Dipartimento di Informatica
Corso di Laurea Magistrale in Informatica
Esame di Artificial Intelligence

Compass Aligned Graph Embeddings

Alberici Federico - 808058
Gherardi Alessandro - 817084

Anno Accademico 2021 - 2022

Indice

Abstract

Studio dell'approccio di word embedding alignment di CADE applicato su corpus estratto da knowledge graph. L'idea è quella di comprendere le similarità di due knowledge base, senza però dover partire da testo scritto. Si parte dalla stessa entità in due knowledge base diverse, da cui, attraverso strategie di random walk si estraggono dei corpus relativi all'argomento. Attraverso i sameAs link di una delle due knowledge base si unifica il vocabolario in modo da poter utilizzare CADE per creare e allineare le rappresentazioni vettoriali delle entità estratte. Queste sono, dunque, utilizzate per analizzarne similitudini e creare un nuovo grafo utilizzando un approccio di unione basato su similarità.

Background Knowledge

Knowledge Graph

Un knowledge graph rappresenta una rete composta da entità del mondo reale e ne codifica le relazioni sotto forma di grafo. Ogni entità (es. un oggetto, concetto, evento...) è rappresentata da un nodo ed ogni arco etichettato definisce la relazione tra i nodi (Fig. 1).

Fig. 1: A rappresenta il soggetto, B il predicato, C l'oggetto.

Questa collezione di elementi interconnessi è descritta attraverso una semantica formale che permette sia alle persone che ai computer di elaborarli in modo efficiente e non ambiguo, abilitando inoltre l'utilizzo di algoritmi di inferenza che permettono di estrarre nuova informazione non presente originariamente. Ciò è possibile attraverso le **ontologies**, ovvero uno "*schema*" del grafo, che definiscono il significato dei dati, assicurandone una comprensione condivisa. Tra le principali si trovano:

- **Classi**: classificazione di un entità rispetto ad una classe o gerarchia di classi.
- **Categorie**: descrivono aspetti della semantica di un entità.
- **Tipologie di relazioni**: forniscono informazioni sulla tipologia di una relazione, anche attraverso proprietà formali (simmetria, transitività...)

Il tutto viene descritto attraverso il **Resource Description Framework**, o RDF, un modello dati standardizzato, semplice e flessibile in cui ogni fatto è rappresentato da una tripla soggetto, predicato e oggetto (Fig. 1).
Lo standard utilizzato per recuperare informazioni da una base di dati RDF è il linguaggio **SPARQL** e consente di interrogare direttamente i knowledge graph.
I grafi utilizzati nel seguente studio sono:

- **DBpedia**: nato nel 2007 è un open knowledge graph (OKG) ottenuto estraendo contenuti strutturati dalle informazioni presenti in vari progetti Wikimedia, in particolare da Wikipedia. Ha diversi vantaggi rispetto alle basi di conoscenza esistenti: copre molti domini, rappresenta l'accordo della comunità, si evolve automaticamente con i cambiamenti di Wikipedia ed è multilingua. Ogni entità è rappresentata da un URI basato sul titolo della pagina Wikipedia di provenienza.

- **Wikidata**: progetto più recente (2012), mira a fornire una base di conoscenza gratuita che chiunque può modificare, non generata automaticamente ma gestita manualmente dai suoi utenti (reali e bot), assicurando una migliore qualità e specificità dei dati. Rispetto a DBpedia mantiene aspetti temporali dei dati (es. popolazione per differenti punti nel tempo), memorizza la sorgente di ogni dato e rappresenta ogni entità con un ID univoco (es. Q38).

Entrambe forniscono un endpoint HTTP per eseguire query in linguaggio SPARQL.

RDF2vec

RDF2vec [1] è un approccio utilizzato per rappresentare parole in uno spazio vettoriale (embedding) utilizzando una base di dati RDF per l'estrazione di informazioni. Per la creazione degli embedding viene sfruttato il modello Word2vec[2]: dato in input un insieme di frasi viene allenata una rete neurale, proposta in due varianti che si differenziano nel task eseguito:

- **Continuous Bag Of Word (CBOW)**: predire una parola dato il suo contesto.
- **Skip Gram (SG)**: predire le parole del contesto data una parola.

