI Abdelatif et que dieu le bénisse

A

*Toute ma famille(**Bellout**).*

A

Tous mes amis.

A tous mes enseignants qui ont fait leurs possibles pour me donner le maximum d'information concernant mes études.

A tous mes collègues de l'université surtout la promotion de l'informatique.

A toutes les personnes qui nous ont aidés de prés ou de loin pour la réalisation de ce mémoire.

Bellout lina manel

Résumé

la résolution d'entité est le processus de liaison d'au moins deux enregistrements d'une base de données à la même entité réelle. «Ces enregistrements ne partagent pas un identifiant commun. «Cela fait de leur connexion entre elles une tâche difficile car elles ne peuvent être liées que sur la base de similitudes dans leurs données. «Ces données peuvent également contenir des erreurs dues à des fautes d'orthographe , ce qui augmente encore la difficulté de la tâche. il y'a, des méthodes communes pour comparer des enregistrements et trouver des doublons sont présentées. Sur la base de ces connaissances, plusieurs expériences comparant ces méthodes ont été menées, en utilisant les données de ensemble de données de référence, y compris Freely Extensible Biomedical Record Linkage (FEBRL)

Sommaire

Liste des Figures

Liste des tableaux

chapitre1

Introduction générale

1.1 Introduction

Avec la croissance de l'industrie informatique, les bases de données sont devenues la solution standard pour les données persistantes. Les systèmes informatiques utilisent ces données pour effectuer des opérations sur la base de l'exactitude des données.

Cependant, ce n'est pas toujours le cas, car les données contiennent souvent des erreurs telles que des fautes d'orthographe ou des champs manquants. Pour aggraver les choses, différentes conventions des mêmes données sont souvent utilisées, ce qui fait que la même entité du monde réel a plus d'une représentation dans la base de données. Lorsque ce problème se produit pour des objets de la vie réelle tels que les humains, les conséquences ont des effets réels, surtout si les données sont utilisées par un système d'agence gouvernementale**.**

1.2 Problématique

Une entité du monde réel peut être une personne, un produit, une entreprise ou tout autre objet. Des représentations multiples de la même entité du monde réel dans plusieurs sources de données sont particulièrement problématiques. Des exemples de représentations multiples (ce que l'on appelle des doublons) comprennent plusieurs représentations du même client dans plusieurs sources de données, différentes représentations du même produit dans un catalogue en ligne, et plusieurs références de la même publication dans différents référentiel du Web. Un défi majeur dans la résolution d'entité est le manque d'identifiants d'entité uniques (clés) à cause de l’autonomie des sources de données à matcher. Par conséquent, la résolution d'entité

doit être effectuée en comparant les attributs communs des sources de données à matcher, telles que des noms, des adresses ou des dates de naissance. Les valeurs de ces attributs sont souvent de qualité médiocre, car elles peuvent contenir des erreurs typographiques et des variations ou peuvent changer à travers le temps. Par conséquent, des mesures de similarités approximatives (telles que Jaccard, Jaro, Jaro-Winkler, etc.) sont généralement utilisées pour comparer les paires d'enregistrements (Christen, 2012a; Elmagarmid et al., 2007)[1][2]. Le résultat de la comparaison de chaque paire d'enregistrements est un vecteur de comparaison qui contient les résultats des mesures de similarité. Enfin, les vecteurs de comparaison représentant les paires d'enregistrements sont classés en matches (où ils sont supposés correspondre à la même entité du monde réel) et non-matches (où ils sont supposés correspondre à des entités différentes) en utilisant un modèle de classification.

Il existe deux approches principales pour l'ER: les approches basées sur les règles et celles basées sur l'apprentissage (Christen, 2012a; Elmagarmid et al., 2007)[1][2]. Les approches basées sur les règles utilisent des règles de matching pour matcher les enregistrements. Ces règles sont élaborées par un expert humain et sont limitées à des domaines bien précis. Les approches basées sur les règles sont manuelles et ne peuvent être utilisées dans tous les domaines. Les approches basées sur l'apprentissage utilisent des données d'apprentissage (ensemble de vecteurs de comparaison étiquetés comme matches ou non matches) pour apprendre un modèle de classification (règles de matching ou fonction de combinaison numérique) qui est utilisé par la suite à prédire le statu des paires d'enregistrement à comparer. Dans des situations réelles, les données d'apprentissage ne sont pas disponibles ou doivent être élaborées par un expert.

1.3 Motivation et objectifs

l'objectif principal est de donner un aperçu des différentes techniques de couplage d'enregistrements disponibles et de leur fonctionnement. les objectifs suivants ont été fixés.

Objectif 1 Contexte et étude théorique du couplage d'enregistrements.

Objectif 2 Évaluation des mesures de similitude et des méthodes de couplage d'enregistrements.

Objectif 3 Évaluation des performances à l'aide des ensembles de données FEBRL[3].

les deux premiers objectifs seront atteints grâce à une étude de recherches similaires dans le domaine.

Une fois le contexte et les techniques présentés, le troisième objectif sera atteint grâce à un ensemble d'expériences empiriques qui évaluent les performances de ces techniques, en utilisant la boîte à outils de python record-linkage pour l'implémentation .

1.5 Organisation du mémoire

Le reste de ce mémoire est organisé comme suit. Le chapitre 2 fournit une introduction aux concepts et technologies utilisés dans tous les chapitres restants. Il présente le concept de qualité des données, la procédure d'intégration des données. Le chapitre 3 présente la théorie et l'approche générale du couplage d'enregistrements, des données aux résultats,.cela est nécessaire pour comprendre la méthodologie et les résultats présentés dans les sections suivantes. Au chapitre 4 présente la méthode proposée pour faire l'évaluation , les résultats sont présentés, en commençant avec une explication des paramètres d'évaluation. Une fois les ensembles de données visualisés et étudiés, les résultats des méthodes de couplage d'enregistrements sont présentés avec différentes métriques. Enfin les conclusions des résultats sont présentées et discutées.

Chapitre 2

Préliminaires

2.1 Introduction

Dans ce chapitre,. Nous fournissons un aperçu sur la qualité de données dans la section 2.2. Ensuite, la procédure d'intégration des données sera abordée dans la section 2.3.

2.2 Qualité de données

La qualité des données est un terme générique décrivant à la fois les caractéristiques de données : complètes, fiables, pertinentes, à jour et cohérentes, mais aussi l'ensemble du processus qui permet de garantir ses caractéristiques. Le but est d'obtenir des données sans doublons, sans fautes d'orthographes, sans omission, sans variation superflue et conforme à la structure définie.

Des données de mauvaise qualité peuvent entraîner des résultats erronés au moment de l'interrogation ou au moment de la prise de décision. Des statistiques récentes révèlent que les entreprises estiment généralement des taux d'erreur de données d'environ 1% à 5% et que les données de mauvaise qualité coûtent aux entreprises US environ 600 milliards de dollars par an .

La qualité des données peut être mesurée par des indicateurs (mesures) tels que le nombre de valeurs distinctes, ou encore le nombre de valeurs en doubles. Ces indicateurs permettent d'évaluer des dimensions de la qualité des données telles que l'intégrité, la standardisation ou la déduplication.

Une dimension de qualité des données est une caractéristique utilisée pour classer les besoins de données. En fait, une dimension offre un moyen pour mesurer et gérer la qualité des données .

Il y'a un certain nombre de divergences dans la définition de la plupart des dimensions en raison de la nature contextuelle de la qualité des. Plusieurs dimensions ont été identifiées pour la qualité de données (Sidi et al., 2012)[41] telles que l'exactitude (accuracy), la complétude (completeness), la consistance (consistency). Toutefois, il n'existe aucun accord général soit sur quel ensemble de dimensions définit la qualité des données, ou sur le sens exact de chacune des dimensions. Nous rappelons brièvement, ci-dessous, les définitions des dimensions essentielles.

Consistance

La consistance se réfère à la violation des règles sémantiques définies sur l'ensemble des données . Dans le modèle relationnel, les contraintes d'intégrités sont des instanciations de ces règles sémantiques.

Complétude

La capacité d'un système d'information à représenter chaque état significatif du monde réel représenté (peu de valeurs absentes ou inconnues) .

Exactitude (Accuracy)

Les données sont exactes lorsque les valeurs des données stockées dans les sources de données correspondent à celles du monde réel .

Duplication

Les données sont répétées. L'entité est gérée par plusieurs systèmes d'informations sous des identifiants différents et donc sa vue n'est pas unifiée.

2.3 Intégration de données

L'intégration de données consiste à combiner les données de plusieurs sources de données hétérogènes pour fournir une vue unifiée des données disponibles à une application ou à un utilisateur final. L'un des avantages des systèmes d'intégration de données est que l'utilisateur d'un tel système obtient une vue d'ensemble complète et concise de toutes les données existantes sans devoir accéder à toutes les sources de données séparément. Elle est complète car aucune entité ou donnée ne manque dans le résultat. Elle est concise car aucune entité n'est représentée deux fois et les données présentées à l'utilisateur sont sans contradiction .La figure 2.1 montre un exemple simple d'intégration de données qui contient des enregistrements de deux sources de données bibliographiques, à savoir, ACM et DBLP. Un système d'intégration de données doit en premier lieu identifier les paires d'enregistrements faisant référence aux mêmes publications, qui sont dans ce cas (564753, conf/sigmod/BumbulisB02) et (872806, conf/sigmod/LometT03). Ensuite, les fusionner pour avoir seulement trois enregistrements au lieu de cinq.

Dans les environnements distribués, les sources de données sont généralement caractérisées par divers types d'hétérogénéités qui peuvent généralement être classées en (i) hétérogénéités des schémas et (ii) hétérogénéités au niveau des instances.

Un système d'intégration de données complet est complexe et nécessite une collaboration entre plusieurs domaines de recherche. La figure 2.2 illustre les trois composants principaux d'un système d'intégration de données. L'alignement de schémas permet de résoudre les hétérogénéités des schémas en détectant les éléments équivalents de schémas dans des sources de données différentes et en particulier les attributs équivalents.

La résolution d'entité et la fusion de données permettent de résoudre les hétérogénéités au niveau des instances. La résolution d'entité permet la détection des descriptions d'entités qui font référence aux mêmes entités du monde réel dans des situations où les identifiants d'entité ne sont pas disponibles. Enfin, la fusion de données permet de résoudre le problème des valeurs d'attributs contradictoires et de représenter les descriptions d'entité équivalentes en une seule description.

ACM

ID	Title	Authors	Venue	Year
564753	A compact B-tree	Peter Bumbulis, Ivan T. Bowman	International Conference on Management of Data	2002
872806	A theory of redo recovery	David Lomet, Mark Tuttle	International Conference on Management of Data	2003

DBLP

ID	Title	Authors	Venue	Year
conf/sigmod/BumbulisB02	A compact B-tree	Ivan T. Bowman, Peter Bumbulis	SIGMOD Conference	2002
conf/sigmod/LometT03	A Theory of Redo Recovery	Mark R. Tuttle, David B. Lomet	SIGMOD Conference	2003
conf/sigmod/DraperHW01	The Nimble Integration Engine	Daniel S. Weld, Alon Y. Halevy, Denise Draper	SIGMOD Conference	2001

Figure 2.1: *Un exemple d'intégration de données dans le domaine bibliographique*

2.3.1 Alignement de schémas

Les systèmes d'intégration de données doivent généralement faire face aux schémas hétérogènes des sources de données à intégrer. Afin de présenter les résultats d'une requête à l'utilisateur dans un schéma unique, les hétérogénéités schématiques doivent être gérées. Pour faire face à l'hétérogénéité et parvenir à l'interopérabilité, deux méthodes sont utilisées le matching des schémas et le mapping des schémas .

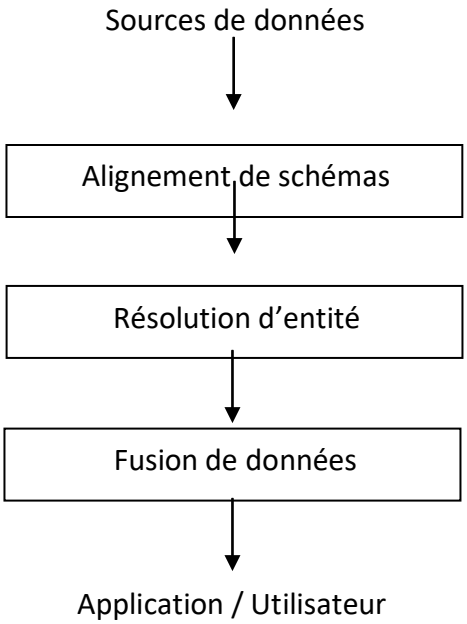

Figure 2.2: *composants principaux d'un système d'intégration de données.*

Comme le montre la figure 2.3, l'alignement des schémas comprend en général trois étapes: création d'un schéma de médiation, matching des schémas et mapping des schémas.

En premier lieu, un schéma médiateur est créé par un expert du domaine pour avoir une vue unifiée et virtuelle des sources de données à intégrer. Le schéma médiateur permet à un utilisateur final ou à une application de poser des requêtes au système d'intégration de données sans connaitre les schémas des sources de données à intégrer.

