Amélioration de l'alignement de propriétés d'ontologies grâce aux plongements et à l'extension d'alignement

Guilherme Sousa[1], Rinaldo Lima[2], Cassia Trojahn[1]

[1] Institut de Recherche en Informatique de Toulouse, Toulouse, France
[2] Universidade Rural de Pernambuco, Recife, Brazil

guilherme.santos-sousa@irit.fr, cassia.trojahn@irit.fr, rinaldo.jose@ufrpe.br

Résumé

Les approches d'alignement de propriétés de schémas de graphes de connaissances restent en retrait par rapport à la mise en correspondance des classes. Les propriétés impliquent souvent une variation plus importante dans leur dénomination (variation du verbe, mots fonctionnels, synonymes) que les classes. Cet article propose une approche d'alignement de propriétés qui combine les plongements et les extensions d'alignement afin d'améliorer les performances de la mise en correspondance de ce type d'entité. L'approche proposée est compétitive par rapport aux systèmes d'alignement existants.

Mots-clés

Alignement d'ontologies, alignement de propriétés, apprentissage automatique, plongements de mots

Abstract

Approaches for matching properties in knowledge graph schemas still behavior behind the matching of classes. Properties frequently involve a higher variation in naming (verb variation, functional words, common synonyms) than classes. This paper proposes a property-matching approach that combines embeddigns and alignment extension to improve the property matching performance. The proposed approach performs competitively with state-of-the-art alignment systems on well-known benchmarks in the field.

Keywords

Ontology matching, property alignment, machine learning, embeddings

1 Introduction

L'objectif du processus d'alignement d'ontologies est de trouver des correspondances entre les entités de différentes ontologies, généralement deux ontologies. L'une des principales tâches de ce processus consiste à trouver des correspondances entre propriétés. Une métrique courante pour trouver des correspondances entre propriétés est la métrique de similarité de chaînes de caractères, par exemple, la distance d'édition, comparant les étiquettes des entités. Cependant, l'utilisation de telles métriques ne permet pas de rappeler une partie des correspondances et ne permet pas de filtrer les entités homonymes puisqu'elles partagent les mêmes étiquettes mais ont des significations différentes [1, 22]. Récemment, les modèles de plongements ont attiré l'attention dans le domaine de l'alignement des ontologies et ont été appliqués dans plusieurs systèmes tels que TOM [15], Fine-Tom [14], ALOD2Vec [20], et AMD [27]. Toutefois, ces techniques se sont révélées utiles lorsqu'elles sont combinées à d'autres stratégies de mise en correspondance.

Les plongements de mots statiques couramment utilisés dans l'alignement d'ontologies, comme Glove [19] et Word2Vec [18], ont cependant des problèmes pour modéliser la notion de similarité. Cela est souligné dans les travaux sur l'analyse des sentiments [30, 13], car ils accordent une similarité forte à des mots présents dans le même contexte, tels que "Day" et "Night" qui ont, en fait, des significations opposées. Ce problème est accentué lors de l'utilisation des plongements statiques pour représenter les phrases, car le sens des mots change en fonction de leur contexte. Pour aider à résoudre ce problème, les modèles contextuels comme BERT [9] sont capables de générer différents plongements pour le même mot en fonction du contexte de la phrase, ce qui permet d'obtenir de meilleurs plongements de phrases. En particulier, pour la tâche d'alignement d'ontologies, ces modèles sont utiles pour représenter les informations textuelles dans les ontologies, puisque les ontologies peuvent être considérées comme un graphe de concepts dont les caractéristiques des nœuds sont représentées dans le texte en langue naturelle (dans les étiquettes et les annotations). En outre, ces modèles facilitent l'application des techniques de plongement de graphes, car ils nécessitent des vecteurs de caractéristiques de longueur fixe pour le traitement, alors que les caractéristiques textuelles des ontologies peuvent être de longueur variable, comme le montre la figure 1. Un exemple d'approche d'alignement qui illustre cette stratégie est DAEOM [29], où BERT [9] est utilisé pour extraire des vecteurs de caractéristiques de taille fixe à partir du contenu textuel de l'entité. Ces caractéristiques sont utilisées dans un réseau de neurones de graphes [24] pour mieux contextualiser les caractéristiques des nœuds. Cependant, comme cette approche est supervisée et qu'aucun jeu de données d'alignement de grande taille n'est disponible pour affiner ces modèles, ces approches ont des difficultés à atteindre des meilleures performances.

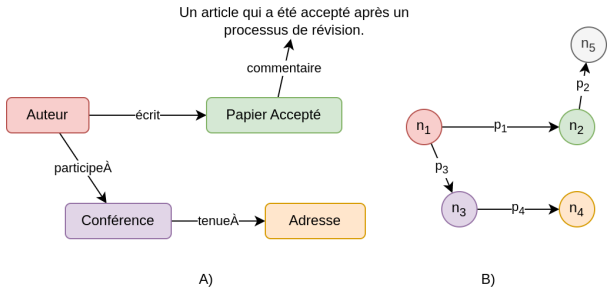

FIGURE 1 – A) Les entités de l'ontologie sont nommées à l'aide d'un texte en langage naturel de longueur variable. B) Les réseaux de neurones de graphe nécessitent des vecteurs de caractéristiques de longueur fixe qui doivent être générés pour pouvoir travailler sur des ontologies.

Cet article aborde le problème de l'alignement de propriétés à l'aide de plongements pré-entraînés et de l'extension de l'alignement [12], en étendant le système PropString [5]. Comme PropString n'utilise que des métriques de similarité lexicale, l'ajout des plongements pré-entraînés donne au système plus de flexibilité pour représenter des entités similaires avec des étiquettes différentes, tout en réduisant le besoin d'un réglage fin lorsqu'ils sont appliqués dans des étapes spécifiques. En complément, l'extension de l'alignement s'est avérée utile pour capturer les patrons d'alignement fréquents lorsque des propriétés similaires sont détectées. Le système proposé est évalué sur des jeux de données utilisés dans le cadre de la campagne OAEI. Les résultats montrent que l'utilisation des techniques proposées peut améliorer l'alignement des propriétés par rapport aux meilleurs systèmes participant à la track Conférence d'OAEI.