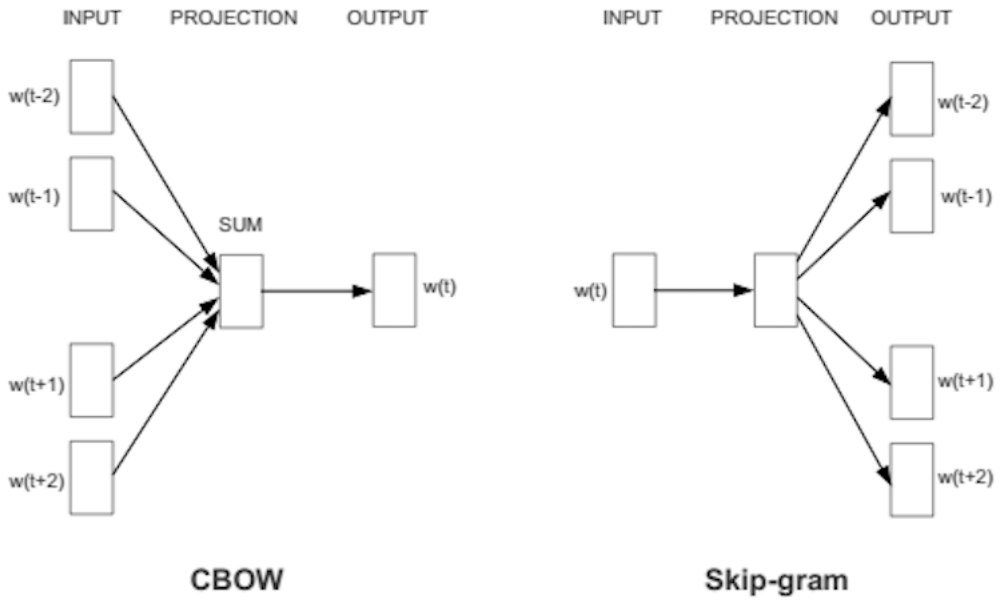

Fig. 2: Diagramma del funzionamento di Word2Vec nelle due modalità.

Costruendo dei cammini sul grafo utilizzando una metodologia di random walk [3] si crea un corpus di lunghezza arbitraria, utilizzato infine come input per l'algoritmo Word2vec. I vettori risultanti hanno proprietà simili agli embeddings Word2vec. In particolare, le entità simili sono più vicine nello spazio vettoriale di quelle dissimili, il che rende queste rappresentazioni ideali per l'apprendimento di modelli su tali entità.

CADE

Il classico modello Word2vec è un ottimo algoritmo per creare embedding, molto efficente ed in grado di creare rappresentazioni di qualità delle parole. Presenta però una limitazione: la creazione di due modelli distinti partendo da uno stesso testo in input produce embedding completamente differenti e tra loro non confrontabili. Ciò è causato dalla natura stocastica intrinseca del processo di training

delle reti neurali. Dunque non è possibile effettuare comparazioni, studiare differenze o similarità nel significato delle parole.
Compass Aligned Distributional Embeddings (CADE) [4] è un evoluzione di Word2vec che permette di allineare le embeddings generate da diversi corpora, abilitandone il confronto. Per prima cosa viene allenato un modello Word2vec in modalità CBOW utilizzando come input la concatenazione D dei D^n testi analizzati. Ciò produce due matrici di pesi C (parte sinistra del modello CBOW in Fig. 3) e U (parte destra del modello CBOW in Fig. 3), contenenti rispettivamente i *context* e *target embeddings* per D. Dunque per ogni specifica slice di testo D^i viene costruito un nuovo modello CBOW utilizzando la matrice *compass* U ed è allenato permettendo la modifica delle sole matrici C^i. In questo modo vengono adattati solo i contesti specifici per ogni slice, mantenendo allineato il target.

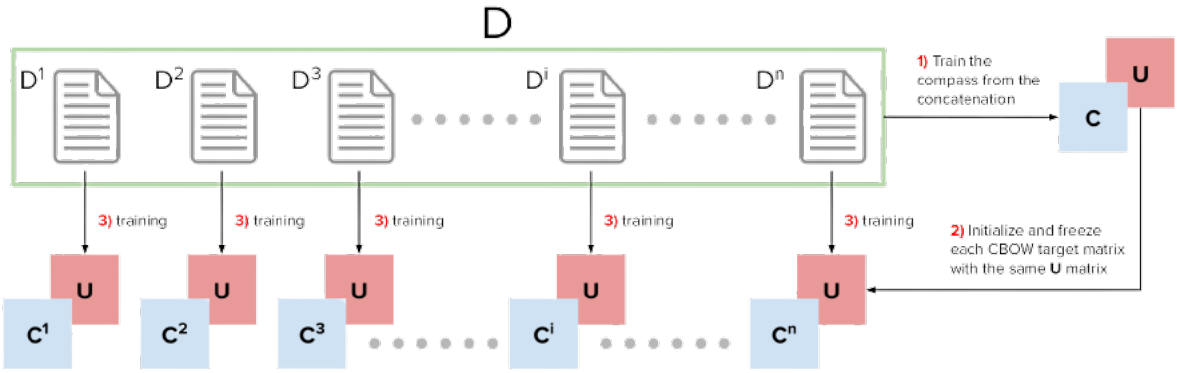

Fig. 3: Modello CADE [4].