La deuxième étape dans l'alignement des schémas est le matching des 0 schémas. Elle consiste à matcher les attributs de chaque schéma source avec les attributs correspondants du schéma médiateur. Dans de nombreux cas, la correspondance d'attribut est un à un; toutefois, un attribut du schéma médiateur peut parfois correspondre à la combinaison de plusieurs attributs d'un schéma source et inversement. Plusieurs techniques ont été proposées pour le matching d'attributs, en explorant la similarité entre les noms d'attributs, les types, les valeurs et les relations de voisinage entre les attributs. Ces techniques ont été revues par (Rahm & Bernstein, 2001)[42], (Bellahsene, Bonifati & Rahm, 2011) [43]et (Euzenat & Shvaiko, 2007)[44].

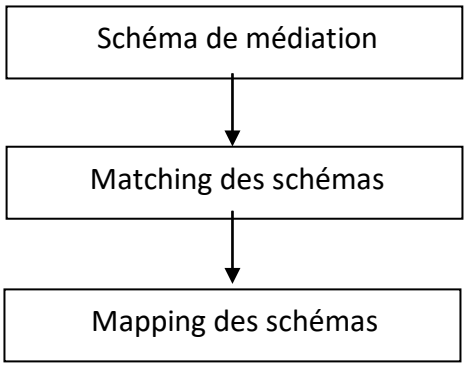

Figure 2.3: *Etapes d'alignement des schémas.*

Enfin, l'étape mapping des schémas utilise les correspondances d'attributs générées par l'étape matching des schémas pour construire un mapping des schémas entre chaque schéma source et le schéma médiateur. De tels mappings spécifient les relations sémantiques entre les contenus des différentes sources de données et seraient utilisés pour reformuler une requête utilisateur sur le schéma médiateur en un ensemble de requêtes sur les sources de données sous-jacentes. Il existe trois types de mapping de schémas: GAV (global as-view), LAV (local-as-view) et GLAV (global-local-as-view). GAV spécifie comment obtenir des données dans le schéma médiateur en interrogeant les données dans les schémas sources; En d'autres termes, les données du schéma médiateur peuvent être considérées comme une vue,

de base de données des données sources.

LAV spécifie les données source en tant que vue des données du schéma médiateur; Cette approche facilite l'ajout d'une nouvelle source de données avec un nouveau schéma. Enfin, GLAV spécifie à la fois les données du schéma médiateur et les données des sources de données sous forme de vues de données d'un schéma virtuel. Clio (Fagin et al., 2009)[45] est un exemple d'outils qui permet de construire de manière semi-automatique des mapping de schémas en utilisant les résultats du matching d'attributs. La figure 2.4 montre un exemple d'alignement de schémas.

r_id	Nom de famille	prénom	Adresse
R1	Brahmi	Sara	Cité 140 logement bloc A5 n=8 saida
R2	Bellout	Lina manel	Cité 114 logement bloc B5 n=8 saida

S_id	Nom	prénom	Domicile
S1	Brahimi	sarah	Cité 140 logement bloc A5 n=8 saida
S2	Belout	Manel lina	Cité 114 logement bloc B5 n=8 saida

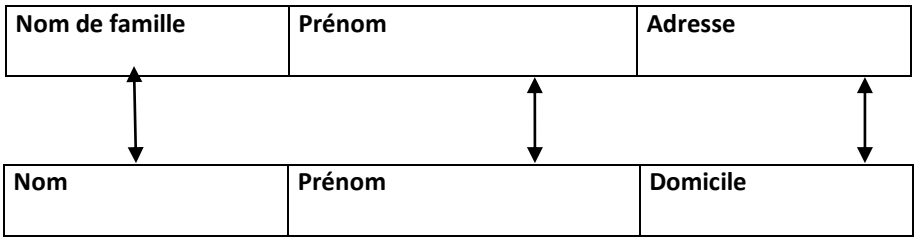

Figure 2.4: *Exemple d'alignement de schémas.*

2.3.2 Résolution d'entité

La seconde étape dans le processus d'intégration des données est l'étape de résolution d'entité (également appelée record linkage (Elfeky et al., 2002)[27], détection des doublons (Elmagarmid et al., 2007; Naumann & Herschel, 2010)[2], data matching (Christen, 2012a) [1]et bien d'autres). L'objectif de cette étape est d'identifier les enregistrements qui représentent la même entité du monde réel. Si les enregistrements ont des identifiants uniques et sans erreur, tels que des numéros de sécurité sociale, la résolution d'entité devient un processus simple pouvant être facilement effectué par une simple opération de jointure de base de données standard. Cependant, dans de nombreux scénarios pratiques, un tel identifiant unique n'existe pas. La figure 2.1 montre un exemple dans le domaine bibliographique dans lequel les sources de données ACM et DBLP utilisent des identifiants différents. Dans ce cas, le processus de résolution d'entité doit être effectué en utilisant des comparaisons approximatives des attributs correspondants des enregistrements. Il est également intéressant de noter que les mêmes données peuvent être représentées de différentes manières dans différentes sources de données en raison de facteurs tels que des conventions différentes, des erreurs typographiques, des valeurs manquantes ou encore aberrantes et obsolètes.

A première vue, la résolution d'entité est simple: comparer chaque paire d'enregistrements en utilisant une mesure de similarité. Si la similarité est supérieure ou égale à un seuil donné, les deux enregistrements sont déclarés comme étant des doublons. En réalité, il y a deux difficultés principales à résoudre: l'efficacité et la mise en échelle. L'efficacité est principalement influencée par la qualité de la mesure de similarité et le choix d'un bon seuil global de similarité. La mise en échelle est un problème dû au volume des sources de données à matcher. Si les sources de données sont très volumineuses, le calcul de la similarité de tous les pairs d'enregistrements peut devenir un obstacle. Le chapitre 3 détaille beaucoup plus cette étape de résolution d'entité.

2.3.3 Fusion de données

L’étape finale dans le processus d'intégration de données est la fusion de données (Bleiholder & Naumann, 2009)[46]. Elle consiste à consolider et à fusionner les paires ou les groupes d'enregistrements faisant référence aux mêmes entités en une seule représentation cohérente et complète pour chaque entité. La fusion de données permet l’enrichissement des enregistrements : renseigner les valeurs nulles, enrichir les données (une personne peut avoir deux numéros de téléphone différents ou plus), corriger les formats des données.

Le principal défi de la fusion des données est la résolution des conflits au niveau des données lorsque les enregistrements correspondant à une entité contiennent des valeurs d'attribut différentes (par exemple contradictoires). La résolution des conflits consiste à choisir une (ou plusieurs) valeur parmi un ensemble de valeurs contradictoires.

Les approches de résolution de conflits sont classées en deux grandes catégories (Bleiholder & Naumann, 2009) [46]: les approches basées sur les instances et les approches basées sur les métadonnées. Les approches basées sur les instances considèrent uniquement les valeurs de données contradictoires pour choisir la valeur correcte comme le choix de la valeur la plus commune, le calcul d’une moyenne, le calcul du minimum ou le calcul du maximum parmi les valeurs contradictoires . Les approches basées sur les métadonnées choisissent les valeurs en se basant sur les métadonnées telles que la fraîcheur des données (la valeur la plus récente) ou la fiabilité d'une source
par apport a un autre.

2.4 Conclusion

De nos jours, l'intégration de données est utilisée dans plusieurs domaines pour présenter à l'utilisateur final ou à une application une vue unifiée de plusieurs sources de données distribuées et hétérogènes. La résolution d'entité est une étape importante dans l'intégration et le nettoyage des données, elle permet d'améliorer la qualité des données en détectant les descriptions qui représentent la même entité du monde réel.

Chapitre3

Résolution d'entités

3 Introduction

ce chapitre décrit une vue d'ensemble des méthodes de couplage d'enregistrements. Il comprend quatre sous-sections; prétraitement des données, méthodes d'indexation, types de méthodes de comparaison et enfin une section sur les méthodes de classification. ce chapitre théorique est destiné à fournir au lecteur suffisamment de connaissances pour comprendre les expériences et l'évaluation présentées dans les chapitres suivants. l'approche générale et le pipeline de couplage d'enregistrements peuvent être vus ci-dessous dans la figure 3.1 .

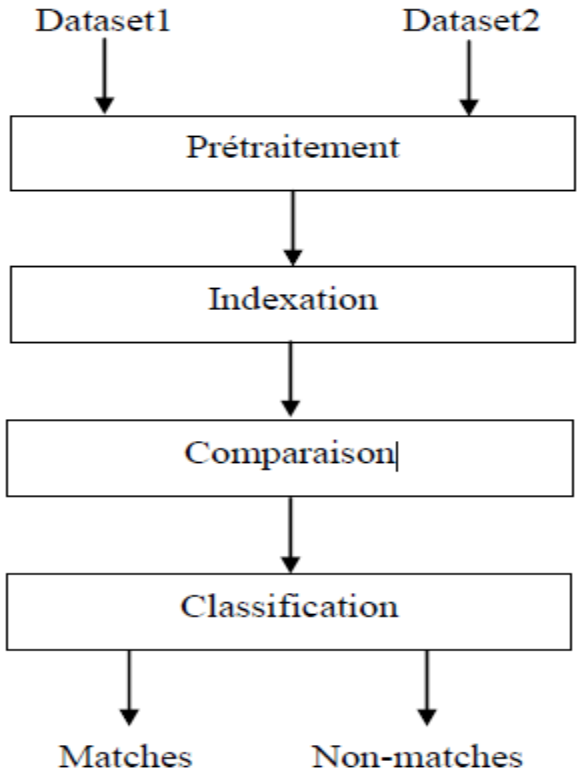

figure 3.1: *composants principaux d'un système de résolution d'entité*

3.1 Prétraitement

La plupart des données collectées dans le monde réel sont de mauvaise qualité et peuvent inclure des erreurs, des valeurs manquantes ou des valeurs incohérentes (Batini & Scannapieco, 2006; Herzog et al., 2007)[47][48]. De plus, les sources de données à matcher peuvent avoir des structures et des formats différents. Le problème des structures différentes (ou hétérogénéités structurelles) est résolu dans l'étape alignement de schémas qui est la première étape de l'intégration de données, alors que le problème des formats différents de données est résolu dans l'étape nettoyage de données. La qualité des données est vitale pour le processus d'ER et affecte de manière significative ses résultats (Herzog et al., 2007; Christen, 2012a; Elmagarmid et al., 2007). L'étape de prétraitement est effectuée en utilisant des techniques de nettoyage et de normalisation de données qui reposent fortement sur des tables de consultation (look-up Tables). Ces tables contiennent, par exemple, des noms personnels et leurs variantes, ainsi que des fautes d'orthographe courantes, des noms de banlieue, de ville ou d'état et des codes postaux d'un pays donné. D'autres techniques utilisent des règles de standardisation élaborées manuellement (Christen, 2012a). Les tables et les règles sont dépendantes du domaine et de la langue des sources de données à matcher (Yousef, 2015).

Les principales approches couramment utilisées pour nettoyer et standardiser les données lors de la première étape du processus d'ER (Elmagarmid et al., 2007) sont les suivantes:

Standardisation: Avant que les valeurs d'attributs ne soient utilisées dans le processus d'ER, les attributs doivent être convertis en un format standard. La standardisation consiste généralement à supprimer les caractères indésirables, à corriger les fautes d'orthographe, à élargir les abréviations et à s'assurer que les mêmes systèmes de codage et unités de mesure sont utilisés dans les sources de données à matcher .

Parsing: L'analyse syntaxique vise à identifier et à isoler les composants individuels dans les valeurs d'attributs en composants cohérents et bien définis (Christen, 2012a; Elmagarmid et al., 2007; Winkler, 1995)[1][2][49]. Un exemple de parsing consiste à diviser le numéro de téléphone en un code pays, un indicatif régional et un numéro de téléphone.

Transformation de données: Dans cette étape, les valeurs des attributs sont converties dans leur forme correcte (Christen, 2012a; Elmagarmid et al., 2007; Rahm & Do, 2000). Les conversions courantes dans cette étape incluent le décodage des valeurs d'attribut codées dans leur forme d'origine, la vérification de la plage et la vérification des dépendances. Un exemple de vérification de dépendance consiste à valider le nom d'une ville et son code postal. Si les deux valeurs ne sont pas cohérentes, cela correspond généralement à une erreur de saisie de données qui doit être corrigé en modifiant une des deux valeurs d'attribut pour qu'elle soit cohérente.

La quantité de recherches dans le domaine du prétraitement des données (nettoyage et normalisation) est étonnamment faible. La plupart des systèmes d'ER ignorent cette étape. Une des raisons à cela pourrait être que le prétraitement des données est une tâche très spécifique à un domaine qui implique des quantités importantes d'expertise du domaine et de personnalisation et d'intervention manuelle (Christen, 2012a).

3.2 Méthodes d'indexation

Étant donné deux sources de données R et S, représentées de manière générique sous la forme d'ensembles d'enregistrements, un système d'ER naïf évaluerait toutes les paires d'enregistrements possibles. Chaque enregistrement de R est comparé en détails avec tous les enregistrements de S en utilisant des mesures de similarité qui sont coûteuses. En supposant un coût constant par évaluation, le temps d'exécution serait $O(|R||S|)$. Pour des sources de données volumineuses, il est impossible de comparer toutes les paires d'enregistrements possibles (Christen & Goiser, 2007)[50]. L'étape d'indexation a pour but de réduire le nombre important de comparaisons potentielles en

supprimant autant que possible les paires d'enregistrements qui correspondent aux non-matches et en regroupant les paires d'enregistrements qui partagent certaines caractéristiques en blocks.