Le reste de l'article est organisé comme suit. La section 2 présente la définition du problème et la définition des représentations des propriétés considérées dans ce travail. La section 3 détaille l'architecture du système ainsi que les techniques utilisées. La section 4 présente les expériences menées pour évaluer la performance du système. La section 5 discute les travaux liés et, enfin, la section 6 conclut l'article.

2 Définition du problème

L'alignement de propriétés consiste à chercher de propriétés similaires entre deux ontologies différentes. Cette tâche peut être définie comme la recherche du meilleur ensemble de correspondances de propriétés A étant donné les ontologies en entrée O_1 et O_2. Dans cet article, nous définissons les propriétés comme toutes les entités S qui satisfont le prédicat $P(S) : \exists!D, \exists!R, domain(S, D) \wedge range(S, R)$. Compte tenu de cette définition, la tâche d'alignement est définie comme suit : étant donné la fonction de similarité des propriétés Sim, trouver l'ensemble de correspondances $A = \{(p_1, p_2) \in O_1 \times O_2 | Sim(p_1, p_2) > t\}$ produit en mesurant la similarité de chaque combinaison de paires de propriétés dans l'ontologie source et l'ontologie cible et en sélectionnant celles dont la similarité est supérieure à un seuil t donné. La Figure 2 présente l'architecture générale pour le calcul de similarité, mise en œuvre par la plupart de systèmes.

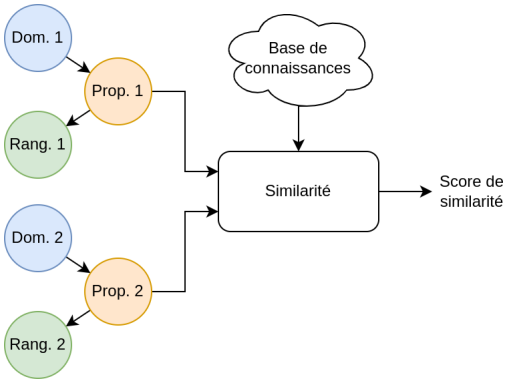

FIGURE 2 – Architecture générale pour le calcul de similarité. Étant donné deux propriétés avec leurs domaines (*rdfs :domain*) et leurs portées (*rdfs :range*), la fonction de similarité génère un score qui mesure leur degré de similarité.

L'une des difficultés rencontrées par les systèmes d'alignement de propriétés pour s'adapter à des domaines distincts réside dans les différentes représentations des propriétés. Par exemple, les ontologies peuvent représenter les propriétés en tant qu'entité d'un type tel que *owl :ObjectProperty* lié à une entité de domaine par le prédicat *rdfs :domain* et à une entité de la portée par le prédicat *rdfs :range*. Dans les graphes de connaissances, les propriétés sont plutôt représentées comme des prédicats qui relient un sujet à un objet. Dans ce cas, la même propriété peut relier des entités ayant une sémantique différente, même dans la même ontologie. Par exemple, *Author* et *Color* peuvent avoir une propriété *name* dans le même graphe de connaissances. Cet exemple montre que, dans ce cas, le domaine de cette propriété est composé de plusieurs entités, ce qui accroît la complexité de la fonction de similarité, car elle doit tenir compte de la manière de mesurer la similarité du domaine composé d'un groupe d'entités.

3 Approche proposée

L'approche proposée dans cet article est basée sur le système PropString intégrant les plongements et l'extension d'alignement. L'hypothèse générale du système original PropString concernant les propriétés similaires est qu'elles doivent avoir des domaines, des portées et des étiquettes similaires. La métrique utilisée pour mesurer la similarité entre les domaines et les portées est la métrique TF-IDF (*Term Frequency-Inverse Document Frequency*), qui s'est avérée être la meilleure métrique lexicale pour l'alignement des classes (dans leur proposition). Le TF-IDF est une métrique couramment utilisée pour la recherche d'information et repose sur l'hypothèse que les mots rares partagés entre deux objets les rendent similaires et que les termes fréquents qui apparaissent à de nombreux endroits ne sont

pas importants. Cette métrique suppose une hypothèse statistique sur la similarité, et puisqu'il s'agit d'une métrique globale, elle compte la fréquence des mots dans toutes les entités de l'ontologie.

Afin d'aligner les étiquettes de propriété, une version souple de TF-IDF est appliquée avec JaroWinkler comme métrique de similarité pour inclure également les mots lexicalement similaires dans les vecteurs de fréquence. Cette métrique est appliquée au concept central de l'étiquette de propriété introduit dans PropString [5]. Ce concept a été conçu par les auteurs de PropString en analysant les modèles de dénomination de propriété communs. Le concept central se compose du premier verbe de plus de quatre caractères ou, si ce verbe n'est pas trouvé, du premier nom et des adjectifs qui les accompagnent. Pour trouver le concept central, un étiqueteur POS est utilisé. Si la valeur minimale entre la similarité du domaine, de la portée et du concept central dépasse un certain seuil, les propriétés sont considérées comme similaires.

L'utilisation des similarités de domaine et de portées aide le système à filtrer les alignements de propriété faussement positifs. Un exemple de faux alignement possible est la propriété "writes" avec le domaine "Author" et la portée "Paper" dans l'ontologie O_1 et la propriété "writtenBy" avec le domaine "Paper" et la portée "Author" dans l'ontologie O_2. Les deux propriétés "writes" et "writtenBy" ont le même préfixe et peuvent avoir une grande similitude avec certaines métriques lexicales, mais la prise en compte des domaines et des portées donne une faible valeur de similarité à cette paire de propriétés en raison de leur différence.