Aligned Graph Embedding

Lo scopo del lavoro è quello di riuscire a confrontare, sotto diversi aspetti, due knowledge base diverse: DBpedia e Wikidata. Per fare ciò sono stati estratti cammini casuali da entrambe le knowledge base, attraverso random walk, al fine di ottenere dei file testuali. Questo output, propriamente preprocessatto per ottenere lo stesso vocabolario, viene utilizzato per costruire embeddings aligned tramite CADE.

Estrazione dei cammini

Durante questa operazione, vengono estratti 50 cammini con una profondità massima di 10, partendo sempre dall'entità "Esports" in entrambe le knowledge base. La strategia utilizzata è quella del random walk mediante l'algoritmo Depth First Search (DFS)(Fig.4). La strategia di ricerca esplora il grafo andando il più possibile in profondità: gli archi del grafo vengono esplorati a partire dall'ultimo vertice scoperto v che abbia ancora degli archi non esplorati uscenti da esso. Una volta terminata l'esplorazione di tutti gli archi non esplorati del vertice si ritorna indietro per esplorare tutti quelli uscenti a partire dal vertice da cui v era stato precedentemente scoperto. Il processo di esplorazione continua fino alla profondità specificata, in quanto il numero di figli per entità risulta molto elevato. Nel caso specifico, vengono analizzati anche gli antenati dell'entità scelta con un valore di profondità uguale alla precedente, in modo da ottenere cammini con l'entità di partenza nel mezzo.
A differenza di altre strategie, tipo Breth First Search (BFS)(Fig.4) che analizza prima i vertici a distanza 1, poi quelli a distanza 2 e così via, DFS permette ad ogni iterazione di ottenere cammini differenti tra loro, poichè man mano che si va in profondità il numero delle ramificazioni aumenta esponenzialmente.

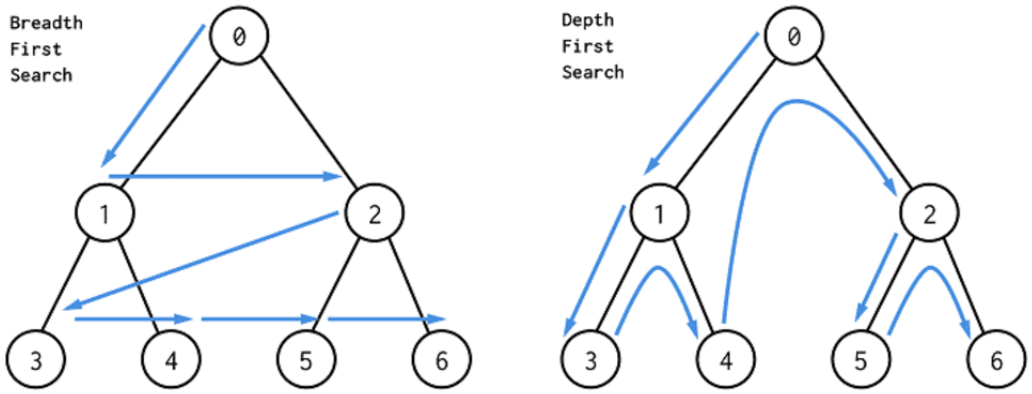

Fig. 4: Differenze tra *BFS* e *DFS*

La differenza sostanziale tra le estrazioni dei cammini è per un parametro di randomness. Mentre su Wikidata si estraggono cammini scegliendo randomicamente le foglie, su DBpedia invece questa scelta viene leggermente instradata. L'85% delle volte, partendo da un vertice v, la scelta della foglia viene fatta casualmente tra il gruppo di entità figlie di v che posseggono un sameAS link a Wikidata. Per il restante 15% delle volte la scelta è puramente casuale. Questo perchè, per ottenere overlap tra i vocabolari delle due knowledge base, è stato deciso di tradurre le entità di DBpeida in entità di Wikidata tramite sameAs link, quindi era necessario avere la sicurezza di poter fare questa traduzione nella maggior parte dei casi. Per il restante 15% si è mantenuta, se trovata, l'entità senza traduzione al fine di cercare di collegare entità non tradotte con quelle di Wikidata, individuando potenziali sameAs link.
Infine, dai cammini vengono eliminati tutti i predicati estratti, poichè la maggior parte di essi non posseggono traduzione (owl:equivalentProperty) verso Wikidata e vengono salvate le entità rimanenti su file di testo pronti per essere tradotti ed elaborati.