L'ER est alors limitée aux enregistrements du même bloc. Le résultat de l'indexation est un ensemble de paires d'enregistrements appelées *paires d'enregistrements candidates* qui seront comparées en détails dans l'étape comparaison. Un exemple d'indexation simple est de regrouper tous les enregistrements qui partagent les trois premières lettres du nom en blocs.

L'indexation utilise une fonction appelée clé de blocage (blocking key) pour regrouper les entités similaires en blocs. Étant donné une source de données R représentée comme un ensemble d'enregistrements, une clé de blocage K est une fonction qui prend un enregistrement de R comme entrée et renvoie un ensemble non vide de valeurs, appelé valeurs de clé de blocage (BKVs) de l'enregistrement. Sans perte de généralité, les valeurs renvoyées par la clé de blocage K sont supposés être des chaînes de caractères. La clé de blocage est généralement déterminée manuellement.

Dans ce qui suit, nous présentons les principales approches d'indexation (Christen, 2012b; Draisbach & Naumann, 2009; Papadakis, Svirsky, Gal & Palpanas, 2016).

Blocage standard

Le blocage standard ou traditionnel (Fellegi & Sunter, 1969) représente chaque enregistrement par une valeur de clé de blocage (BKV). Tous les enregistrements avec la même BKV sont insérés dans le même bloc.

La manière de définition des clés de blocage influencera la qualité et le nombre des paires candidates générées. Un inconvénient majeur est que des clés de blocage dont les valeurs contiennent des erreurs entraineront une diminution de la complétude des paires (les matches sont affectés à des blocs différents et ne seront pas détectés dans la phase de comparaison). Un second inconvénient est que les tailles des blocs générés dépendent de la distribution de fréquence des BKVs, et cela peut entrainer la génération

de blocs larges contenant un nombre important de comparaisons.

La figure 3.2 illustre la fonctionnalité du blocage standard pour quatre enregistrements. La clé de blocage utilisé est une concaténation des trois premières lettres du nom avec les trois premières lettres du prénom.

Dans cet exemple, deux blocs sont générés et qui sont représentés par les deux BKVs : GEO_CAR et WIL_GEO. Dans l'exemple de la figure 3.2, l'ensemble final des paires d'enregistrements candidates sera {(R1,R3), (R2,R4)}.

ID	Nom	Prénom	BKVs
R1	WILLIAM	GEORG	WIL_GEO
R2	GEORGE	CARTIER	GEO_CAR
R3	WILIAM	GEORGE	WIL_GEO
R4	GEORG	CARTER	GEO_CAR

WIL_GEO
R1 R3

GEO_CAR
R2 R4

figure 3.2 : *Exemple de blocage standard avec BK étant les trois premières lettres du non concaténées avec les trois premières lettres du prénom*

Voisinage trié (Sorted Neighborhood)

Une autre méthode de blocage influente qui a été fondamentalement conçue pour garantir une limite de la taille de l'ensemble des paires candidates est la méthode du voisinage trié en anglais Sorted Neighborhood (SN) (Hernández & Stolfo, 1995)[51]. Cette méthode utilise les mêmes BKVs que le blocage standard, mais construit des blocs en se basant sur leur similarité au lieu de leur égalité. Elle trie les BKVs par ordre alphabétique et glisse une fenêtre d'un nombre fixe d'enregistrements w ($w > 1$) sur les valeurs triées. À chaque étape, Les paires d'enregistrements candidates sont générées uniquement à partir des enregistrements de la fenêtre en cours.

La fenêtre glissante a deux implications pour la génération des paires candidates. Tout d'abord, les enregistrements qui n'ont pas la même BKV peuvent être insérés dans

le même bloc. Deuxièmement, certains enregistrements avec la même BKV peuvent être insérés dans des blocs différents.

L'algorithme SN est sensible à la taille de la fenêtre *w*, les petites fenêtres entraînent un rappel faible, mais les grandes entrainent de nombreuses comparaisons et une précision faible. Plusieurs variations de cette technique existent (Christen, 2012b).

Les principales différences entre ces versions et la version d'origine sont les différentes méthodes pour régler le paramètre de la fenêtre glissante (e.g., adaptatif ou constant) pour des performances maximales (Yan et al., 2007). Une autre extension de l'algorithme SN est l'approche Multi-Pass (Hernández & Stolfo, 1998)[52] qui vise à surmonter le problème de clé de blocage unique sensible aux erreurs. Elle consiste à exécuter l'algorithme plusieurs fois avec plusieurs clés de blocage différentes.

La figure 3.3 illustre la fonctionnalité de l'indexation voisinage trié pour les quatre enregistrements présentés dans la figure 3.3. La clé de blocage utilisée est le prénom et la taille de la fenêtre *w*=3. L'intervalle de la fenêtre représente les paires d'enregistrements sélectionnés comme candidats pour chaque glissement de la fenêtre. Dans l'exemple de la figure 2.7, l'ensemble final des paires d'enregistrements candidates sera {(R4,R2), (R4,R1), (R2,R1), (R2,R3), (R1,R3)}.

Position fenêtre	BKVs (Prénom)	ID
1	CARTER	R4
2	CARTIER	R2
3	GEORG	R1
4	GEORGE	R3

Intervalle fenêtre	Paires candidates
1-3	(R4,R2), (R4,R1), (R2,R1)
2-4	(R2,R1), (R2,R3), (R1,R3)

figure 3.3 : Exemple d'indexation voisinage trié avec BK étant prénom à partir de la figure 2.2 et taille de la fenêtre w=3.

3.3 Comparaison

La comparaison des champs joue un rôle central dans la liaison des enregistrements. c'est la partie où la similitude réelle entre les différents domaines est évaluée.

il existe de nombreux algorithmes différents pour comparer les champs, mais ils peuvent être divisés en quelques catégories, voir la figure 3.4. cette recherche se concentrera sur la comparaison basée sur les caractères et sur les jetons.

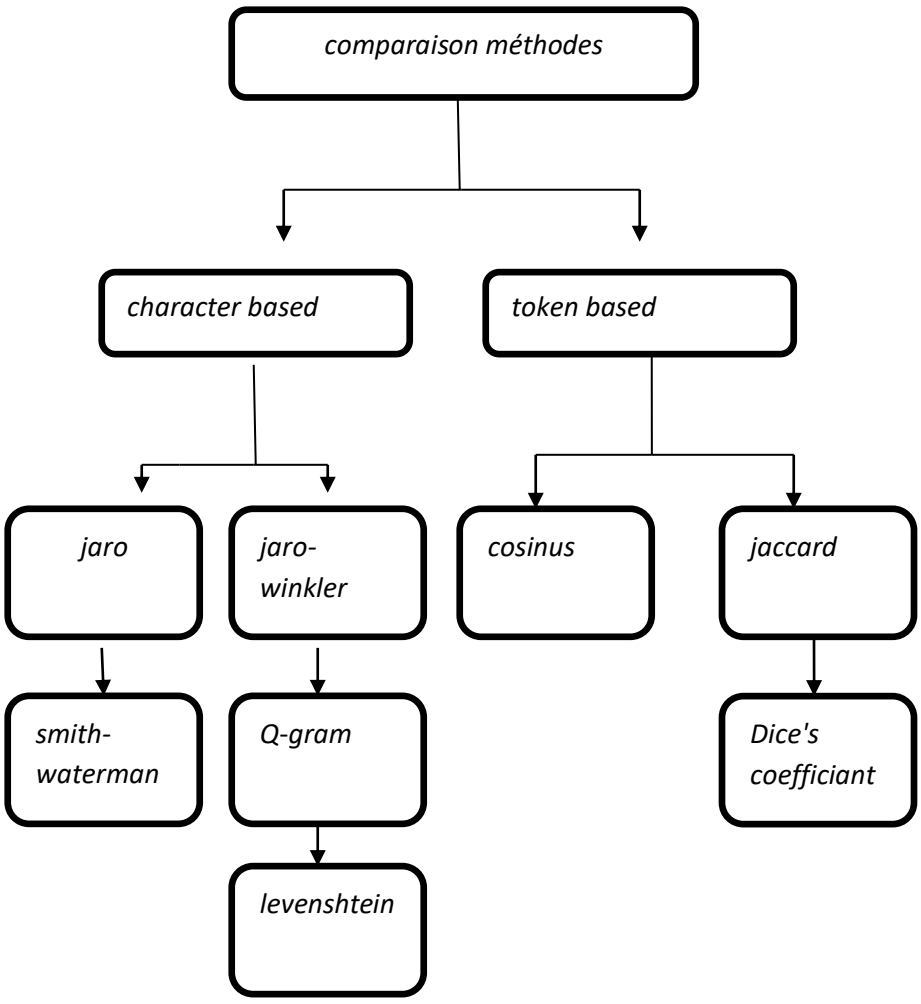

figure 3.4 : Types de méthodes de comparaison

3.3.1 Basé sur les caractères

Les algorithmes de comparaison basés sur les caractères conviennent aux cas où des erreurs typographiques se produisent. la base de bon nombre de ces algorithmes est de découvrir à quel point il est difficile de passer d'une chaîne à une autre et sont destinés à être utilisés avec des données de chaîne telles que des noms ou des adresses. Tous les algorithmes renvoient une valeur entre 0 et 1, 0 représentant une dissemblance totale et 1 étant une correspondance exacte. cette section couvrira un aperçu des algorithmes suivants:

- Jaro
- Jarowinkler
- Levenshtein
- Q-gram
- Smith-Waterman

3.3.1.1 jaro

La mesure de Jaro se base sur le nombre et l'ordre des caractères communs entre les deux chaînes de caractères. Ainsi, la mesure de Jaro entre S et T est définie par l'équation (3.4.1).

$$Sim_{jo}(A,B) = \frac{1}{3}\left(\frac{m}{|A|} + \frac{m}{|B|} + \frac{m - t/2}{m}\right)$$

avec :

m :est le nombre de caractères de A communs avec B ; un caractère a_i de A est dit en commun avec b s'il existe bj dans B tel que ai = bj et i − H ≤ j ≤ i + H, où H = 1 2 . min(A, B);

$t/2$: est la moitié du nombre de transpositions de A et B , où une transposition de A et B est une position i tel que $a_i \neq b_i$.

Exemple 1:

Soient les chaînes de caractères : S="*Xerox*" et T ="*Xerox0*". La mesure de Jaro entre S *et* T

est donnée par :

$$Jaro(S,T) = \frac{1}{3}.(\frac{5}{6} + \frac{5}{5} + \frac{5-0.5}{5}) = 0.92$$

3.3.1. 2 Jaro-Winkler

Winkler et thibaudeau ont créé une version modifiée de l'algorithme Jaro qui améliore les performances des noms en particulier. la version modifiée donne plus de poids aux correspondances de préfixe, qui sont plus importantes pour faire correspondre les noms de famille le raisonnement pour changer ce poids provient d'études empiriques qui ont montré que les erreurs sont moins susceptibles de se produire au début des noms. la similitude de Jarowinkler est donnée par l'aquation (3.4.2) :

$$Sim_{jw}(A,B) = Sim_{jo}(A,B) + \frac{c}{10}\Big(1.0 - Sim_{jo}(A,B)\Big)$$

où c est le nombre de caractères d'accord au début de chaque chaîne (par exemple, James et Jamie ont une valeur de c = 3 puisque les trois premières lettres sont d'accord) (maximum 4 caractères).

Exemple 2:

Soient les chaînes de caractères : *S*=“*Xerox*” et *T* =“*Xerox0*”. La mesure de Jarowinkler entre *S et T* est donnée par :

jaroW(S,T)=0.92+4/10(1-0.92)=0.952

3.3.1.3 Levenshtein

la distance de Levenshtein, également connue sous le nom de distance de modification, est la plus faible quantité d'insertions, de suppressions et de substitutions nécessaires pour convertir une chaîne A en une autre chaîne B. l'algorithme Levenshtein fournira un nombre dans la plage [0, max (| A |, | B |)]. ce nombre sera le

nombre d'opérations nécessaires pour changer A en B, ou zéro si les chaînes sont égales. la distance de Levenshtein, Ld, est définie par l'équation (3.4.3):

$$Sim_{ld}(A,B) = 1.0 - \frac{Levenshtein(A,B)}{max(|A|,|B|)}$$

Exemple 3:

Soient les trois chaînes de caractères : S="Xerox", T ="Xerox0" et U ="Xeroxfs". Les mesures de Levenshtein entre S, T et S, U sont données par :

$$Levenshtein(S,T) = 1 - \frac{coût(\epsilon \rightarrow \text{`0'})}{|T|} = 1 - \frac{1}{6} = 0.83$$

$$Levenshtein(S,U) = 1 - \frac{coût(\epsilon \rightarrow \text{`f'}) + coût(\epsilon \rightarrow \text{`s'})}{|U|} = 1 - \frac{2}{7} = 0.71$$

3.3.1.4 Q-grammes

Les Q-grammes, également connus sous le nom de n-grammes, sont des sous-séquences de longueur q dans des chaînes plus grandes. la valeur 'q' définit la longueur de ces sous-séquences et a généralement la taille de un, deux ou trois.la logique derrière l'idée de q-grammes est que des chaînes similaires devraient avoir de nombreux q-grammes en commun et une approximation de leur similitude. les q-grammes peuvent être obtenus en divisant une chaîne en toutes les sous-séquences possibles de longueur q. les q-grammes résultants peuvent alors être comparés à l'ensemble de q-grammes appartenant à une autre chaîne.