Les améliorations proposées à PropString reposent sur deux observations principales. La première observation est que dans certains cas, le système PropString trouve des étiquettes et des portées de propriétés avec une grande similarité, mais la similarité de domaine est nulle. Ce cas se produit parce que les deux domaines sont synonymes de mots différents et que la métrique lexicale leur donne une faible similarité. Dans ce cas, des approches plus robustes sont nécessaires pour récupérer ces correspondances. Une deuxième observation est que les propriétés inverses des propriétés alignées sont plus susceptibles d'avoir une correspondance entre elles [12]. L'extension de l'alignement repose sur le principe de localité qui stipule que les entités proches d'entités précédemment mises en correspondance sont susceptibles d'être similaires, et les correspondances établies peuvent donc être utilisées pour détecter les correspondances potentielles entre les entités proches dans le voisinage du graphe. En ce sens, nous pouvons étendre les correspondances en incluant un alignement entre les inverses des propriétés comparées s'ils existent et si les propriétés comparées sont similaires. Cette approche peut améliorer le rappel du système car elle récupère les correspondances avec des relations sémantiques complexes qui seront incluses si elles ont des inverses 'alignables' plus simples.

Dans les sections suivantes, nous présentons l'architecture du système proposé et les modifications que nous avons apportées à PropString, y compris l'utilisation des plongements et des extensions de l'alignement. L'architecture est

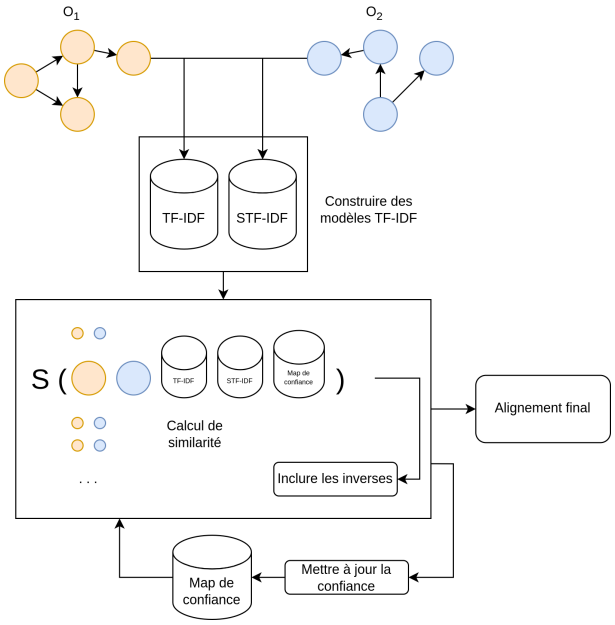

FIGURE 3 – Architecture du système proposé.

présentée dans la Figure 3 et se compose de deux éléments principaux : la construction du modèle TF-IDF et le calcul de la similarité.

3.1 Construction de modèles TF-IDF

Étant donné que les métriques TF-IDF et Soft TF-IF (décrites ci-dessous) sont globales, elles prennent en compte la fréquence des mots dans toutes les informations relatives aux étiquettes des entités dans les ontologies source et cible afin de construire les vecteurs de fréquence. De ce fait, la construction des modèles intervient avant le processus de mise en correspondance.

La métrique de similarité du modèle est composée du TF-IDF et du Soft TF-IDF. Le modèle TF-IDF est utilisé pour calculer la similarité entre les étiquettes de domaine et les étiquettes des portées. Pour construire le TF-IDF et sa version souple, un document virtuel est créé pour chaque entité dans les deux ontologies, contenant des informations relatives à chaque entité, puis les vecteurs sont calculés. Le document virtuel généré pour les classes de l'ontologie est composé du nom de la classe et, pour les propriétés, du nom de la propriété, du domaine et des portées. En cas de domaines et de portées multiples, toutes les valeurs sont ajoutées au document virtuel. Dans tous les cas, les étiquettes sont divisées à l'aide d'un tokenizer capable de diviser les conventions de nommage en camel case [2] ou en underscore [2]. A la fin de ce processus, les étiquettes sont mises en minuscules et jointes pour constituer le document final. Par exemple, la propriété *writePaper* avec le domaine *Author* et la portée *Paper* produit le document "author write paper paper paper". L'ensemble des documents est utilisé pour construire les modèles de fréquence, le vocabulaire et l'IDF. La IDF utilisée pour la métrique générale est énoncée dans l'équation 1, où N est le nombre de documents et $df(t)$ est le nombre de documents dans lesquels le terme t

apparaît.

$$idf(t) = \log(\frac{N+1}{df(t)+1}) + 1 \quad (1)$$

Après avoir construit le vocabulaire et calculé les valeurs IDF, la similarité entre les documents est mesurée en calculant le vecteur de fréquence des termes et en multipliant chaque mot par son IDF respectif. Les vecteurs sont ensuite normalisés et la similarité finale est calculée à l'aide de la similarité cosinus, soit d_1 et d_2 deux vecteurs comme décrit dans l'équation 2.

$$sim(d_1, d_2) = \frac{d_1 \cdot d_2}{||d_1|| \cdot ||d_2||} \quad (2)$$

Pour les étiquettes de propriétés, le système utilise le Soft TF-IDF. Le Soft TF-IDF inclut également les mots similaires qui dépassent un seuil de score de similarité dans le décompte des fréquences. Le Soft TF-IDF est construit en utilisant la métrique de Jaro-Winkler avec un seuil de 0.8.
Il est à noter que le système ne fonctionne qu'avec l'alignement monolingue des propriétés en anglais.

3.2 Plongements dans la similarité

Après la génération des modèles TF-IDF, le système calcule le score de similarité pour chaque paire de propriétés. Le score de similarité final est le minimum de trois valeurs de similarité basées sur les similitudes de domaine, de la portée et d'étiquette de propriété. Tout d'abord, la similarité de domaine est calculée à l'aide des vecteurs TF-IDF, et une similarité de plongement est utilisée comme solution de repli lorsque la métrique donne une similarité nulle. Cette approche est plus fiable lorsque les domaines comparés ne contiennent qu'un seul mot. Ainsi, dans ce cas, la similarité cosinus entre les plongements des mots du domaine remplace la similarité du domaine en utilisant les plongements pré-entraînés de la Finnish Internet Parsebank [17]. La similarité de la portée est calculée selon les mêmes étapes que le calcul de similarité de domaine, sans le repli de plongement, en filtrant d'abord les adjectifs de l'étiquette de la portée.
La première étape du calcul de la similarité des étiquettes de propriété consiste à retirer le dernier mot de l'étiquette de propriété s'il est égal au premier mot de l'étiquette de la portée. Par exemple, la propriété *hasPaper* avec le domaine *Author* et la portée *Paper* produira la phrase "Author has Paper Paper". Après le traitement, la phrase "Author has Paper" est retenue. Ensuite, l'étiquette de propriété est étiquetée à l'aide d'un marqueur POS et le système utilise le concept central pour calculer la similarité à l'aide des vecteurs Soft TF-IDF. Le premier verbe avec plus de quatre caractères (cette heuristique est appliquée pour filtrer les verbes anglais courts comme "has" ou "let" et est considérée après l'analyse des patrons de noms de propriétés [5]) ou le nom avec ses adjectifs, si aucun verbe n'est trouvé, est utilisé comme entité centrale de l'étiquette de propriété. Le système applique la même stratégie que celle décrite précédemment pour la similarité des domaines aux étiquettes de propriété. Dans ce cas, un modèle de similarité des phrases