Preprocessing dei dati

Dopo aver estratto i cammini, nasce la necessità di elaborare i dati ottenuti per trasformarli in un formato compatibile e elaborabile con CADE. Le operazioni svolte sono quattro: "pulizia" del file testuale, traduzione delle entità da DBpedia a Wikidata mediante sameAs link, trattamento delle entità senza sameAs e creazione di un dizionario che mappa entità di Wikidata nelle rispettive label in linguaggio naturale.
Innanzitutto, le entità di DBpedia e Wikidata vengono estratte nei formati completi:

- DBpedia: *https://DBpedia.org/resource/**nome_Entità***
- Wikidata: *http://www.Wikidata.org/entity/**ID_univoco***

In fase di salvataggio su file di testo, i prefissi vengono rimossi e vengono mantenuti soltanto i nomi e gli ID, in maniera da semplificare e velocizzare l'elaborazione. Quando necessario, per esempio durante la compilazione delle query, verranno riaggiunte dinamicamente.
Dopo di che, vengono mappate le entità estratte da DBpedia in quelle di Wikidata utilizzando i sameAs link (Fig.5). Ad ognuna però possono corrispondere più link con Wikidata, spesso con totale incoerenza tra le due entità. Diventa necessario utilizzare una strategia di filtering per individuare il miglior candidato per il collegamento.

Neighborhood based filtering

Il processo di filtering avviene confrontando il vicinato dell'entità analizzata con quello delle entità collegate: quella che condivide il maggior numero di corrispondenze sarà scelta come sameAs link.

Sia E_d l'insieme di entità DBpedia estratte, $sa()$ una funzione che mappa un'entità ai suoi sameAs link e $nb()$ una funzione che mappa un'entità al suo vicinato nel knowledge graph relativo (es. $nb(e_d^j)$ mappa solo entità aventi un URI nella forma *https://DBpedia.org/resource/**). Per ogni entità DBpedia $e_d^j \in E_d$ esiste un insieme $sa(e_d^i) = \{e_{wd}^1, e_{wd}^2, ..., e_{wd}^i, ..., e_{wd}^n\}, 0 \leq i \leq n$ di entità Wikidata collegate. Siano infine $\{nb(e_{wd}^1), nb(e_{wd}^2), ..., nb(e_{wd}^i), ..., nb(e_{wd}^n)\}, 0 \leq i \leq n$ i vicinati delle entità Wikidata estratte e $sa(nb(e_d^j))$ le entità Wikidata collegate via sameAs link al vicinato di e_d^i. Il match viene dunque trovato come risultato della seguente funzione:

$$\max_i |sa(nb(e_d^j)) \cap nb(e_{wd}^i)|$$

Il processo viene ripetuto $\forall j \in E_d$ così da tradurre ogni entità. La funzione $nb()$ estrae tutto il vicinato senza distinzioni, rendendo il processo abbastanza esoso in termini di tempo di esecuzione e memoria. Le entità che non possegono sameAs vengono lasciate invariate(Fig.5), al fine di poterle elaborare e trovare un potenziale nuovo sameAs.

Infine, viene creato un file di testo unendo i dati tradotti da DBpedia e quelli estratti da Wikidata e vengolo eliminati i doppioni (Fig.5). Per ogni parola viene utilizzata una query a Wikidata che estrae le label in linguaggio naturale dall'ID e si crea un dizionario (chiave = "ID_univoco", valore = "label")(Fig.5), che verrà utilizzato per tradurre i risultati di CADE in un formato più comprensibile. Le entità non tradotte vengono mappate in se stesse.