En comptant les q-grammes en commun, une mesure de similitude peut être calculée en divisant les q-grammes correspondants par le nombre de q-grammes dans la chaîne la plus longue. la similitude Q-gram est définie comme l'équation(3.4.5):

$$Sim_{qg}(A, B, q) = \frac{|Qgram(A, q) \cap Qgram(B, q)|}{max(|Qgram(A, q)|, |Qgram(B, q)|)}$$

3.3.1.5 Smith-Waterman

l'algorithme de Smith-Waterman a été développé en 1981 dans le but de comparer les séquences de protéines et l'ADN. Elle est similaire à la distance levenshtein / edit, mais permet des écarts et des scores de correspondance spécifiques aux personnages. cela permet un système de score où des personnages similaires obtiendraient un score inférieur à un match complet, mais un score supérieur à un score négatif ou nul.

Un exemple serait l'utilisation des tables de transformation Soundex. Soundex est une bibliothèque phonétique qui classe comment des caractères ou des chaînes similaires sont basés sur leur son.

cela pourrait donner aux caractères «m» et «n» un score positif car ils semblent similaires, mais les caractères «o» et «j» obtiendraient un score négatif car ils ne semblent pas similaires. Smith-Waterman tient également compte des lacunes lors de la comparaison de deux chaînes et permet une notation différente des lacunes de différentes longueurs.

3.3.2 Les mesures Basé sur des jetons (à base de sacs de mots)

Les mesures de similitude basées sur les caractères conviennent lorsque les erreurs typographiques sont courantes. Cependant, lorsque les données comparées ont des conventions différentes, par exemple, "John, Doe" et "Doe, John", les algorithmes basés sur les caractères ne parviennent pas à capturer leurs similitudes. Les algorithmes basés sur les jetons tiennent compte de ces conventions en utilisant des techniques qui permettent différentes conventions de données. Tout comme les mesures de similarité basées sur les caractères, les algorithmes basés sur les jetons renvoie également une valeur comprise entre zéro et un, zéro étant une dissemblance totale et un étant une correspondance exacte. cette section explique deux mesures de similitude, à savoir:

* Dice’s coefficiant * Jaccard *cosinus

3.3.2. 1 Jaccard

la similitude Jaccard est utilisée avec une technique de séparation de mots telle que les q-grammes, pour générer une mesure entre deux chaînes quelle que soit leur convention. En transformant deux chaînes A et B en deux ensembles de jetons, Sa et Sb, la similitude Jaccard est calculée par l'équation (3.4.6)

$$Sim_{jc}(S_a, S_b) = \frac{|S_a \cap S_b|}{|S_a \cup S_b|}$$

Exemple 4 :

Soient les chaînes de caractères *S*=“Xerox financial services”, *T* = “Financial services xerox”,

U=“Xerox financial servicex”et *V* =“Xerox finacial services”.

Les mesures de Jaccard entre *S*, *T* et *U* , *V* sont données par

$$Jaccard(S, T) = \frac{3}{3} = 1$$

$$Jaccard(U, V) = \frac{1}{5} = 0.2$$

V est faible (0.2) car elle utilise une comparaison stricte entre les mots et ne tolère donc pas les erreurs de saisie .

3.3.2.2 Dice’s coefficiant

Le coeficiant de Dice est similaire à la similitude de Jaccard, mais n'exige pas que les termes soient uniques . par conséquent, les jetons répétés seront toujours comptés, donnant ainsi un score final plus élevé. Soit Sa et Sb les termes générés à partir de deux chaînes A et B, le coeficiant de Dice peut alors être calculé par

l'équation(3.4.7)

$$Sim_{dc}(S_a, S_b) = 2 * \frac{|S_a \cap S_b|}{|S_a| + |S_b|}$$

Exemple 5 :

Soient les chaînes de caractères S="Xerox financial services", T = "Financial services xerox", U="Xerox financial servicex"et V ="Xerox finacial services".

Les mesures de dice entre S, T et U , V sont données par

Dice(S,T)= 2*(3/6)=1

Dice(U,V)=2*(1/6)=0.33

3.3.2.3 Mesure Tf-idf (ou Cosinus)

La mesure Tf-idf [Jones72] est très utilisée dans le domaine de recherche d'information et se base sur la fréquence et l'importance des termes en commun entre les deux chaînes relativement à un corpus ou une collection de mots. Soient S et T deux chaînes de caractères composées d'ensembles de mots. $Tf\ (w,S)$ représente le nombre d'occurrences du terme w dans la chaîne de caractères S. $Idf\ w$ représente le nombre total d'enregistrements dans la base de données divisé par le nombre d'enregistrements qui contiennent au moins une occurrence de w. La mesure Tf-idf est donnée par l'équation (3.4.8).

$$Tf\text{-}idf(S,T) = \sum_{w \in S \cap T} V(w,S).V(w,T)$$

avec $V(w,S) = V'(w,S)/\sqrt{\sum_{w \in S} V'(w,S)^2}$ et $V'(w,S) = \log(Tf_{w,S} + 1).\log(Idf_w)$.

Exemple 6 :

t	Nom	Adresse	Code postal	Ville	Téléphone
Enregistrement 1	Xerox	24 rue de la chapelle	97122	Baie Ma-Hault	0825082081
Enregistrement 2	Xerox0	(null)	(null)	(null)	(null)
Enregistrement 3	Xeroxfs	CDG	92000	Neuilly-sur-Seine	0825082090
Enregistrement 4	Xerox financial Servicex	120 avenue Charles de Gaulle	92202	Neuilly	0825082081
Enregistrement 5	Xerox finacial Services	Charles de gaulle	92202	Neuilly	0825082081
Enregistrement 6	Xerox financial Sces	120 av. Charles de Gaulle	92000	Neuilly sur Seine	0825082081
Enregistrement 7	Xerox financial Services	120 av Charles de Gaulle	92202	Neuilly	0825082090
Enregistrement 8	Financial services xerox	Av Charles de gaulle	92000	Neuilly sur Seine	(null)

a b l e a u 3 . 1 :

Échantillon de 8 enregistrements de la table des entreprises+

Soient les chaînes de caractères S=“Xerox”, T = “Xerox financial servicex”

et U =“Xerox finacial services”. Nous supposons que la base de données est composée seulement des 8 enregistrements de l’exemple dans le Tableau 3.1.

La mesure Tf-idf entre S et T est donnée par l'équation :

$$\text{Tf-idf}(S, T) = V(\text{“Xerox ”}, S).V(\text{“Xerox ”}, T) = 0.13$$

avec :

— Tf “Xerox”,S = 1 et Idf “Xerox ” = 8/ 6 = 1.33 d’où V (“Xerox”, S) = 1 ;

— Tf “financial”,T = 1 et Idf “financial” = 8 /4 = 2 d’où V' (“financial”, T) = 0.09 ;

— Tf “servicex”,T = 1 et Idf “servicex” = 8 /1 = 8 d’où V' (“servicex”, T) = 0.27 ;

En suivant le même calcul précédent, la mesure Tf-idf entre *T* et *U* est donnée par :

$$\textit{Tf-idf}(T,U) = V(\text{“}\textit{Xerox}\text{”},T).V(\text{“}\textit{Xerox}\text{”},U) = 0.13 * 0.12 = 0.02$$

Nous remarquons que cette mesure donne une valeur faible à cause des erreurs de saisie

3.4 Méthodes de classification

Dans l'étape de classification, les vecteurs de comparaison (représentant les paires d'enregistrements candidats) générés lors de l'étape de comparaison sont classés en général en matches et non-matches . Plusieurs approches ont été proposées pour l'étape de classification. Ces approches peuvent être classées en deux grandes catégories: les approches basées sur l'apprentissage(supervisée et non supervisée) et les approches manuelles (les approches basées sur les règles).

les méthodes de classification évaluées ont été choisies en fonction de leur prévalence dans le domaine de l'apprentissage automatique(en anglais machine-learning), ainsi que d'être disponibles dans la boîte à outils de couplage d'enregistrements python(record-linkage toolkit).

Voici un aperçu des approches basées sur l'apprentissage et basées sur les les régles).

3.4.1 Méthodes basées sur les régles

Les approches basées sur les règles utilisent des langages déclaratifs pour spécifier un ensemble de règles nommées *règles de matching* qui sont utilisées pour décider si deux enregistrements sont des matches ou non-matches. Ces règles sont liées aux similarités attribuées à chaque valeur d'attribut dans les vecteurs de comparaison des enregistrements comparés (qui sont générés à l'étape de comparaison).

elle peut être une condition qui stipule que la valeur d'une seule mesure de similarité *sim* soit supérieure à un *seuil* bien défini : $C = sim_j\ (r[a_i], s[a_i]) > seuil$. Par exemple, deux livres peuvent être considérés comme identiques si les similarités de leurs attributs titres et auteur dépassent toutes les deux certains seuils.

3.4.2 méthode d'apprentissage supervisée

Depuis le début des années 2000, l'apprentissage automatique a été activement utilisé pour l'ER (Elmagarmid et al., 2007). Un système d'ER basé sur l'apprentissage apprend de manière adaptative de bonnes règles de matching ou une fonction de combinaison

numérique à partir de données d'apprentissage (ensemble de vecteurs de comparaison préalablement étiquetés comme étant matches ou non-matches). ces données sont transmises à un algorithme d'apprentissage automatique qui forme un modèle qui sera utilisé pour prédire les paires futures(**prediction**). Enfin, le modèle de classification appris dans l'étape d'apprentissage est utilisé pour classer les vecteurs de comparaison non étiquetés.

l'inconvénient de l'apprentissage supervisé est qu'il nécessite généralement de grandes quantités de données d'apprentissage et peut être difficile à former avec suffisamment de cas ambigus ... la boîte à outils de couplage d'enregistrements python prend en charge deux algorithmes de classification supervisés différents; naive bayes et classificateur SVM (support vector machine).

3.4.3 Méthodes d'apprentissage non supervisées

Enfin, nous utiliserons l'apprentissage non supervisé pour classer nos paires d'enregistrements candidates. Les modèles non supervisés s'entraînent sans que l'utilisateur n'ait à fournir de données d'apprentissage au modèle la boîte à outils de recorde linkage python prend en charge l'algorithme de classification non supervisés; Algorithme d'attente / maximisation conditionnelle,en englais

(Expectation/Conditional Maximization Algorithm) (ECM).

4 Contexte et travaux connexes

Le problème de résolution d'entité a été défini à l'origine par Newcombe et al. (1959) et a été formalisé par Fellegi and Sunter (1969) dix ans après. Depuis les années 1990, le développement de l'informatique a contribué à la promotion de la recherche sur l'ER (Winkler, 1990)[4]. Depuis lors, ce problème a été considéré au sein de différentes communautés, y compris la communauté d'intelligence artificielle et la communauté de base de données. Plusieurs approches et frameworks ont été proposés pour l'ER. La plupart des approches se focalisent sur l'étape de classification.

4.1 Approches à base de règles

SERF (Stanford Entity Resolution Framework) (Benjelloun et al., 2009)[5] est un framework générique d'ER qui vise à améliorer la mise en échelle. Les mesures de similarité sont considérées comme des boites noires. Différents algorithmes sont proposés pour minimiser le nombre d'appels à ces boîtes noires potentiellement coûteuses en gardant une trace des valeurs comparées précédemment, évitant ainsi les comparaisons redondantes. Plusieurs mesures de similarité peuvent être combinées par une disjonction de règles de matching simples définies manuellement.

MOMA (mapping-based object matching) (Thor & Rahm, 2007)[6]est un framework d'ER qui fournit une bibliothèque extensible de mesures de similarité. Le matching est basé à la fois sur les valeurs d'attributs et le contexte des entités. Le matching basé sur le contexte utilise les relations sémantiques entre différentes entités, par exemple deux auteurs sont susceptibles d'être le même individu s'ils ont des co-auteurs similaires. Différentes fonctions de combinaison (moyenne, min, max, pondérées) peuvent être utilisées pour combiner les valeurs des mesures de similarité individuelles.

DuDe (duplicate detection toolkit) (Draisbach & Naumann, 2010)[7] est boîte à outils pour l'ER qui offre trois benchmarks (pour la déduplication) et plusieurs composants avec des interfaces pouvant être desservies avec un code individuel. L'outil propose deux méthodes d'indexation: Sorted-Neighborhood et une variante de cette méthode qui modifie de manière dynamique la taille de la fenêtre en fonction du fait que la dernière paire renvoyée a été interprétée comme match ou non. Plusieurs mesures de similarité peuvent être combinées par ce que l'on appelle des comparateurs multiples et qui permettent de calculer la valeur minimale ou maximale, la moyenne pondérée et la moyenne harmonique.

FRIL (Finegrained Record Integration and Linkage Tool) (Jurczyk, Lu, Xiong, Cragan, & Correa, 2008)[8] est un outil développé en Java. Il fournit un ensemble de

paramètres ajustables par l'utilisateur, complétés par des outils de visualisation graphiques, pour aider les utilisateurs à comprendre les effets des choix des paramètres. Sorted-Neighborhood est utilisé comme méthode d'indexation. Les mesures de similarité disponibles sont: Éditer Distance, Soundex, Q-grams et Egalité. Une somme pondérée normalisée est fournie pour combiner les valeurs des mesures de similarité en une valeur *v*. Deux seuils *t1* et *t2* (*t1<t2*) doivent être définis par l'utilisateur. Si *v>t2* alors les deux enregistrements sont considérés comme matches, si *v<t1* alors les deux enregistrements sont considérés comme non-matches et si *v* est comprise entre *t1* et *t2* alors les deux enregistrements sont considérés comme matches probables et doivent être vérifiés par un expert.