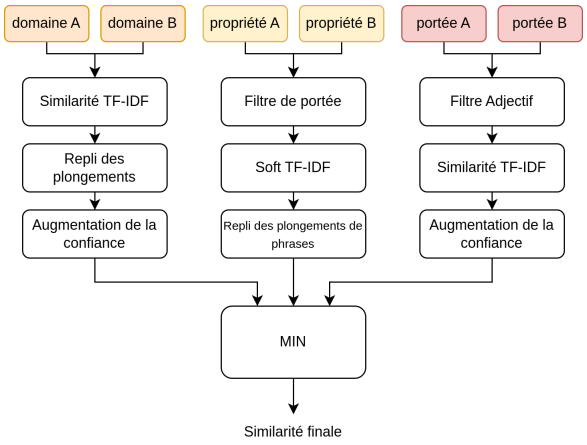

FIGURE 4 – Pipeline de similarité des propriétés.

est utilisé pour générer des représentations de plongement des étiquettes de propriété, et ces représentations sont comparées à l'aide de la similarité cosinusoïdale. Le repli se produit lorsque la similarité de domaine et de la portée est supérieure à 0.9 et que la similarité d'étiquette est inférieure à 0.1. Le modèle de similarité de phrases [25] utilisé dans ce travail provient du référentiel HuggingFace [28]. La phrase utilisée est composée de l'étiquette de propriété concaténée avec des étiquettes de la portée. Dans le cas des propriétés d'objet, l'étiquette de la classe présente dans la portée est utilisée. Une illustration de la procédure de calcul de la similarité est présentée dans la Figure 4.
En raison des différentes structures syntaxiques présentes dans les étiquettes de propriétés et les étiquettes de classe, deux modèles de plongement distincts ont été appliqués. La majorité des propriétés contiennent des verbes qui sont mieux capturés à l'aide de modèles de plongement de phrases en raison de la distribution similaire qu'ils ont avec les données d'apprentissage. Les tests empiriques appliquant les plongements de phrases aux étiquettes de classe donnent de moins bons résultats. Pour cette raison, l'incorporation de mots est appliquée lorsque la similarité du domaine est calculée puisque les domaines de propriété sont des classes.

3.3 Extension de l'alignement

Après le calcul final de la similarité des propriétés, le système ajoute la paire de propriétés à l'ensemble de correspondances si la valeur de similarité dépasse le seuil donné. Si certaines des correspondances précédentes contiennent l'une des propriétés alignées, seule la paire de domaines présentant une similarité élevée est conservée. En outre, sur la base du principe de localité, les inverses des propriétés alignées sont inclus dans l'ensemble d'alignement final car ils sont plus susceptibles d'être alignés. Basé également sur le principe de localité, les alignements sont encore étendus en utilisant les informations structurelles des propriétés concernant les domaines. Cette extension est réalisée en maintenant une carte de confiance qui stocke la similarité entre les paires de domaines qui augmentent la similarité

des propriétés entre les domaines de propriété précédemment alignés. La carte de confiance est une structure clé-valeur dans laquelle la clé est une paire d'entités et la valeur est la similarité entre elles. Il est utilisé dans l'étape de calcul de similarité pour augmenter la similarité des domaines s'ils apparaissent comme des domaines dans des propriétés précédemment alignées. Ensuite, le système augmente la similarité de la paire de domaines de propriétés alignées dans la carte de confiance de 0.66 (établi empiriquement). Comme la carte de confiance est mise à jour, plusieurs étapes sont nécessaires pour découvrir de nouvelles correspondances basées sur ce processus d'inférence, car les calculs de similarité incluront la valeur de similarité mise à jour.

4 Expérimentations

4.1 Évaluation sur le jeu de données *Conférence*

La première série d'expériences a été menée sur le jeu de données Conférence [1] disponible dans le cadre des campagnes d'évaluation OAEI. Ce jeu de données consiste en 21 paires d'alignements entre 7 ontologies. Sur les 21 paires, 7 alignements de référence ne contiennent aucune propriété. L'évaluation ne prend en compte que les paires qui contiennent des alignements de référence de propriété. La performance du système est évaluée pour chaque paire individuellement et en considérant le résultat global qui est le total des alignements dans toutes les paires d'alignements. Les résultats globaux des systèmes évalués peuvent être consultés sur la page OAEI 2021 [2]. Les résultats pour chaque paire sont calculés à partir des alignements de référence basés sur les alignements produits par les systèmes participants pour l'année 2021 et sont équivalents à l'évaluation ra1-M2 (alignements de référence entre propriétés). Les systèmes de référence sont ceux qui ont participé à la campagne et sont comparés à l'implémentation de base équivalente à PropString et à la version améliorée (appelé ici PropMatch) disponible sur le Gitlab de l'IRIT sur licence MIT [3]. Le seuil de similarité utilisé par le système proposé dans toutes les évaluations est de 0.65. Le système obtient la meilleure métrique F-measure avec cette valeur lorsqu'il a été testé avec un seuil allant de 0 à 1 par pas de 0.05 avec toutes les modifications. La progression des performances est illustrée dans la Figure 5.