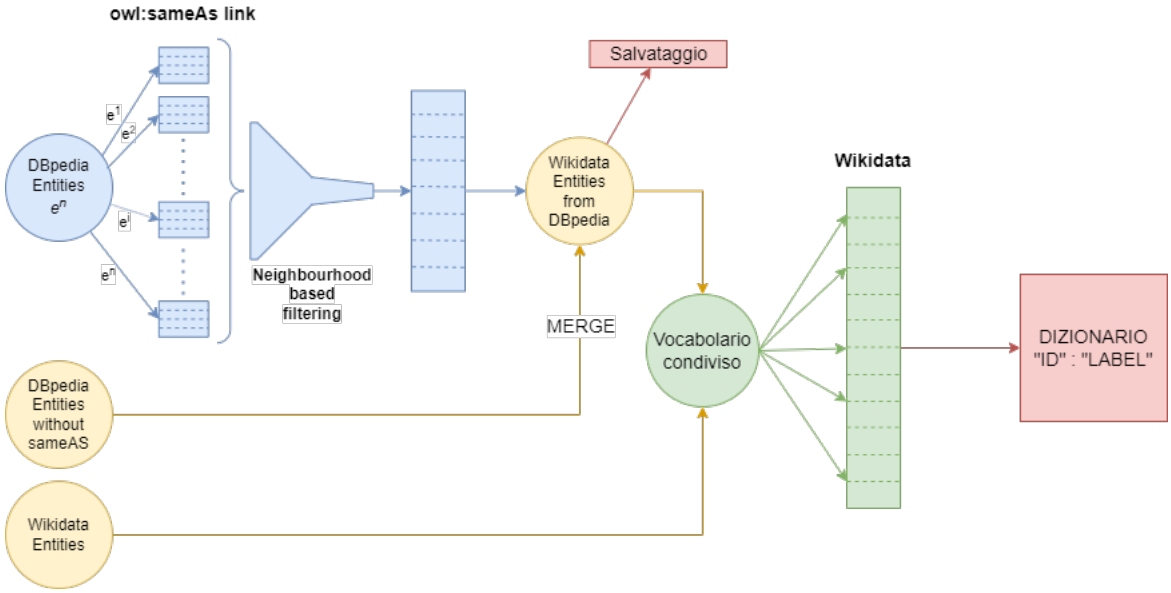

Fig. 5: Operazioni del preprocessing dei dati

Embedding alignment con CADE

Dopo aver preparato i dati, i file di testo vengono elaborati modellizzati attraverso CADE. Come primo passaggio viene creato un compasso, ovvero un file di testo che unisce quello di DBpedia tradotto e quello di Wikidata. Vengono contate, all'interno del nuovo file, le frequenze di comparizione delle singole entità nel formato "Q_numero" e separatamente quelle non tradotte da DBpedia. Vengono scelti dei valori soglia per eliminare le parole a frequenza minore e viene creata una lista, che verrà utilizzata dal modello word2vec di CADE per decidere quali entità modellizzare e quali scartare. Ottenuto così il modello sul compasso, vengono trainati i singoli modelli su i due testi. Dopo di che vengono analizzate singolarmente le parole non tradotte in maniera manuale,così da

filtrare le entità meno sensate. Vengono scelti i vocaboli da tenere in relazione alla sensatezza con l'entità di partenza Esport e alla qualità delle parole più simili con il modello estratto da Wikidata. Le parole scelte vengono aggiunte a un nuovo vocabolario contenente solo parole tradotte con la frequenza scelta precedentemente. A questo punto si riaddestra CADE e i due modelli.
In questo modo viene a crearsi uno spazio vettoriale in cui esistono e sono allineate le entità scelte per frequenza sotto forma di vettori. Questi vettori verranno utilizzati in seguito per cercare similitudini tra entità, creare un nuovo grafo che unisce le due kb e per cercare similarità tra le entità non tradotte e quelle di Wikidata.

Risultati

Per comprendere bene i risultati è importante specificare che tutti i valori di similarità che vengono estratti non indicano similitudine tra le parole in linguaggio naturale, poichè i testi utilizzati sono liste di collegamenti tra entità e non parole connesse con una vera e propria grammatica. La similarità indica più la quantità e la distanza di collegamenti estratti tra due entità, oppure se considerate intra knowledge base, la similitudine dell'intorno delle due entità. Per esempio se "Esports" e "Gaming" hanno cammini nelle knowledge base di distanza 1 o 2 e nei testi queste parole compaiono spesso vicine, avranno un valore di similarità alto. Differentemente se due entità sono molto "distanti" e i loro collegamenti compaiono poco spesso, allora la similarità sarà bassa. Parlando intra knowledge base, se "Esports" da DBpedia e "Gaming" da Wikidata hanno indice di similarità alto, allora probabilmente gli intorni delle due entità si somiglieranno per tipologia di entità.
Infine, gli obiettivi prefissati sono l'analisi della qualità dell'embedding, la creazione di un nuovo grafo, che possa collegare le entità estratte dalle due knowledge base mediante valori di similarità e l'analisi delle entità senza sameAs link con quelle di Wikidata.

Qualità dell'embedding

Come controllo della qualità dell'embedding di CADE si è deciso di ricercare alcune entità per frequenza maggiore all'interno dei cammini estratti. Dopo di che sono state ricercate quelle più simili incrociando i modelli. Le tabelle seguenti mostrano la media della similarità estratta tra ogni entità e le 5 più simili.