BN (Bayesian network) (Leitão, Calado & Weis, 2007) [9]est un framework pour l'ER des entités représentées en XML sur la base d'un modèle de réseau bayésien. Les réseaux bayésiens fournissent un formalisme basé sur des graphes pour représenter explicitement les dépendances entre les entités d'un domaine. Ce modèle est dérivé de la structure des entités XML à matcher. L'approche combine numériquement la similarité des valeurs d'attributs des deux entités XML ainsi que celles des entités XML descendantes. L'utilisateur doit spécifier un seuil de probabilité de matching au-dessus duquel les entités sont considérées comme des matches.

Les approches à base de règles sont des approches manuelles qui nécessitent la spécification d'un modèle de classification (ensemble de règles ou fonction de combinaison numérique) par un expert du domaine. Par conséquent, ces systèmes manuels deviennent difficiles à utiliser pour tous les utilisateurs et dans tous les domaines.

4.2 Approches à base d'apprentissage supervisé

Les approches basées sur l'apprentissage supervisé apprennent de bonnes règles de matching à partir de données d'apprentissage (ensemble de vecteurs de comparaison préalablement étiquetés comme étant matches ou non-matches).

MARLIN (Multiply Adaptive Record Linkage with INduction) (Bilenko & Mooney, 2003a)[10] est un framework basé sur l'apprentissage qui utilise SVM à deux niveaux. Au premier niveau, SVM est utilisé pour apprendre pour chaque attribut la mesure de similarité à utiliser. Au deuxième niveau, SVM est utilisé pour déterminer une combinaison de mesures de similarité. MARLIN utilise la méthode d'indexation Canopy Clustering utilisant la mesure de similarité Jaccard. Pour la sélection des données d'apprentissage, deux méthodes semi-automatiques sont utilisées: la *sélection statique active* et la *sélection négative faiblement étiquetée* (Bilenko & Mooney, 2003b)[11]. Dans la sélection statique active, les paires d'enregistrements les plus similaires et les plus dissimilaires en utilisant la mesure de similarité TF-IDF sont sélectionnées et présentées à l'utilisateur pour être étiquetées manuellement et utilisées comme données d'apprentissage. La sélection négative faiblement étiquetée est une méthode non supervisée, elle est utilisée si les exemples positifs sont disponibles et l'objectif est de sélectionner les exemples négatifs. Dans ce cas, un ensemble de paires d'enregistrements parmi ceux qui ne partageant pas plus de 20% de tokens sont sélectionnées d'une façon aléatoire et considérées comme des exemples négatifs.

Le système de classificateur multiple (Zhao & Ram, 2005) [12]utilise une variété de classificateur pour apprendre à combiner les mesures de similarité, notamment les arbres de décision, Naïve Bayes, la régression linéaire et logistique, les réseaux de neurones et les k-voisins les plus proches. En outre, les approches de méta- combinaison permettant de combiner plusieurs apprenants supervisés sont prises en charge, à savoir le bagging, le boosting et le stacking. Tandis que le bagging et le boosting combinent plusieurs classificateurs supervisés du même type, le stacking est utilisé pour combiner des classificateurs supervisés de types différents (par exemple, les arbres de décision et la régression logistique). Le support d'indexation n'est pas explicitement mentionné, mais l'évaluation suggère qu'une méthode d'indexation a été utilisée.

Chaudhuri, Chen, Ganti and Kaushik (2007) [13]spécifient l'ER par une arborescence d'opérateurs qui correspond à l'union (disjonction) de *similarity joins*. Des échantillons d'apprentissage étiquetés manuellement sont utilisés pour construire les arbres d'opérateurs selon une stratégie récursive de division et de conquête. Le nombre maximal de *similarity joins* dans une arborescence d'opérateurs et le nombre maximal de fonctions de similarité par *similarity joins* peuvent être limités par l'utilisateur. L'indexation n'est pas explicitement prise en charge. Cependant, les auteurs indiquent que la méthode d'indexation Canopy Clustering utilisant la mesure de similarité Jaccard est utilisée dans l'évaluation.

Dans (Reyes-Galaviz, Pedrycz, He & Pizzi, 2017)[14], les auteurs ont proposé un algorithme d'apprentissage supervisé basé sur les gradients issu d'une certaine topologie d'un réseau de neurones artificiels. Pour l'agrégation des scores des mesures de similarité, une somme pondérée normalisée est utilisée. L'algorithme proposé permet d'apprendre les poids des attributs et deux seuils. Les paires dont le score est au-dessus d'un seuil supérieur sont considérées comme des matches et les paires dont le score est au-dessous d'un seuil inférieur sont considérées comme des non-matches. Les paires avec un score compris entre les deux seuils sont considérées comme des matches potentiels et sont examinés par un expert. Pour l'indexation, le blocage standard (traditionnel) est utilisé.

Les approches basées sur l'apprentissage supervisé se traduisent généralement par une qualité de matching plus meilleure car elles sont adaptatives (Christen, 2012a)[1], mais nécessitent un grand nombre d'exemples d'apprentissage (vecteurs de comparaisons étiquetés comme matches ou non-matches) qui n'est pas disponible dans de nombreuses situations réelles.

4.3 Approches à base d'apprentissage semi supervisé

Ces dernières années, des approches semi-supervisées ont été proposées pour alléger le problème des approches supervisées en utilisant un ensemble de données d'apprentissage réduit. Parmi les approches semi-supervisées, les approches basées

sur l'apprentissage actif procèdent d'une façon itérative et commencent par un ensemble réduit d'exemples d'apprentissage. À chaque itération, les vecteurs de comparaison considérés comme représentatifs sont identifiés, étiquetés manuellement par l'utilisateur et ajoutés à l'ensemble des données d'apprentissage pour retrainer le classificateur.

Active Atlas (Tejada, Knoblock & Minton, 2002) [15]est un système qui permet d'apprendre un ensemble de règles de matching en utilisant l'apprentissage actif. Plusieurs apprenants (arbre de décision) sont utilisés. Les vecteurs de comparaison non étiquetés qui présentent plus de désagrément entre un comité de classificateurs sont considérés comme représentatifs et présentés à l'utilisateur pour étiquetage. Pour la comparaison des valeurs d'attributs, une variante de TF-IDF est utilisée. Pour l'indexation, une stratégie de blocage disjointe basée sur le hachage est prise en charge.

ALIAS (Sarawagi & Bhamidipaty, 2002)[16] est un système similaire à Active Atlas qui prend en charge trois apprenants comme classificateur de base (arbre de décision, SVM et naive Bayes). Pour la comparaison des valeurs d'attributs, plusieurs mesures de similarité sont supportées comme 3-grams et Levenshtein. Pour l'indexation, Sorted Neighborhood et le blocage standard sont supportés.

T3S (Dal Bianco, Galante, Goncalves, Canuto & Heuser, 2015)[17] est une approche qui vise à minimiser le nombre des vecteurs de comparaison à étiqueter par l'utilisateur. En premier lieu, les paires d'enregistrements candidates sont partitionnées en plusieurs groupes selon leurs scores de similarité. Ensuite, un ensemble de paires est sélectionné d'une façon aléatoire à partir de chaque groupe. Dans la deuxième étape, les paires représentatives sont sélectionnées d'une façon itérative en utilisant la méthode SSAR (Selective Sampling using Association Rules) (Silva, Gonçalves & Veloso, 2014)[18]. La mesure de similarité utilisée est Jaccard des n-grams. L'indexation est basée sur le filtrage PPJoin+ (Xiao, Wang, Lin, Yu & Wang, 2011)[19] (avec la mesure de similarité jaccard) et le seuil est déterminé automatiquement.

Dans (Wang, Vatsalan & Christen, 2015)[20], les auteurs ont présenté une

approche qui permet de prendre en compte le nombre alloué des vecteurs de comparaison à étiqueter par l'utilisateur. Dans chaque itération, l'algorithme de clusturing farthest-first est utilisé pour sélectionner les vecteurs de comparaison les plus distants entre eux. Ces vecteurs sont étiquetés par l'utilisateur, enlevés de l'ensemble des vecteurs non étiquetés et ajoutés aux données d'apprentissage. Ensuite, les vecteurs non étiquetés sont fragmentés en deux groupes en utilisant le classificateur et sont traités dans la prochaine itération. L'algorithme se termine lorsque le nombre alloué des vecteurs de comparaison à étiqueter par l'utilisateur est atteint ou tous les vecteurs non étiquetés sont traités.

SILK (Volz, Bizer, Gaedke & Kobilarov, 2009) [21]est un framework hybride pour l'ER des entités représentées en RDF. Il fournit un langage déclaratif permettant à l'utilisateur de spécifier les mapping entre propriétés, les mesures de similarité (jaro, jaroWinkler, q-grams, egalité, etc.), la méthode d'agrégation des scores des similarités (moyenne pondérée, max, min, produit pondéré), les seuils et la méthode d'indexation à utiliser. Pour l'indexation, SILK permet la spécification (manuelle) de plusieurs clés de blocage, c'est-à-dire que seules les instances partageant une des clés de blocage doivent être comparées les unes aux autres. SILK prend en charge aussi l'apprentissage supervisé utilisant les algorithmes génétiques grâce à son algorithme GenLink (Isele & Bizer, 2012)[22] et l'apprentissage actif utilisant les algorithmes génétiques grâce à son algorithme Active-GenLink (Isele & Bizer, 2013)[23].

Une deuxième catégorie d'approches semi-supervisées utilisée dans (Kejriwal & Miranker, 2015)[24] repose sur l'auto apprentissage. L'idée est de commencer avec un ensemble réduit de données d'apprentissage. Ensuite, à chaque itération, les vecteurs de comparaison pour lesquels le classificateur est le plus sûr sont ajoutés à l'ensemble de données d'apprentissage pour retrainer le classificateur. Ce processus itère jusqu'à ce que tous les vecteurs de comparaison non étiquetés aient été classés par le classificateur final. 28 mesures de similarité sont utilisées : 2 numériques, 8

basées sur les caractères et les tokens, et 18 phonétiques. Pour l'indexation, *Attribute Clustering* (Papadakis et al., 2013)[25] est utilisé. Pour la classification, les auteurs utilisent l'apprentissage d'ensemble (Dietterich, 2000)[26] et en particulier le boosting qui consiste à trainer plusieurs classificateurs de base afin de construire un classificateur général. Le classificateur d'ensemble utilise le vote à la majorité pondérée pour prendre une décision.

Les approches semi supervisées nécessitent un ensemble réduit de données d'apprentissage, elles sont itératives et peuvent aussi nécessiter l'intervention continue d'un humain pour l'étiquetage.

4.4 Approches non supervisées

Les approches semi-supervisé nécessite l'existence d'un ensemble réduit de données d'apprentissage et / ou l'intervention d'un expert pour l'étiquetage. Pour faire face à ce problème, plusieurs approches non supervisées (automatiques) ont été proposées.

TAILOR (Elfeky et al., 2002) [27]est un framework hybride qui prend en charge des approches de combinaison numériques et des approches basées sur l'apprentissage utilisant les arbres de décision. Cinq mesures de similarité peuvent être utilisées (distance de Hamming, Levenshtein, Jaro, q-grams et soundex). TAILOR supporte le blocage standard et Sorted Neighborhood comme méthodes d'indexation. Pour l'automatisation, deux approches sont proposées. La première approche non supervisée utilise l'algorithme de clusturing k-means. La deuxième approche est hybride, elle utilise l'algorithme k-means pour classer un sous ensemble de vecteurs de comparaison en matches et non matches. Les vecteurs classés par k-means sont ensuite utilisés comme données d'apprentissage par les arbres de décision. Les auteurs ont constaté dans l'expérimentation que l'approche hybride est plus performante que l'approche non supervisée utilisant k-means.

FEBRL (Freely Extensible Biomedical Record Linkage) (Christen, 2008)[28] est un framework hybride qui prend en charge des approches basées sur l'apprentissage utilisant SVM ainsi que des approches basées sur les règles (fonction

de combinaison numérique). 26 mesures de similarité différentes sont disponibles pour la comparaison des valeurs d'attributs et 6 méthodes d'indexation sont supportées. Pour l'automatisation, l'auteur propose deux méthodes pour la génération automatique des données d'apprentissage (Christen, 2007) [29]: *threshold based* et *nearest based.* Dans la méthode threshold based, les vecteurs de comparions qui ont

dans tous leurs éléments une valeur inferieure (supérieure) à un seuil *s1* (*s2*) sont considérés comme non-matches (matches). Dans la méthode nearest based, les vecteurs de comparaisons les plus proches du vecteur 1(1,1,…,1) (du vecteur 0(0,0,…,0)) en utilisant par exemple la distance euclidienne sont considérés comme matches (non- matches). Pour cette approche, la sélection des vecteurs de comparaison est basée sur le score d'une seule mesure de similarité (Jaro-Winkler), qui est une mesure de similarité basée sur les caractères et qui ne prend pas en charge les erreurs de réarrangement de mots, alors que notre générateur de données d'apprentissage utilise plusieurs mesures de similarité pour chaque attribut pour prendre en charge plusieurs types d'erreurs (typographiques et réarrangements de tokens). Le travail de (Christen, 2007) [29]sera comparé avec notre générateur de données d'apprentissage dans l'expérimentation.