PropMatch a été implémenté en Python à partir de l'implémentation Java originale de PropString et amélioré par l'ajout des modifications décrites dans la section 3.2. Pour cette expérience, l'impact progressif de chaque modification est évalué dans les mêmes conditions, ainsi que le nombre de comparaisons totales dans toutes les paires d'ontologies. Le résultat de l'évaluation est présenté dans la Table 1.

Le système proposé a également été comparé aux systèmes qui ont participé à OAEI en 2021, ainsi que avec PropString. Les résultats sont présentés dans la Table 2. PropMatch surpasse le meilleur système AML de 11% en rappel et 6% en F-mesure. Il a une moins bonne précision par rapport à PropString et à AML. Ce résultat est dû à l'impact de l'utilisation de plongements et de l'inclusion d'inverses, comme cela a été observé dans les résultats des modifications progressives. Les plongements utilisés ont donné des valeurs de similarité élevées à des éléments qui pouvaient être liés mais qui n'étaient pas similaires, ce qui a entraîné une baisse de la précision du système. Un autre problème est l'inclusion aveugle des inverses, car certains inverses ne sont pas présents dans l'alignement de référence, et ce schéma peut ne pas être présent dans toutes les formulations de l'ontologie.

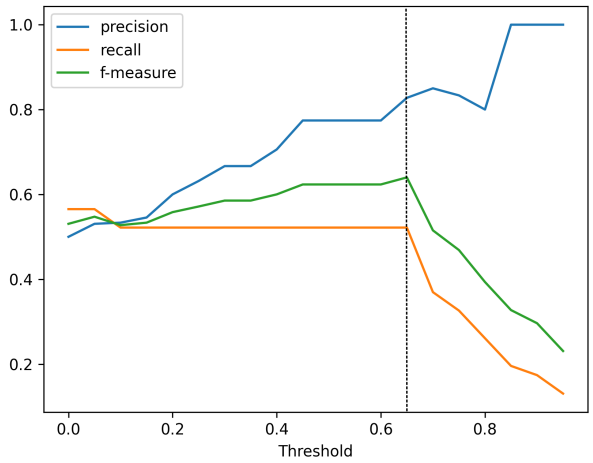

FIGURE 5 – Progression de la précision, du rappel et de la F-mesure à mesure que le seuil augmente.

Enfin, le système proposé a été comparé, pour chaque paire d'ontologies, aux systèmes AML et LogMap, qui ont obtenu la meilleure F-mesure dans l'alignement des propriétés. Comme le montre la Table 3, PropMatch obtient la meilleure F-mesure dans 8 paires, AML dans 6 paires et LogMap dans 4 paires d'alignement. Il peut récupérer les alignements dans 12 paires, tandis que AML et LogMap récupèrent les alignements dans 6 paires. AML atteint une précision de 100% dans toutes les paires d'alignements où un alignement est trouvé, tandis que LogMap et le système proposé peuvent atteindre une précision de 100% dans 67% des paires d'alignements. Aucun des systèmes ne peut récupérer d'alignements entre Conference et Iasted. Dans cette paire d'alignements, le seul alignement de propriétés existant est entre *Person contributes Conference_document* et *Person write Item*, décrits respectivement dans la propriété de domaine et de la portée. La similarité des chaînes lexicales ne permet pas de retrouver la similarité entre les libellés de *contributes* et *write* car leur similarité lexicale est faible, et il en va de même pour *Conference_document* et *Item*. En fait, dans l'ontologie Iasted, la classe *Item* a une sous-classe nommé *Document* qui peut être utilisée comme information pour récupérer cette correspondance.

1. http ://oaei.ontologymatching.org/2021/conference/index.html
2. http ://oaei.ontologymatching.org/2021/results/conference/index.html
3. https ://gitlab.irit.fr/melodi/ontology-matching/propmatch

Description	Précision	Rappel	F-mesure
PropString	**1.00**	0.28	0.44
Utilisation des plongements dans la similarité des domaines	0.84	0.35	0.49
Addition d'inverses	0.81	0.39	0.53
Similitude de plongements des phrases appliquée aux étiquettes des propriétés	0.68	0.41	0.51
Filtre des mots répétés et les adjectifs dans les étiquettes des portées	0.71	0.48	0.57
similarité des domaines pour les propriétés précédemment alignées	0.73	**0.52**	0.61
Limitation de la cardinalité (1-1)	0.83	**0.52**	**0.64**

TABLE 1 – Progression des performances avec l'application de modifications.

Nom	Précision	Rappel	F-mesure
PropMatch	0.83	**0.52**	**0.64**
AML	**1.0**	0.41	0.58
PropString	**1.0**	0.28	0.44
LogMap	0.62	0.28	0.39
GMap	0.56	0.2	0.29
Wikitionary	0.24	0.28	0.26
TOM	0.27	0.24	0.25
ALOD2Vec	0.22	0.3	0.25
LogMapLt	0.24	0.22	0.23
FineTOM	0.24	0.22	0.23
OTMapOnto	0.13	0.48	0.2
edna	0.21	0.11	0.14
StringEquiv	0.07	0.02	0.03

TABLE 2 – Résultats obtenus par les systèmes participant à l'OAEI 2021.

4.2 Évaluation sur d'autres jeux de données

L'autre jeu de données de teste est celui des *OAEI knowledge graph*[4]. Dans ce jeu de données, huit graphes de connaissances sont alignés en 5 paires. Ces graphes de connaissances représentent les propriétés comme un prédicat qui relie une instance à une valeur de la portée. Mais dans cette structure, plusieurs instances peuvent partager la même propriété. Par exemple, des auteurs, des films et des entreprises peuvent avoir le même nom de propriété avec des portées différentes. La propriété peut être considérée comme ayant un domaine et une portée complexes. Le système sélectionne la paire domaine/portée la plus fréquente contenant les types d'instances pour servir de domaine et de portée uniques pour la propriété. L'étape de chargement du graphe effectue ces transformations afin que le système puisse voir la même structure pour différentes représentations des propriétés.