DBpedia	**Top 5 most similar in Wikidata**	**Similarity mean**
esports	'esports commentator', 'gamer', 'professional gamer', 'esports', 'competitive player'	0.8271
double-elimination tournament	'manufactured good', 'job', 'professional', 'video game design', 'audiovisual work'	0.6431
internet café	'esports commentator', 'audiovisual work', 'professional gamer', 'esports', 'Gender Glossar'	0.8390
augmented reality	'physical fitness', 'physical exercise', 'Template:Infobox sport overview', 'sports equipment', 'audiovisual work'	0.6507
occupational burnout	'German', 'Gender Glossar', 'esports team', 'esports', 'manufactured good'	0.6907

Wikidata	Top 5 most similar in DBpedia	Similarity mean
esports	'U.S. National Video Game Team', 'Category:Video_game_culture', 'addiction', 'internet café', 'electronic sports at the 2018 Asian Games'	0.8457
professional gamer	'U.S. National Video Game Team', 'electronic sports at the 2018 Asian Games', 'Category:Video_game_culture', internet café', 'esports'	0.8595
type of sport	'Category:Video_game_culture', 'Greg Fields', 'electronic sports at the 2018 Asian Games', 'addiction', 'internet café'	0.8047
esports commentator	'U.S. National Video Game Team', 'Category:Video_game_culture', 'internet café', 'esports', 'Category:South_Korean_inventions'	0.8574
video game	'electronic sports at the 2018 Asian Games', 'U.S. National Video Game Team', 'Category:2018_in_esports', 'Category:Video_game_culture', 'internet café'	0.7472

Si può notare che i valori di media sono abbastanza alti, migliori però per Wikidata. Questo perchè, data la struttura interna delle due knowledge base, DBpedia estrae molte entità diverse in bassa quantità(ordine di grandezza 10^2), mentre Wikidata estrae meno entità con una frequenza maggiore (ordine di grandezza 10^3). Questo porta una modellizzazione migliore per i vocaboli di Wikidata e quindi a una miglior precisione sulla similarità.
Nel complesso, comunque, si può dire che l'embedding ha ottenuto una buona qualità, almeno per le entità vicine spazialmente all'entità di partenza.

Fusione dei grafi

Per la creazione del nuovo grafo, si parte da una entità presente in entrambi i vocabolari: "Esports" e da un numero di iterazioni massimo. La creazione si fermerà quando il numero di itezione raggiungerà zero. Ad ogni aggiunta di vertice o arco ne verrà prima controllata l'esistenza, in maniera da non avere doppioni. Si aggiunge il vertice V al grafo e mediante CADE si estraggono le 3 entità più simili al vertice di partenza, con rispettivi valori di similarità, da entrambi i modelli, per un totale di 6 entità. Per ogni vertice V' estratto ci sono 3 possibilità con altrettante conseguenze:

- **V' appartiene a entrambi i vocabolari**: in questo caso viene aggiunto il vertice, viene aggiunto l'arco (V, V') con peso uguale a valore di similarità e si applica ricorsione con valore di iterazione-1
- **V' appartiene solo al vocabolari di DBpedia**: in questo caso viene aggiunto il vertice con label="DBpedia" o "DBpedia_no_sameas" se appartiene alla categoria dei non tradotti. Si aggiunge l'arco (V, V') con peso pari a valore di similarità. Non si applica ricorsione ma viene lanciata una funzione che crea 3 foglie da V'.
- **V' appartiene solo al vocabolari di Wikidata**: questo caso è identico al precedente, solo che il vertice viene aggiunto con label = "wikidata".

Le funzioni "foglia" si occupano di cercare da V' i più simili V" nel modello da cui non provengono, da DBpedia se V' appartiene a Wikidata e viceversa. Fatto ciò per ogni entità estratta viene aggiunto il nodo V", con rispettiva label in base alla provenienza di V" (both, wikidata, dbpedia o dbpedia_no_sameas), e l'arco (V', V") con peso uguale alla similarità. Ciò che si ottiene è un grafo diretto pesato con vertici etichettati in base alla provenienza dell'entità.