Une deuxième catégorie d'approches non supervisées repose sur l'optimisation de la valeur d'une fonction objective appelée pseudo F-mesure (PFM) sur des données non étiquetées. La PFM est une heuristique visant à donner une approximation de la F-mesure réelle, elle est formulée en supposant que deux enregistrements de sources de données différentes représentent la même entité, tandis que deux enregistrements de la même source de données représentent des entités distinctes. L'objectif de ces approches est de trouver des règles de matching qui maximisent la valeur de la PFM.

KnoFuss (Nikolov, Uren & Motta, 2007)[30] est un framework hybride pour l'ER des entités représentées en RDF qui prend en charge la spécification manuelle

des règles de matching (combinaison numérique) ainsi qu'un algorithme non déterministe (KnoFuss+GA) qui utilise la PFM comme fonction de fitness dans les algorithmes génétiques (Nikolov, d'Aquin & Motta, 2012)[31].

LIMES (Ngomo & Auer, 2011) [32]est un framework pour le matching des entités représentées en RDF qui est similaire à SILK. Il fournit un langage déclaratif permettant à l'utilisateur de spécifier les mapping entre propriétés, les mesures de similarité, la méthode d'agrégation des scores des similarités et les seuils. LIMES applique un filtrage en exploitant l'inégalité triangulaire pour optimiser l'exécution. LIMES prend en charge aussi l'apprentissage supervisé, l'apprentissage actif utilisant les algorithmes génétiques grâce à son algorithme EAGLE (Ngomo & Lyko, 2012)[33], l'apprentissage actif utilisant son algorithme déterministe RAVEN (Ngomo, Lehmann, Auer & Höffner, 2011)[32] qui effectue une recherche hiérarchique pour trouver un classificateur linéaire et un classificateur booléen, et enfin l'apprentissage non supervisé en utilisant un algorithme déterministe (EUCLID) et un autre non déterministe (EAGLE-UN) (Ngomo & Lyko, 2013)[34].

EUCLID (Ngomo & Lyko, 2013) [34]est un algorithme déterministe qui effectue une recherche hiérarchique pour trouver un classificateur linéaire et un classificateur booléen en optimisant la PFM. Les auteurs ont présenté aussi un algorithme non déterministe qui utilise la PFM comme fonction de fitness dans les algorithmes génétique. Dans une étude récente, les auteurs dans (Ngomo & Lyko, 2013) [34]ont constaté que, sur des données réelles, la PFM n'est pas bien corrélée avec la vraie F- mesure. Avec l'approche génétique, les sources de données doivent être analysées en plusieurs itérations et les résultats ne sont pas déterministes. Contrairement aux approches non supervisées basées sur la PFM.

4.5 Approches pour langue arabe

La plupart des frameworks ont été développés pour effectuer la résolution d'entité dans les sources de données en latin et en particulier l'anglais comme Febrl

(Christen, 2008) [28]et TAILOR (Elfeky et al., 2002)[27]. Ces frameworks ne reconnaissent même pas les caractères non latins et en particulier l'arabe parce qu'ils n'utilisent pas le système Unicode (Higazy et al., 2013)[35]. Pour cette raison, plusieurs travaux ont été développés pour prendre en charge la langue arabe.

Gueddah et al. (2012)[36] ont proposé une variante de la distance de Levenshtein. Pour calculer la similarité, le cout des opérations d'édition dépend de trois matrices de fréquences (M_insertion, M_suppression, M_substitution) qui ont été calculées à partir d'un test effectué par un ensemble d'opérateurs professionnels. Cette approche nécessite l'existence d'un corpus afin d'évaluer et d'estimer les matrices de fréquence des erreurs d'édition.

Une autre approche pour améliorer l'algorithme de Levenshtein est présentée dans (Ghafour et al., 2011)[37]. Les auteurs utilisent trois matrices de similarité pour calculer le cout d'opérations d'éditions (ajout, suppression et modification) : la matrice de similarité de forme de caractère, la matrice de similarité phonétique et la matrice de proximité des lettres dans le clavier (en fonction de la distance euclidienne).

El-Shishtawy (2013)[38] a proposé une extension de la distance de Levenshtein pour les noms arabes composés de plusieurs tokens. L'idée est de traiter chaque token comme un caractère. Le cout des opérations d'édition (insertion, suppression et modification) est calculé par une fonction qui calcule la position du token dans le nom et sa fréquence dans un corpus de noms.

D'autres travaux se sont concentrés sur les mesures de similarités phonétiques pour la langue arabe. Aqeel, Beitzel, Jensen, Grossman and Frieder (2006)[39] ont proposé la version arabe de l'algorithme Soundex. Yousef (2014) [40]a proposé une mesure phonétique pour les noms arabes composés de plusieurs tokens. L'idée est de calculer le code soundex de chaque token et ensuite concaténer les codes résultants pour avoir le code soundex du nom composé.

Les approches citées précédemment se sont concentrées sur la résolution

d’entité dans les sources de données composées d’un seul attribut (en général les noms en arabe). En plus, ces approches nécessitent des données d’apprentissage élaborées manuellement par un expert et/ou la définition d’un seuil pour décider si deux enregistrements (noms) matchent ou non.

Higazy et al. (2013)[35] ont proposé un framework d’ER qui prend en charge les sources de données en anglais ainsi que celles en arabe. Le taux du rappel été amélioré lorsqu'une extension pour la langue arabe a été ajoutée à l’étape de prétraitement. Le framework présenté dans (Higazy et al., 2013)[35] nécessite l’intervention d’un expert pour l’élaboration des règles de matching ou l’existence des données d’apprentissage.

5 Mesures de performance

Lorsque les paires d'enregistrements ont été classées comme concordantes et non concordantes, il est important d'évaluer la performance de les mesures de similarité et l'algorithme de classification. l'explication suivante suppose que l'algorithme de classification prend un ensemble d'enregistrements, R, et produit une sortie de correspondances, M et de non-correspondances, N. Dans cette hypothèse, les classifications sont divisées en quatre catégories, qui sont présentées ci-dessous dans le tableau 3.2

Confusion matrix.

Actual	Classification	
	Match (M)	Non-Match (N)
Match (M)	(True match) True Positive (TP)	(False non-match) False Negative (FN)
Non-match (N)	(False match) False Positive (FP)	(True non-match) True Negative (TN)

tableau 3.2 :matrice de confusion

Les vraies correspondances (true matche)classées comme correspondance sont appelées vrais positifs (true positive)et les vraies correspondances classées comme non-correspondances sont appelées faux négatifs(false negative). Les non-correspondances (non-matche)réelles classées comme correspondance sont appelées faux positifs(false positive) et les non-correspondances classées comme non-correspondance sont appelées vrais négatifs(true negative).

le résultat donné par l'algorithme de classification, M et N, peut donc être défini comme M = T P + FP et N = FN + T N. Sur la base de la valeur seuil, t, le nombre de paires classées comme concordantes ou non concordantes différera. Si la valeur de coupure est diminuée, elle peut provoquer plus de faux positifs et si elle est augmentée, elle peut provoquer plus de faux négatifs.

cela est dû au fait qu'un seuil bas entraînerait des paires avec une faible similitude comme marquées et inversement pour les seuils élevés.

5.1 Indicateurs de performance

cette section contient différentes mesures utilisées pour évaluer les paires classées. les mesures de qualité sont basées sur les classifications possibles données dans le tableau 1. la liste suivante contient un aperçu de certaines mesures de qualité couramment utilisées la précision est la proportion de paires d'enregistrements classées comme concordance qui sont des correspondances réelles. Le rappel est la proportion de toutes les correspondances réelles identifiées comme correspondant. Enfin, la mesure F peut être considérée comme une métrique qui équilibre la précision et le rappel.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - mesure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Chapitre 4

Implémentation et évaluation

4.1 Introduction

Ce chapitre a pour objectif d'évaluer et de tester les performances du méthode proposé les expériences ont été menées en générant d'abord un espace de comparaison à évaluer. Chaque champ individuel a ensuite été comparé et marqué comme une correspondance (1) si leur similitude était supérieure au seuil donné, sinon ce champ a été marqué comme une non-correspondance (0). Chaque mesure de similarité a été évaluée. Pour chaque mesure de similarité, les trois méthodes de classification ont été utilisées pour classer les paires comme correspondance ou non correspondance. les tests et mesures effectués ont été exécutés sur un ordinateur avec les spécifications présentées dans le tableau 4.1 System spécification

Name	Type
CPU	**Intel(R) Core(TM)i5-4210U @1.70GHz 2.40GHz**
RAM	**6.00 Go**
Hard drive	**SSD 1 TB**
OS	**Windows 8.1 Professionnel**
Programming language	**Python 3.8.0**
Library	**Python Record Linkage Toolkit v0.12**

tableau 4.1 : les spécifications ordinateur qui fait l'exécution

4.2 méthode proposée pour faire l'évaluation :

On a mené plusieurs recherche et des testes afin d'étudier notre approche propose et qui consiste a utiliser le blocage standard pour minimiser le nombre des paire enregistrements et utilise 5 mesures de similarité (jaro , jarowinkler , levnshtein ,cosinus et dice) on a fait not recherche pour chacune de ses mesures et elle a 3 classificateur 2 qui sont superviser (naives bayes et SVM) et 1 non superviser (ecm) puis évaluer par les indications de performance(rappel et percision et f-score) sur une base de donne choisit qui contient 10000 enregistrement et tout cela est présenter :

sous forme algorithme

Début

Entrée data A, data B

affiche("étape de l'indexation");

indexer = recordelinkage.index();

indexer. Block("le nom de l'attribut");

candidats_lien = indexer.index(data A,data B);

afficher("étape de comparaison");

comparer. String("le nom de l'attribut", méthode = "le nom de la mesure de similarité");

resultat = comparer.compute(candidats_lien ,data A ,data B);

afficher("étape de classification");

var char A[15];

si A='svm' alors

svm = recordelinkage .svmclassifier(resultat , vrai_correspondant)

resultats_c = svm(resultat)

sinon

si A='naive_bayes' alors

```
    naive_bayes=recordelinkage.naivebayesclassifier(resultat, vrai_correspondant);
    resultat_c=naive_bayes (resultat);
                sinon
    resultat_c=recordelinkage.ecmclassifier(resultat);
                 Fin si
              Fin si

afficher(''étape de l'évaluation');
     confusion=recordelinkage.confusion_matrix(vrai_correspondant, resultat_c,
                          len(candidate_links));
          afficher('confusionmatrix');
          fscore=recordelinkage.fscore(confusion)
          afficher('fscore',fscore)
          recall=recordelinkage.recall(vrai_correspondant,resultat_c)
          afficher('recall',recall)
          precision=recordelinkage.precision(vrai_correspondant,resultat_c)
          afficher('precision', precision).
```

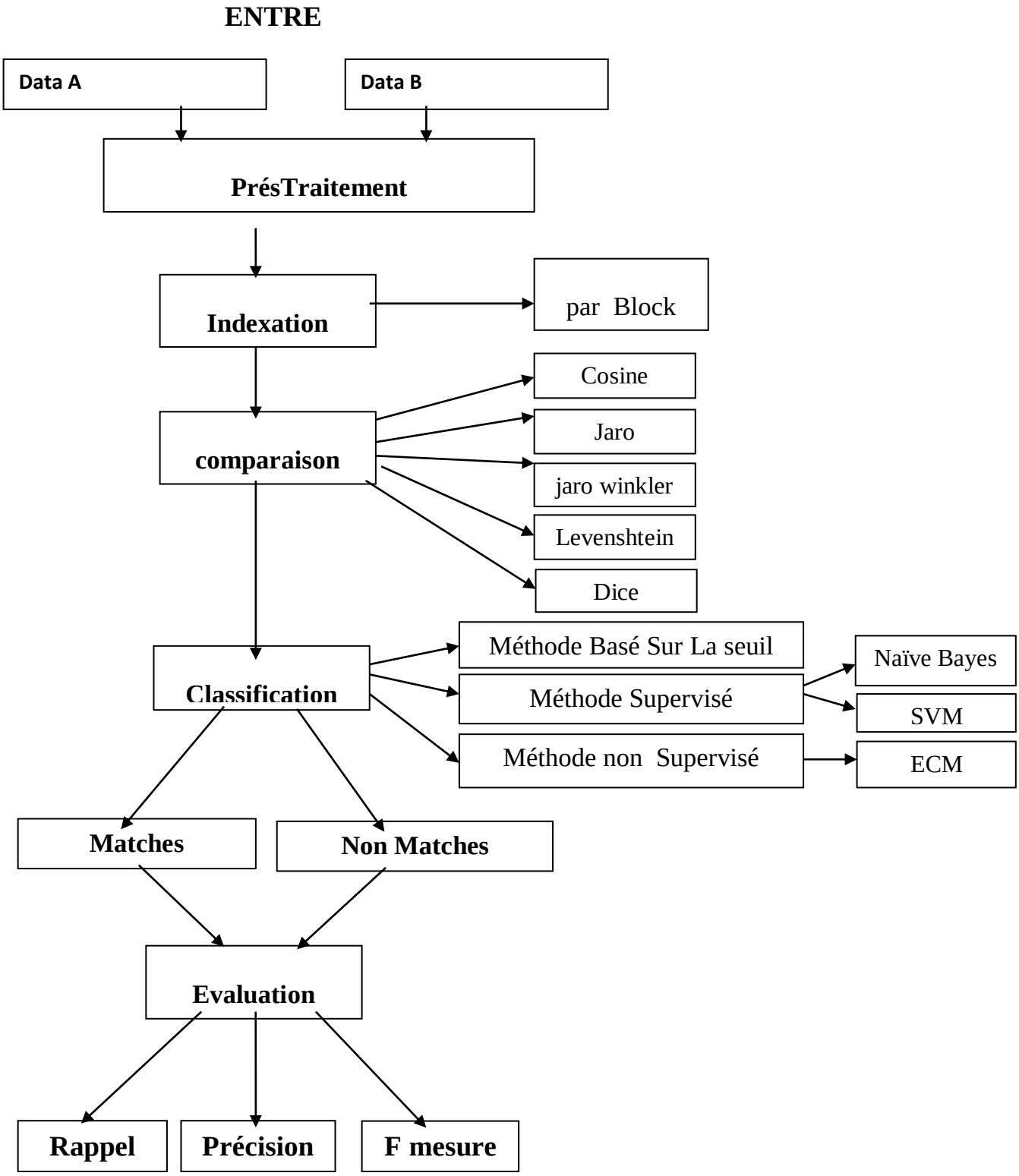

figure 4 Schéma du système de résolution d'entité présenté .