Le système est comparé à ceux qui ont participé à la campagne OAEI pour la track graphe de connaissances, en fonctionnant avec une valeur de seuil de 0.0 et avec 1 itération. Les résultats sont présentés dans la Table 5. Le seuil sélectionné était 0 car c'est le seuil qui produit des correspondances. Comme on peut le voir dans les résultats, sur les 8 systèmes évalués, seuls 4 systèmes ont été capables de produire des alignements, le système proposé étant l'un d'entre eux. En ce sens, PropMatch obtient de meilleurs résultats globaux entre les jeux de données qu'AMD et LogMap. Cependant, il est possible d'observer qu'il ne peut pas atteindre la performance de base dans cette track. Ce fait est dû à l'utilisation par le système de la valeur de similarité minimale entre le domaine, la portée et la similarité des étiquettes, ce qui rend difficile l'alignement du domaine et de la portée, même lorsque les étiquettes des propriétés sont identiques.

Afin de mieux analyser l'impact de l'utilisation des similitudes, nous avons testé PropMatch avec différentes combinaisons de similitudes, toutes avec un seuil de 0. Quatre combinaisons de similarité ont été testées : uniquement les étiquettes de propriété (p), le domaine et la propriété (d+p), la propriété et la portée (p+r) et la configuration de base qui prend en compte toutes les combinaisons (d+p+r). Les résultats de cette évaluation sont présentés dans la Table 4. Il est possible de voir dans les résultats que l'utilisation de la seule similarité d'étiquette de propriété (p) permet d'obtenir des résultats similaires en termes de rappel aux systèmes les plus performants, avec une précision réduite en raison de la faible valeur du seuil. Cependant, avec l'ajout des similitudes de domaine et de la portée (d+p, p+r, et d+p+r), les performances du système diminuent.

Comme nous l'avons vu ci-dessus, une étape de pré-traitement doit être appliquée pour que le système puisse traiter des domaines et des portées complexes. En raison de cette étape de pré-traitement, le système reçoit une seule paire domaine/portée qui peut ne pas représenter toutes les combinaisons possibles décrites par la propriété. De plus, étant donné que l'alignement de chaînes obtient des résultats élevés en ne tenant compte que des étiquettes des propriétés, le repli des plongements se produit rarement, de sorte que l'utilisation des plongements par le système n'a qu'un faible impact dans ce domaine. Un autre problème est que, aucune propriété inverse n'est présente, de sorte que la stratégie d'extension de l'alignement n'est pas appliquée. Les domaines sont complexes et la stratégie proposée ne peut pas représenter les informations nécessaires pour produire une comparaison de similarité suffisante.

5 Travaux liés

L'un des premiers travaux comparant les performances de différentes techniques basées sur les chaînes de caractères dans l'alignement d'ontologies est [3]. Ces travaux comparent différentes mesures de similarité pour l'alignement

4. http://oaei.ontologymatching.org/2021/knowledgegraph/index.html

Paire	Total*	LogMap	AML	PropString	PropMatch
conference-iasted	1	0.00	0.00	0.00	0.00
cmt-conference	3	**0.50**	**0.50**	**0.50**	0.33
edas-ekaw	4	0.40	0.00	0.00	**0.67**
conference-ekaw	2	**0.40**	0.00	0.00	0.00
cmt-ekaw	3	0.00	0.00	0.00	**1.00**
edas-sigkdd	4	0.00	**0.86**	0.67	0.57
cmt-confOf	6	0.00	0.00	0.29	**0.80**
confOf-sigkdd	1	0.00	0.00	**1.00**	0.67
cmt-sigkdd	2	**1.00**	**1.00**	**1.00**	**1.00**
conference-edas	3	0.67	0.00	**0.80**	**0.80**
cmt-edas	5	0.00	**0.57**	0.00	0.28
conference-sigkdd	3	**0.50**	**0.50**	**0.50**	**0.50**
confOf-edas	5	0.00	**0.80**	0.33	0.75
conference-confOf	4	0.00	0.00	**0.67**	**0.67**

TABLE 3 – Évaluation du système proposé, de l'AML et de LogMap dans le cadre de la conférence en termes de F-mesure. *Nombre d'alignements de référence.

	p			d+p			p+r			d+p+r		
Paire	P	R	F-m	P	R	F-m	P	R	F-m	P	R	F-m
starwars-swtor	0.29	0.96	0.45	0.18	0.54	0.27	0.26	0.59	0.36	0.21	0.52	0.30
malpha-stexpand	0.32	0.95	0.48	0.24	0.65	0.35	0.24	0.43	0.30	0.16	0.32	0.21
starwars-swg	0.21	1.00	0.34	0.14	0.65	0.23	0.16	0.50	0.24	0.11	0.35	0.16
mcu-marvel	0.22	0.91	0.35	0.09	0.36	0.14	0.19	0.45	0.26	0.13	0.36	0.19
malpha-mbeta	0.29	0.90	0.44	0.22	0.63	0.32	0.19	0.43	0.27	0.19	0.45	0.26

TABLE 4 – Le nombre de combinaisons de similitudes différentes dans les paires de référence du graphique de connaissances. Dans un souci d'espace, P (précision), R (rappel) et F-m (F-mesure).

	mcu-marvel			malpha-mbeta			malpha-stexpand			starwars-swg			starwars-swtor		
Paire	P	R	F-m	P	R	F-m	P	R	F-m	P	R	F-m	P	R	F-m
AMD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ATMatcher	0.91	0.91	0.91	0.98	0.92	0.95	0.95	0.95	0.95	1.00	1.00	1.00	1.00	0.98	0.99
BaselineLabel	1.00	0.36	0.53	1.00	0.34	0.51	0.97	0.68	0.80	1.00	1.00	1.00	1.00	0.98	0.99
KGMatcher	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LogMap	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LSMatch	0.82	0.82	0.82	0.62	0.58	0.60	0.62	0.61	0.62	0.72	0.65	0.68	0.88	0.79	0.83
Matcha	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
PropMatch	0.13	0.36	0.19	0.19	0.45	0.26	0.16	0.32	0.21	0.11	0.35	0.16	0.21	0.52	0.30

TABLE 5 – Le résultat des systèmes qui participent à la track *Knowledge Graph*.