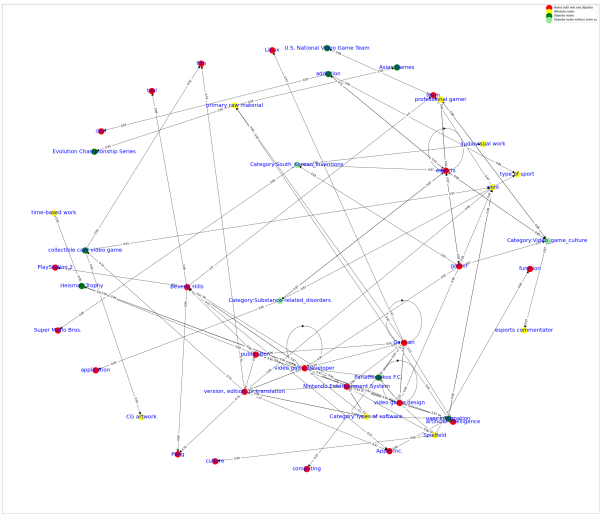

Fig. 6: Piccolo estratto di un grafo ottenuto con 4 iterazioni

Come si può già notare, solo dopo quattro iterazioni il grafo presenta molteplici connessioni tra nodi di diverso tipo. I rossi, quelli più presenti, sono i nodi appartenenti a entrambi i vocabolari. Essi a volte posseggono anche dei cappi, questo perchè la stessa entità nei due vocabolari ha ottenuto un alto valore di similarità. Questo ci indica anche che le entità con cappi sono quelle più presenti all'interno dei file di testo. Le entità gialle (Wikidata) e verdi scuro (DBpedia), sono meno presenti, ma ci permettono di individuare collegamenti tra le due knowledge base. Le più interessanti sono le entità verde chiaro, ovvero quelle non tradotte da dbpedia, che attraverso i loro collegamenti ci possono indicare nuovi sameAs link.

Analisi delle entità senza sameAs link

In particolare si è deciso di analizzare singolarmente le entità senza sameAs link, allo scopo di trovare potenziali nuovi collegamenti. Il grafico sottostante (fig7) mostra le entità di DBpedia senza sameAs link e le prime tre più simili trovate sul modello di Wikidata. Si può notare che vengono trovati alcuni collegamenti significativi:

- Category:Video_game_culture con esports
- Category:Video_game_culture con esports commentor
- Category:Super_Smash_Bros con esports
- Teemu Jasskelainen (Hockeista) con physical exercise

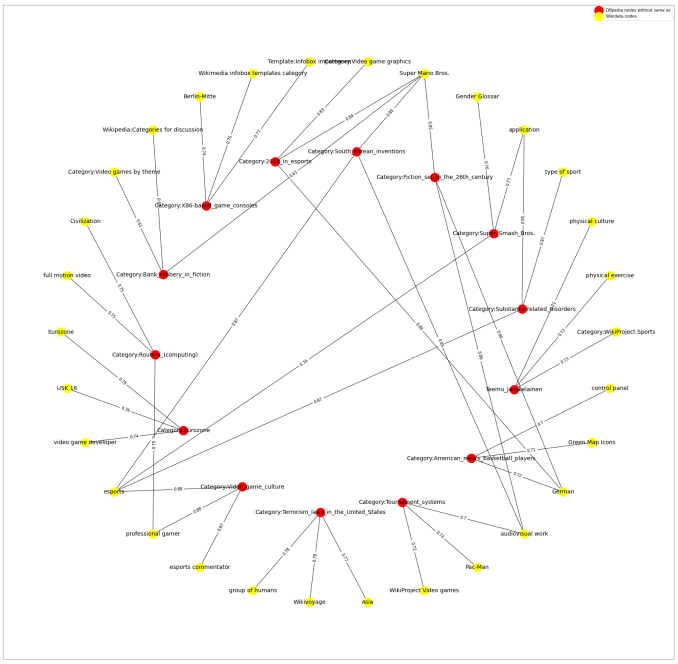

Fig. 7: Collegamenti delle entità senza sameAs con quelle di wikidata

Questi collegamenti, adeguatamente filtrati, potrebbero essere nuovi potenziali sameAs link. Interessante risulta compiere, anche, l'azione contraria, ovvero partire dalle entità modellizzate da Wikidata e cercare la similarità con le parole di DBpedia senza sameAs. In questo caso viene estratta, da ogni entità di Wikidata, una pool di 5 elementi più simili dal modello di DBpedia e ne vengono cercati i vocaboli senza sameAs link. Alcuni risultati significativi e alcuni senza senso vengono mostrati nella tabella seguente:

DBpedia	Wikidata	Similarity
Category:2018_in_esports	sportsperson	0,763468325
	Category:Esports	0,756392181
	sport	0,736316979
	Wikimedia category (disambiguation)	0,731792688
	Portal:Sports	0,731577635
Category:Eurozone	Wikidata value	0,617542267
	Wikibase data model element	0,613431931
	abstract object	0,598986804
	value	0,548606575
Category:Fiction_set_in_the_26th_century	Category:Video game	0,775799572
	video game console	0,765537024
	Category:Esports	0,765254378
	Template:Infobox esports team	0,763017774
	Category:Video games	0,751680851
Category:Routers_(computing)	Category:Professions	0,727530658
	sports commentator	0,707353532
	Wikipedia:Vital articles	0,661730766
	toy	0,633785307
	specialty	0,591178238
Category:South_Korean_inventions	esports	0,865026951
	esports commentator	0,844506383
	professional gamer	0,838164508