4.3 Données de test

Pour l'expérimentation, nous avons utilisé l'ensemble de données provient du système FEBRL accessible au public. Il s'agit d'un registre de données personnelles générées aléatoirement avec des champs communs tels que les noms, adresses et dates de naissance. Tous les enregistrements sont étiquetés en double ou originaux.

l'ensemble de données contient 10000 lignes avec 5000 enregistrements originaux et 5000 doublons, exactement une pour chaque enregistrement original.

Un exemple de l'ensemble de données FEBRL est présenté dans le tableau 4.2.

id	Give-name	surname	Stree-number	Adress-1	Adress-2	suburb	postcode	state
495	brodee	kirkmoe	71	Brisbane avenue	sherwood	toowoomba	4127	nsw
495-0	brodee	pyper	72	Brisban e avenue	sherwood	toowomba	4127	nsw

Tableau 4.2 : Exemple d'ensemble de données FEBRL.

4.4 Expérimentations et résultats

4.4.1 évaluation par les Indicateurs de performance

Pour l'ensemble de données FEBRL utilisée indexation par block pour 3champ :'give name' 'surname' 'date of birth' et un espace de comparaison de 160 789 paires candidates. les champs choisis pour la comparaison étaient le 'give name','surname', suburb',' ,'state' ,'l'adresse 1' et ' l'adresse 2' .

pour la seuil>3 signifie la somme des seuils entre chaque paire d' attributs dans chaque paire enregistrement qui est >3.

les meilleurs indicateur de performance trouvés pour chaque combinaison de classificateur et de mesure de similarité sont donnés ci-dessous dans les tableau 4.3 au 4.8 es tableau contient les scores F1 ,rappel et précision pour les ensembles de données FEBRL.

tableau 4.3 : les indicateur de performance pour la mesure de similarité jaro.

Mesure de similarité	classificateur	Indicateur de performance		
		Rappel	précision	f-score
Jaro	Naive bayes	0.908	0.988	0.947
	Svm	0.906	0.990	0.946
	Ecm	0.954	0.996	0.974
	Seuil>3	0.98	0.068	0.128

tableau 4.4 : les indicateur de performance pour la mesure de similarité jaro-winkler.

Mesure de similarité	classificateur	Indicateur de performance		
		Rappel	précision	f-score
Jaro-Winkler	Naive bayes	0.909	0.991	0.948
	Svm	0.905	0.989	0.945
	Ecm	0.954	0.999	0.977
	Seuil>3	0.98	0.07	0.13

tableau 4.5: les indicateur de performance pour la mesure de similarité levenshtein .

Mesure de similarité	classificateur	Indicateur de performance		
		rappel	précision	f-score
levenshtein	Naive bayes	0.969	0.986	0.977
	Svm	0.955	0.989	0.972
	Ecm	0.970	0.983	0.976
	Seuil>3	0.969	0.610	0.748

tableau 4.6 : les indicateur de performance pour la mesure de similarité cosinus

Mesure de similarité	classificateur	Indicateur de performance		
		Rappel	précision	f-score
Cosinus	Naive bayes	0.971	0.972	0.974
	Svm	0.958	0.988	0.973
	Ecm	0.971	0.984	0.977
	Seuil>3	0.957	0.904	0.929

tableau 4.7 : les indicateur de performance pour la mesure de similarité dice .

Mesure de similarité	classificateur	Indicateur de performance		
		Rappel	précision	f-score
Dice	Naive bayes	0.967	0.986	0.976
	Svm	0.754	0.778	0.766
	Ecm	0.969	0.981	0.975
	Seuil>3	0.953	0.774	0.854

tableau 4.8 : les indicateur de performance pour l'utilisation de tous les mesure de similarité pour chaque attribut.

Mesure de similarité	classificateur	Indicateur de performance		
		Rappel	percision	f-measeur
Tous les mesures	Naive bayes	0.979	0.989	0.983
	Svm	0.964	0.989	0.976
	ecm	0.978	0.983	0.981

Discussion des résultats

Comme le montre les tableau, les indicateur de performance diffère considérablement selon la combinaison de classificateur et de mesure de similarité utilisée. Pour cette raison, la connaissance la plus importante à retenir est l'importance d'évaluer quel classificateur, mesure de similarité et seuil fonctionne le mieux pour votre ensemble de données. cela est dû au fait qu'une combinaison de ces éléments pourrait bien fonctionner pour un ensemble de données, mais n'est pas garantie de fonctionner aussi bien pour un autre ensemble de données. Dans l'ensemble, pour les ensembles de données, Naive bayes avait les meilleures performances des classificateurs supervisés et même la classificateur non supervisée EcM avait les meilleures performances .

Le tableau 4.8 montre également que l'utilisation de tous les mesure de similarité pour chaque attribut avait les meilleurs score f-score ,et que l'algorithme levenshtein avait les meilleurs scores F-score des algorithmes basés sur les caractères, et que l'algorithme de cosinus avait les meilleurs f-score des algorithmes basés sur les jetons.

on peut de nouveau constater que la mesure de similitude Jaro-Winkler et jaro est une valeur aberrante en ce qu'elle nécessite des seuils de similitude plus élevés pour produire un F-score élevé par rapport aux autres mesures de similitude.

4.4.2 évaluation par la matrices de confusion

Cosine

réel

	vraie	faux
vraie	**4793**	**207**
faux	**57**	155732

prédiction

levenshtein

réel

	vraie	faux
vraie	**4777**	**223**
faux	**52**	155737

prédiction

jaro

réel

	vraie	faux
vraie	**4543**	**466**
faux	**42**	155747

prédiction

jarowinkler

réel

	vraie	faux
vraie	**4525**	**475**
faux	**49**	155740

prédiction

Dice

réel

	vraie	faux
vraie	**3771**	**1229**
faux	**1072**	154717

prédiction

tous les mesures

réel

	vraie	faux
vraie	**4894**	**106**
faux	**81**	155708

prédiction

figure4. 1 :Matrice de confusion pour SVM.

Cosine

réel

	vraie	faux
vraie	**4857**	**143**
faux	**127**	

prédiction

levenshtein

réel

	vraie	faux
vraie	**4849**	**154**
faux	**67**	155722

prédiction

jaro

réel

	vraie	faux
vraie	4544	456
faux	52	155737

prédiction

jarowinkler

réel

	vraie	faux
vraie	**4544**	**456**
faux	**40**	155749

prédiction

Dice

réel

	vraie	faux
vraie	**4835**	**165**
faux	**67**	155722

prédiction

tous les mesures

réel

	vraie	faux
vraie	**4888**	**112**
faux	**136**	155653

prédiction

figure4. 2 :Matrice de confusion pour Naive bayes

Cosine

réel \ prédiction	vraie	faux
vraie	***4856***	***144***
faux	***75***	*155714*

levenshtein

réel \ prédiction	vraie	faux
vraie	**4850**	**150**
faux	**97**	155710

jaro

réel \ prédiction	vraie	faux
vraie	**4772**	**228**
faux	**15**	155774

jarowinkler

réel \ prédiction	vraie	faux
vraie	**4772**	**228**
faux	**3**	155786

Dice

réel \ prédiction	vraie	faux
vraie	**4846**	**154**
faux	**92**	155697

tous les mesures

réel \ prédiction	vraie	faux
vraie	**4823**	**177**
faux	**52**	155737

figure 4.3 :Matrice de confusion pour ECM

Cosine

réel \ prédiction	vraie	faux
vraie	**4758**	**215**
faux	**506**	155283

levenshtein

réel \ prédiction	vraie	faux
vraie	**4847**	**153**
faux	**3096**	152693

jaro

réel \ prédiction	vraie	faux
vraie	**4901**	**99**
faux	**66843**	88946

jarowinkler

réel \ prédiction	vraie	faux
vraie	**4901**	**99**
faux	**65000**	90789

Dice

réel \ prédiction	vraie	faux
vraie	**4767**	**233**
faux	**1389**	154400

figure 4.4:Matrice de confusion pour seuil>3

Discussion des résultats

Dans l'ensemble, les taux de prédiction sont très bons. il y a cependant quelques combinaisons qui méritent d'être mentionnées. Une combinaison intéressante peut être observée en étudiant la matrice de confusion des seuil>3 de la figure 4.4, où l'on peut voir que la mesure de similarité Jaro-Winkler et jaro produit de nombreuses fausses correspondances. . Il est également intéressant de voir que Jaro-Winkler fonctionne très bien avec les classificateurs unsupervisés, produit de minimum nombre fausses correspondances . . Dans l'ensemble, les chiffres montrent que les mesures de similitude basées sur des jetons et sur des caractères peuvent avoir des performances similaires.

4.5Estimation des paramètres :

cette section présente les différents paramètres utilisés pour chaque classificateur. les modèles supervisés ont utilisé 70% de données de formation et 30% de données de test. Les méthodes non supervisées ont utilisé l'ensemble des données comme données de test car aucune formation préalable n'est requise. Les paramètres utilisés pour chaque algorithme de classification peuvent être consultés ci-dessous.

4.5.1 Naive Bayes

Le classificateur Naive Bayes partitionne les paires d'enregistrements candidats en correspondances et non-correspondances. Le classificateur est basé sur des principes probabilistes. La méthode de classification Naive Bayes a un lien mathématique étroit avec le modèle de Fellegi et Sunter. Le noyau du classificateur. Le classificateur NaiveBayesClassifier basé sur **sklearn.naive_bayes.BernoulliNB**

les paramètres utilisés sont présentés dans le tableau 4.5.1

Paramètre	Valeur
Alpha	1e-4
Binarize	None
Use-col-names	True

tableau 4.5.1 : les paramètres utilisés pour naive bayes

binarize (float ou None, facultatif (par défaut = None)): Seuil de binarisation (mappage aux booléens) des exemples de fonctionnalités. Si None, l'entrée est présumée être déjà constituée de vecteurs binaires.

alpha (float) -: Paramètre de lissage additif (Laplace) (0 pour aucun lissage). Par défaut 1e-4.

use-col-names (bool) - : Utilisez les noms de colonnes de pandas.DataFrame pour identifier les paramètres. Si la valeur est False, l'index de colonne de la fonction est utilisé. True par défaut.

4.5.2 Svm :

Le classificateur Support Vector Machine partitionne les paires d'enregistrements candidates en correspondances et non-correspondances. Cette implémentation est un classificateur linéaire binaire non probabiliste. Les machines à vecteurs de support sont des modèles d'apprentissage supervisé. Par conséquent, les classificateurs SVM ont besoin de données d'apprentissage .

Le classificateur SVMClassifier utilise l'algorithme de classification sklearn.svm.LinearSVC de SciKit-learn .

les paramètres utilisés sont présentés dans le tableau 4.5.2

Paramètre	Valeur
match_index	5000
Max-iter	1000
Tol	1e-4

tableau 4.5.2 : paramètres utilisés pour SVM

match_index (pandas.MultiIndex) - Un objet pandas.MultiIndex avec les vraies correspondances. Le MultiIndex contient uniquement les vraies correspondances. None par défaut

max_iter (int) - Le nombre maximum d'itérations de l'algorithme SVM . Par défaut 100.

atol (float) - La tolérance entre les paramètres entre chaque interaction. Si la différence entre les paramètres entre les itérations est inférieure à cette valeur, l'algorithme est considéré comme convergé. 10e-4 par défaut.