de chaînes, telles que Levenshtein et Jaccard, ainsi que des techniques de prétraitement telles que le stemming et l'élimination des mots vides dans différents jeux de données. Les travaux montrent également que les métriques sont moins performantes lorsqu'elles sont utilisées pour faire correspondre des propriétés plutôt que des classes. Sur la base de cette analyse, les mêmes auteurs ont développé un système appelé PropString [5] avec des performances améliorées pour la mise en correspondance des propriétés en tenant compte de la similarité entre les domaines et les portées. Une autre technique importante utilisée dans le système est l'utilisation de l'étiquetage de la partie du discours (POS) [8] pour récupérer l'entité centrale des étiquettes de propriété, c'est-à-dire l'entité composé du verbe dans l'étiquette de propriété ou du nom et de ses adjectifs associés lorsqu'aucun verbe n'est trouvé. Cette méthode atteint une précision de 100% lorsqu'elle est évaluée dans le cadre du jeu de données Conférence [4]. Cependant, la métrique utilisée dans le système n'est pas en mesure de traiter les entités conceptuelles qui sont similaires mais qui n'ont pas d'étiquettes similaires. Par exemple, la correspondance entre les propriétés *hasLocation* et *heldIn*, qui ont une sémantique similaire et une faible similarité lexicale. En outre, les domaines de ces propriétés sont *Place* et *Location*, qui peuvent être considérés comme des synonymes et présentent une faible similarité lexicale. Ces exemples montrent que les méthodes utilisées pour comparer la similarité ne sont pas suffisamment robustes pour récupérer

des correspondances avec des structures de texte plus complexes.

Dans la compétition OAEI, AML [11] est le système qui a obtenu les meilleurs résultats dans l'alignement des propriétés dans le jeu de données Conférence. Il dispose d'une méthode d'alignement spécifique pour aligner les propriétés et utilise plusieurs stratégies de correspondance de chaînes enrichies de synonymes pour mesurer la similarité entre les étiquettes des propriétés. Outre sa précision élevée de 100%, son rappel est de 41%, le système ne parvient toujours pas à récupérer certaines correspondances. Ces correspondances sont sémantiquement liés et contiennent des relations logiques qui ne peuvent pas être récupérées avec la seule métrique de similarité des chaînes de caractères. Ces problèmes posent des difficultés à l'utilisation exclusive des techniques d'alignement de chaînes et montrent la nécessité de prendre en compte le contexte de l'entité dans les mesures de similarité.

La plupart des modèles utilisés dans l'alignement d'ontologies ne peuvent toujours pas répondre aux exigences de la métrique de similarité pure, car ces plongements capturent certaines relations qui ne décrivent forcément une similarité réelle. Dans la tâche d'analyse des sentiments dans le domaine du traitement du langage naturel (NLP), par exemple, certains travaux [30, 13] ont constaté que des mots apparentés tels que "bon" et "mauvais" peuvent apparaître dans un contexte similaire mais avoir des significations sémantiques opposées. Étant donné que l'apparition dans un contexte similaire ne garantit pas la similarité entre deux entités, certains travaux [31, 10, 16] ont montré que le fait d'affiner les plongements en fonction de leurs cas d'utilisation spécifiques peut conduire à de meilleurs résultats. Par exemple, les entités *Author* et *Book* sont liées et la majorité des modèles de langage peuvent donner une forte similarité entre les deux en se basant uniquement sur la similarité des étiquettes. Mais comme il ne s'agit pas de la même entité, dans l'alignement ontologique, cette similarité peut conduire le système à classer à tort ces entités comme équivalentes.

Les modèles utilisés dans les systèmes d'alignement tels que RDF2Vec [21] dans Alod2Vec [20], OWL2Vec [6] utilisé dans une version de LogMap [7], ou encore RotatE [23] utilisé dans AMD [26], sont basés sur l'hypothèse distributionnelle qui donne une similarité élevée pour les entités qui apparaissent dans le même contexte comme décrit précédemment comme Auteur et Livre ou Bon et Mauvais et limitant la capacité de ces systèmes à trouver tous les alignements pertinents. Dans ces cas, des techniques telles que l'extension de l'alignement et la réparation des correspondances [12] peuvent être complémentaire.

6 Conclusion et travaux futurs

Dans cet article, nous avons présenté des améliorations au système PropString. Notre proposition inclut l'utilisation des plongements et de l'extension de l'alignement. Cependant, bien qu'il soit capable de produire des alignements dans différentes représentations d'ontologies, le système a encore des problèmes pour mesurer la similarité des propriétés qui ont plusieurs entités dans le domaine ou dans la portée.

Pour améliorer encore les résultats du système, le traitement de domaines et portées complexes doit être adressé. Nous voudrons également exploiter l'utilisation d'autres modèles de langages, capables de mieux exprimer la similarité sémantique. Comme les modèles plus sophistiqués ont le potentiel de mieux encoder la similarité sémantique, une partie de l'architecture du système peut migrer vers une utilisation de différents modèles de plongements, ce qui donne au système plus de flexibilité et de généralité pour l'alignement des ontologies de différents domaines.

Références

[1] Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. Recent trends in word sense disambiguation : A survey. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4330–4338. ijcai.org, 2021.

[2] Dave Binkley, Marcia Davis, Dawn Lawrie, and Christopher Morrell. To camelcase or under_score. In *2009 IEEE 17th International Conference on Program Comprehension*, pages 158–167. IEEE, 2009.

[3] Michelle Cheatham and Pascal Hitzler. String similarity metrics for ontology alignment. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha F. Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II*, volume 8219 of *Lecture Notes in Computer Science*, pages 294–309. Springer, 2013.

[4] Michelle Cheatham and Pascal Hitzler. Conference v2.0 : An uncertain version of the OAEI conference benchmark. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig A. Knoblock, Denny Vrandecic, Paul Groth, Natasha F. Noy, Krzysztof Janowicz, and Carole A. Goble, editors, *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, volume 8797 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2014.