Table 3 continued from previous page

DBpedia	Wikidata	Similarity
	application	0,834588528
	gamer	0,832287312
Category:Substance-related_disorders	esports	0,872202456
	type of sport	0,866340339
	application	0,845249951
	esports commentator	0,806651652
	second-order class	0,782135606
Category:Super_Smash_Bros.	esports team	0,743600249
	Category:Sports by type	0,681823254
	Category:Esports teams	0,643220246
	video game term	0,641958535
	Category:Science	0,635872185
Category:Terrorism_laws_in_the_United_States	Category:MediaWiki websites	0,759788394
	Wikiversity	0,746234298
	community project	0,689347446
	dot-com company	0,666140258
	Internet	0,665249228
Category:Tournament_systems	Doom	0,685964942
	Template:Infobox Galgame	0,676052749
	Portal:Video games	0,666646361
	Q26210085	0,646048784
	Humble Store	0,579195082
Category:Video_game_culture	esports	0,88309449
	professional gamer	0,878826737
	esports commentator	0,873822629
	gamer	0,868131876
	type of sport	0,82224685
Category:X86-based_game_consoles	Wikimedia infobox templates category	0,746209025
	human-readable medium	0,693867624
	Infobox	0,689828217
	Category:WikiProject Infoboxes	0,689355075
	visualization	0,688968003
Teemu_Jaaskelainen	sports organization	0,630729437

Si può notare che anche questo processo estrae potenziali sameAs link, ma anche collegamenti senza senso, basti vedere *Category:Routers_(computing) con toy*. Questo fa capire quanto un azione del genere può essere utile per creare nuovi collegamenti, ma altrettanto "distruttiva" se non adeguatamente controllata e filtrata.

Conclusioni

Concludendo si è riusciti a creare un nuovo grafo, esso collega le entità estratte dalle due knowledge base attraverso random walk utilizzando indici di similarità estratti dall'embedding di CADE. Si è riusciti a dare senso a entità senza sameAS link, trovando dei vocabili di Wikidata che potrebbero ben collegarsi o comunque fornire un punto di partenza per il sameAs link più compatibili. Oltre ai risultati sono emerse anche alcune problematiche. Il principale problema è la grande differenza del numero di volte che un'entità compare nei cammini: più è vicina all'entità di partenza maggiore sarà la sua frequenza nel corpus finale. Una soluzione sarebbe applicare random walk su un insieme non fissato per il random walk e con profondità di ricerca minore, eseguendo nuove iterazioni dell'algoritmo su quelle entità che compaiono con meno frequenza, in modo da ottenere un testo più uniforme.
Oppure si potrebbero sfruttare altre strategie di walk: Hierarchical random walk (HALK) [3], ad

esempio, rimuove le entità rare durante l'esplorazione in quanto spesso portano poche informazioni, aumentando la probabilità di incontrare entità più interessanti e diminuendo al contempo l'utilizzo della memoria.
Esistono inoltre diverse varianti di RDF2vec, molto interessanti in questo caso d'uso:

- Node2vec [5], definito per grafi con un solo tipo di archi, apprende un nuovo mapping dei nodi che massimizza la probabilità di conservarne il vicinato.
- KG2vec [6]: per le reti la tripla (entità di testa, relazione e entità di coda) è la chiave per costruire il contesto del nodo. La costruzione dei corpora effettuata nel seguente studio trascura questa informazione chiave, per cui la qualità dell'embedding è compromessa. Per risolvere tale problema KG2vec astrae le relazioni creando dei nuovi nodi ed adatta random walk, differenziando nella ricerca nodi relation-type e entity-type.

References

[1] Petar Ristoski and Heiko Paulheim. "RDF2Vec: RDF Graph Embeddings for Data Mining". In: *SEMWEB*. 2016.

[2] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[3] Gilles Vandewiele et al. *Walk Extraction Strategies for Node Embeddings with RDF2Vec in Knowledge Graphs*. 2020. arXiv: 2009.04404 [cs.LG].

[4] Federico Bianchi et al. *Compass-aligned Distributional Embeddings for Studying Semantic Differences across Corpora*. 2020. arXiv: 2004.06519 [cs.AI].

[5] Aditya Grover and Jure Leskovec. *node2vec: Scalable Feature Learning for Networks*. 2016. arXiv: 1607.00653 [cs.SI].

[6] *Expeditious Generation of Knowledge Graph Embeddings*. 2018.