4.5.3 Expectation/Conditional Maxisation classifier(ECM):

ECM utilisé pour classer les paires d'enregistrements. Cet algorithme probabiliste de couplage d'enregistrements est utilisé en combinaison avec les modèles Fellegi et Sunter. Ce classificateur n'a pas besoin de données d'apprentissage (training data)(non supervisées).

le classificateur ECM Basé sur l'implémentation dans python record-linkage toolkit.

les paramètres utilisés sont présentés dans le tableau 4.5.3

Paramètre	Valeur
Atol	10e-4
Binarize	0.8
Init	Jaro
Max-inter	100
Use-col-names	True

tableau 4.5.3 :les paramètres utilisés pour ECM

init (str) - Méthode d'initialisation de l'algorithme. ‘Jaro’ par défaut.

max_inter (int) - Le nombre maximum d'itérations de l'algorithme EM. Par défaut 100.

binarize (float ou None, facultatif (par défaut = None)) - Seuil de binarisation (mise en correspondance avec les booléens) des exemples de fonctionnalités. Si None, l'entrée est présumée être déjà constituée de vecteurs binaires.

atol (float) - La tolérance entre les paramètres entre chaque interaction. Si la différence entre les paramètres entre les itérations est inférieure à cette valeur, l'algorithme est considéré comme convergé. 10e-4 par défaut.

use-col-names (bool) - : Utilisez les noms de colonnes de pandas.DataFrame pour identifier les paramètres. Si la valeur est False, l'index de colonne de la fonction est utilisé. True par défaut.

4.6 Conclusions :

On peut conclure d'après nos recherche approfondit et not résultats des testes que :

Les indicateur de performance diffère considérablement selon la combinaison de classificateur et de mesure de similarité utilisée Comme le montre les tableau

Naive bayes avait les meilleures performances des classificateurs supervisés et même la classificateur non supervisée EcM avait les meilleures performances .

Le tableau 4.8 montre également que l'utilisation de tous les mesure de similarité pour chaque attribut avait les meilleurs score f-score ,et que l'algorithme levenshtein avait les meilleurs scores F-score des algorithmes basés sur les caractères, et que l'algorithme de cosinus avait les meilleurs f-score des algorithmes basés sur les jetons.

Et on peut apprendre par rapport aux autres mesures de similarité que la mesure de similitude Jaro-Winkler et jaro est une valeur aberrante en ce qu'elle nécessite des seuils de similitude plus élevés pour produire un F-score élevé

Il est également intéressant de voir que Jaro-Winkler fonctionne très bien avec les classificateurs unsupervisés, produit de minimum nombre fausses correspondances

4.7 Conclusions générale

Certaines conclusions finales sont importantes à évaluer. la première partie fournit une compréhension large mais non détaillée du domaine du couplage d'enregistrements et de la façon dont il a évolué, et ainsi atteindre l'objectif 1. On peut conclure que la partie théorique fournit des informations sur toutes les étapes de l'approche générale de record -linkage et, à un niveau plus détaillé, explique et évalue les méthodes courantes utilisées dans le couplage d'enregistrements. , cela suffit pour atteindre l'objectif 2. Après cela, l'objectif 3 est atteint dans les chapitres ultérieurs en présentant et en évaluant les performances des méthodes de couplage d'enregistrements mentionnées précédemment sur un ensemble de données. j'espèrent qu'une personne qui ne connaissait pas auparavant le domaine du couplage d'enregistrements, mais qui possède des connaissances antérieures en informatique, peut lire cette recherche et, par la suite, avoir une meilleure compréhension du

domaine, de certaines de ses méthodes courantes et de la façon d'utiliser pour leurs propres besoins.

Références bibliographiques

[1] Christen, P. (2012a). Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection. Springer Science & Business Media.

[2]Elmagarmid, A. K., Ipeirotis, P. G., & Verykios, V. S. (2007). Duplicate record detection: A survey. IEEE Transactions on knowledge and data engineering, 19(1), 1-16.

[3]P. Christen, "Febrl - an open source data cleaning, deduplication and record linkage system with a graphical user interface," Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '08, pp. 1065–1068, 2008.

[4]Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage.

[5]Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S. E., & Widom,

J. (2009). Swoosh: a generic approach to entity resolution. The VLDB Journal—The International Journal on Very Large Data Bases, 18(1), 255- 276.

[6]Thor, A., & Rahm, E. (2007, January). MOMA-A Mapping-based Object Matching System. In CIDR (pp. 247-258).

[7]Draisbach, U., & Naumann, F. (2010). DuDe: The duplicate detection toolkit. In Proceedings of the International Workshop on Quality in Databases (QDB).

[8]Jurczyk, P., Lu, J. J., Xiong, L., Cragan, J. D., & Correa, A. (2008). FRIL: a tool for comparative record linkage. In AMIA annual symposium proceedings (Vol. 2008, p. 440). American Medical Informatics Association.

[9]Leitão, L., Calado, P., & Weis, M. (2007, November). Structure-based inference of XML similarity for fuzzy duplicate detection. In Proceedings of the

sixteenth ACM conference on Conference on information and knowledge management (pp. 293-302). ACM.

[10]Bilenko, M., & Mooney, R. J. (2003a, August). Adaptive duplicate detection using learnable string similarity measures. In Proceedings of the ninth ACM

SIGKDD international conference on Knowledge discovery and data mining (pp. 39-48). ACM.

[11]Bilenko, M., & Mooney, R. J. (2003b, August). On evaluation and training-set construction for duplicate detection. In Proceedings of the KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation (pp. 7-12).

[12]Zhao, H., & Ram, S. (2005). Entity identification for heterogeneous database integration—a multiple classifier system approach and empirical evaluation. Information Systems, 30(2), 119-132.

[13]Chaudhuri, S., Chen, B. C., Ganti, V., & Kaushik, R. (2007, September). Example-driven design of efficient record matching queries. In Proceedings of the 33rd international conference on Very large data bases (pp. 327-338). VLDB Endowment.

[14]Reyes-Galaviz, O. F., Pedrycz, W., He, Z., & Pizzi, N. J. (2017). A supervised gradient-based learning algorithm for optimized entity resolution. Data & Knowledge Engineering, 112, 106-129.

[15]Tejada, S., Knoblock, C. A., & Minton, S. (2002, July). Learning domain-independent string transformation weights for high accuracy object identification. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 350-359). ACM.

[16]Sarawagi, S., & Bhamidipaty, A. (2002, July). Interactive deduplication using active learning. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 269-278). ACM.

[17] Bianco, G., Galante, R., Goncalves, M. A., Canuto, S., & Heuser, C. A. (2015). A practical and effective sampling selection strategy for large scale deduplication. IEEE Transactions on Knowledge and Data Engineering, 27(9), 2305-2319.

[18]Silva, R. M., Gonçalves, M. A., & Veloso, A. (2014). A Two stage active learning method for learning to rank. Journal of the Association for Information Science and Technology, 65(1), 109-128.

[19]Xiao, C., Wang, W., Lin, X., Yu, J. X., & Wang, G. (2011). Efficient similarity joins for near-duplicate detection. ACM Transactions on Database Systems (TODS), 36(3), 15.

[20]Wang, Q., Vatsalan, D., & Christen, P. (2015, May). Efficient interactive training selection for large-scale entity resolution. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 562-573). Springer, Cham.

[21]Volz, J., Bizer, C., Gaedke, M., & Kobilarov, G. (2009). Silk-a link discovery framework for the web of data. LDOW, 538.

[22]Isele, R., & Bizer, C. (2012). Learning expressive linkage rules using genetic programming. Proceedings of the VLDB Endowment, 5(11), 1638-1649.

[23]Isele, R., & Bizer, C. (2013). Active learning of expressive linkage rules using genetic programming. Web Semantics: Science, Services and Agents on the World Wide Web, 23, 2-15.

[24]Kejriwal, M., & Miranker, D. P. (2015, May). Semi-supervised instance matching using boosted classifiers. In European Semantic Web Conference (pp. 388-402). Springer, Cham.

[25]Papadakis, G., Ioannou, E., Palpanas, T., Niederee, C., & Nejdl, W. (2013). A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE Transactions on Knowledge and Data Engineering, 25(12), 2665-2682.

[26]Dietterich, T. G. (2000, June). Ensemble methods in machine learning. In International workshop on multiple classifier systems(pp. 1-15). Springer, Berlin, Heidelberg.

[27]Elfeky, M. G., Verykios, V. S., & Elmagarmid, A. K. (2002). TAILOR: A record linkage toolbox. In Proceedings 18th International Conference on Data Engineering (pp. 17-28). IEEE.

[28]Christen, P. (2008, January). Febrl: a freely available record linkage system with a graphical user interface. In Proceedings of the second Australasian workshop on Health data and knowledge management-Volume 80 (pp. 17- 25). Australian Computer Society, Inc..

[29]Christen, P. (2007, December). A two-step classification approach to unsupervised record linkage. In Proceedings of the sixth Australasian conference on Data mining and analytics-Volume 70 (pp. 111-119). Australian Computer Society, Inc..

[30]Nikolov, A., Uren, V., & Motta, E. (2007, October). KnoFuss: A comprehensive architecture for knowledge fusion. In Proceedings of the 4th international conference on Knowledge capture (pp. 185-186). ACM.

[31]Nikolov, A., d'Aquin, M., & Motta, E. (2012, May). Unsupervised learning of link discovery configuration. In Extended Semantic Web Conference (pp. 119- 133). Springer, Berlin, Heidelberg.

[32]Ngomo, A. C. N., Lehmann, J., Auer, S., & Höffner, K. (2011). Raven–active learning of link specifications. Ontology Matching, 2011.

[33]Ngomo, A. C. N., & Lyko, K. (2012, May). Eagle: Efficient active learning of link specifications using genetic programming. In Extended Semantic Web Conference (pp. 149-163). Springer, Berlin, Heidelberg.

[34]Ngomo, A. C. N., & Lyko, K. (2013, October). Unsupervised learning of link specifications: deterministic vs. non-deterministic. In Proceedings of the Ontology Matching Workshop.

[35]Higazy, A., El Tobely, T., Yousef, A. H., & Sarhan, A. (2013, November). Web-based Arabic/English duplicate record detection with nested blocking technique. In 2013 8th International Conference on Computer Engineering & Systems (ICCES) (pp. 313-318). IEEE.

[36]Gueddah, H., Yousfi, A., & Belkasmi, M. (2012). Introduction of the weight edition errors in the levenshtein distance. International Journal of Advanced Research in Artificial Intelligence, 1(5), 30-32.

[37]Ghafour, H. H. A., El-Bastawissy, A., & Heggazy, A. F. A. (2011, November). AEDA: Arabic edit distance algorithm Towards a new approach for Arabic name matching. In The 2011 International Conference on Computer Engineering & Systems(pp. 307-311). IEEE.

[38]El-Shishtawy, T. (2013). A hybrid algorithm for matching arabic names. arXiv preprint arXiv:1309.5657.

[39]Aqeel, S. U., Beitzel, S., Jensen, E., Grossman, D., & Frieder, O. (2006). On the development of name search techniques for Arabic. Journal of the American Society for Information Science and Technology, 57(6), 728-739.

[40]Yousef, A. H. (2014). Cross-language personal name mapping. arXiv preprint arXiv:1405.6293.

[41]Sidi, F., Panahy, P. H. S., Affendey, L. S., Jabar, M. A., Ibrahim, H., & Mustapha,

A. (2012, March). Data quality: A survey of data quality dimensions. In 2012 International Conference on Information Retrieval & Knowledge Management (pp. 300-304). IEEE.

[42]Rahm, E., & Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. the VLDB Journal, 10(4), 334-350.

[43]Bellahsene, Z., Bonifati, A., & Rahm, E. (2011). Schema matching and mapping (Vol. 20). Heidelberg (DE): Springer.

[44]Euzenat, J., & Shvaiko, P. (2007). Ontology matching (Vol. 18). Heidelberg: Springer.

[45]Fagin, R., Haas, L. M., Hernández, M., Miller, R. J., Popa, L., & Velegrakis, Y. (2009). Clio: Schema mapping creation and data exchange. In Conceptual modeling: foundations and applications(pp. 198-236). Springer, Berlin, Heidelberg.

[46]Bleiholder, J., & Naumann, F. (2009). Data fusion. ACM Computing Surveys (CSUR), 41(1), 1.

[47]Batini, C., & Scannapieco, M. (2006). Data Quality: Concepts, Methodologies and Techniques. Data-Centric Systems and Applications. Springer-Verlag, New York, Ine Secaucus, NJ, USA.

[48]Herzog, T. N., Scheuren, F. J., & Winkler, W. E. (2007). Data quality and record linkage techniques. Springer Science & Business Media.

[49]Winkler, W. E. (1995). Matching and record linkage. Business survey methods, 1, 355-384.

[50]Christen, P. (2007, December). A two-step classification approach to unsupervised record linkage. In Proceedings of the sixth Australasian

conference on Data mining and analytics-Volume 70 (pp. 111-119). Australian Computer Society, Inc..

[51]Hernández, M. A., & Stolfo, S. J. (1995, June). The merge/purge problem for large databases. In ACM Sigmod Record (Vol. 24, No. 2, pp. 127-138). ACM.

[52]Hernández, M. A., & Stolfo, S. J. (1998). Real-world data is dirty: Data cleansing and the merge/purge problem. Data mining and knowledge discovery, 2(1), 9-37.

[53] J. de Bruin, About - python record linkage toolkit 0.12 documentation. [Online]. Available: https://recordlinkage.readthedocs.io/en/latest/about.html, [Accessed: 2020-02-02].

[54] Sklearn.naive bayes.bernoullinb. [Online]. Available: https://scikit- learn. org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB. html, [Accessed: 2020-03-02].

[55] Sklearn.svm.linearsvc. [Online]. Available: https://scikit-learn.org/stable/ modules/generated/sklearn.svm.LinearSVC.html, [Accessed: 2020-03- 02].

[56] J535D165, J535d165/recordlinkage. [Online]. Available: https : / / github . com / J535D165/recordlinkage/blob/master/recordlinkage/classifiers. py, [Accessed: 2020-03-02].