[5] Michelle Cheatham and Pascal Hitzler. The properties of property alignment. In Pavel Shvaiko, Jérôme Euzenat, Ming Mao, Ernesto Jiménez-Ruiz, Juanzi Li, and Axel Ngonga, editors, *Proceedings of the 9th International Workshop on Ontology Matching collocated with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Trentino, Italy, October 20, 2014*, volume 1317 of *CEUR Workshop Proceedings*, pages 13–24. CEUR-WS.org, 2014.

[6] Jiaoyan Chen, Pan Hu, Ernesto Jiménez-Ruiz, Ole Magnus Holter, Denvar Antonyrajah, and Ian

Horrocks. Owl2vec* : embedding of OWL ontologies. *Mach. Learn.*, 110(7) :1813–1845, 2021.

[7] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, Denvar Antonyrajah, Ali Hadian, and Jaehun Lee. Augmenting ontology alignment by semantic embedding and distant supervision. In Ruben Verborgh, Katja Hose, Heiko Paulheim, Pierre-Antoine Champin, Maria Maleshkova, Óscar Corcho, Petar Ristoski, and Mehwish Alam, editors, *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings*, volume 12731 of *Lecture Notes in Computer Science*, pages 392–408. Springer, 2021.

[8] Alebachew Chiche and Betselot Yitagesu. Part of speech tagging : a systematic review of deep learning and machine learning approaches. *J. Big Data*, 9(1) :10, 2022.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT : pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[10] Mennatallah El-Assady, Rebecca Kehlbeck, Christopher Collins, Daniel A. Keim, and Oliver Deussen. Semantic concept spaces : Guided topic model refinement using word-embedding projections. *IEEE Trans. Vis. Comput. Graph.*, 26(1) :1001–1011, 2020.

[11] Daniel Faria, Catia Pesquita, Emanuel Santos, Matteo Palmonari, Isabel F. Cruz, and Francisco M. Couto. The agreementmakerlight ontology matching system. In Robert Meersman, Hervé Panetto, Tharam S. Dillon, Johann Eder, Zohra Bellahsene, Norbert Ritter, Pieter De Leenheer, and Dejing Dou, editors, *On the Move to Meaningful Internet Systems : OTM 2013 Conferences - Confederated International Conferences : CoopIS, DOA-Trusted Cloud, and ODBASE 2013, Graz, Austria, September 9-13, 2013. Proceedings*, volume 8185 of *Lecture Notes in Computer Science*, pages 527–541. Springer, 2013.

[12] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. Logmap : Logic-based and scalable ontology matching. In Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist, editors, *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, volume 7031 of *Lecture Notes in Computer Science*, pages 273–288. Springer, 2011.

[13] Mohammed Kasri, Marouane Birjali, Mohamed Nabil, Abderrahim Beni Hssane, Anas El-Ansari, and Mohamed El Fissaoui. Refining word embeddings with sentiment information for sentiment analysis. *J. ICT Stand.*, 10(3), 2022.

[14] Leon Knorr and Jan Portisch. Fine-tom matcher results for OAEI 2021. In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, and Cássia Trojahn, editors, *Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 25, 2021*, volume 3063 of *CEUR Workshop Proceedings*, pages 144–151. CEUR-WS.org, 2021.

[15] Daniel Kossack, Niklas Borg, Leon Knorr, and Jan Portisch. TOM matcher results for OAEI 2021. In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, and Cássia Trojahn, editors, *Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 25, 2021*, volume 3063 of *CEUR Workshop Proceedings*, pages 193–198. CEUR-WS.org, 2021.

[16] Guoxuan Li. Deepfca : Matching biomedical ontologies using formal concept analysis embedding techniques. In *ICMHI 2020 : 4th International Conference on Medical and Health Informatics, Kamakura City, Japan, August, 2020*, pages 259–265. ACM, 2020.

[17] Juhani Luotolahti, Jenna Kanerva, Veronika Laippala, Sampo Pyysalo, and Filip Ginter. Towards universal web parsebanks. In Eva Hajicová and Joakim Nivre, editors, *Proceedings of the Third International Conference on Dependency Linguistics, DepLing 2015, August 24-26 2015, Uppsala University, Uppsala, Sweden*, pages 211–220. Uppsala University, Department of Linguistics and Philology, 2015.

[18] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013.

[19] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove : Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

[20] Jan Portisch and Heiko Paulheim. Alod2vec matcher results for OAEI 2021. In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, and

Cássia Trojahn, editors, *Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 25, 2021*, volume 3063 of *CEUR Workshop Proceedings*, pages 117–123. CEUR-WS.org, 2021.

[21] Petar Ristoski and Heiko Paulheim. Rdf2vec : RDF graph embeddings for data mining. In Paul Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, volume 9981 of *Lecture Notes in Computer Science*, pages 498–514, 2016.

[22] Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. With more contexts comes better performance : Contextualized sense embeddings for all-round word sense disambiguation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3528–3539. Association for Computational Linguistics, 2020.

[23] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate : Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[24] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20) :10–48550, 2017.

[25] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm : Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33 : Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[26] Zhu Wang. AMD results for OAEI 2022. In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, and Cássia Trojahn, editors, *Proceedings of the 17th International Workshop on Ontology Matching (OM 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022), Hangzhou, China, held as a virtual conference, October 23, 2022*, volume 3324 of *CEUR Workshop Proceedings*, pages 145–152. CEUR-WS.org, 2022.

[27] Zhu Wang and Isabel F. Cruz. Agreementmakerdeep results for OAEI 2021. In Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, and Cássia Trojahn, editors, *Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 25, 2021*, volume 3063 of *CEUR Workshop Proceedings*, pages 124–130. CEUR-WS.org, 2021.

[28] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers : State-of-the-art natural language processing. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing : System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics, 2020.

[29] Jifang Wu, Jianghua Lv, Haoming Guo, and Shilong Ma. Daeom : A deep attentional embedding approach for biomedical ontology matching. *Applied Sciences*, 10(21) :7909, 2020.

[30] Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. Refining word embeddings for sentiment analysis. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 534–539. Association for Computational Linguistics, 2017.

[31] Michelle Yuan, Mozhi Zhang, Benjamin Van Durme, Leah Findlater, and Jordan L. Boyd-Graber. Interactive refinement of cross-lingual word embeddings. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5984–5996. Association for Computational Linguistics, 2020.