
**UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO**

DIPARTIMENTO DI INFORMATICA

Corso di Laurea Triennale in
**INFORMATICA**

---

Tesi di laurea in
Ingegneria della Conoscenza

# Embedding Model per Knowledge Graph: Classificazione di Triple ed Estrazione di Conoscenza

**Relatore:**
Prof. Nicola Fanizzi

**Laureando:**
Matteo Balice

**Correlatore:**
Prof.ssa Claudia d'Amato

**Anno accademico:**
2022/2023

# Sommario

Il web semantico è un'estensione del *World Wide Web* in un ambiente in cui le risorse al suo interno sono associate a informazioni e dati che ne specificano il contesto semantico in un formato adatto al ragionamento automatico da parte delle macchine.
La struttura che viene utilizzata per rappresentare graficamente il *semantic web* è il **Knowledge Graph**, una struttura basata su grafo in grado di rappresentare la conoscenza: i nodi rappresentano gli individui (*entità*, *oggetti*, *concetti*...) mentre gli archi rappresentano le relazioni che legano i vari individui.
La chiave del successo di questa rappresentazione è da ritrovare nella loro flessibilità, in quanto consentono di collegare risorse tra loro di natura eterogenea. D'altro canto, però, uno dei principali problemi del semantic web è l'*incompletezza dei dati*: a causa anche dell'enorme mole di dati che bisogna gestire nei Knowledge Graph è impossibile disporre a priori di tutta la conoscenza. Una soluzione a questo problema è il *link prediction*, cioè predire le relazioni mancanti all'interno dei Knowledge Graph (e *triple classification* che invece predice la verità di date triple). A questo scopo, sono stati proposti molti modelli, utilizzando anche tecniche molto differenti tra di loro. Tuttavia, molti di questi modelli non riescono a fornire una spiegazione del risultato ottenuto dalla predizione, il che potrebbe rivelarsi un requisito fondamentale in alcuni domini (come in ambito medico). Obiettivo della tesi è quindi quello di risolvere il problema dell'incompletezza dei Knowledge Graph tramite *triple classification* analizzando e confrontando le performance di diversi modelli e valutarne la loro interpretabilità (*estrazione di conoscenza*). Nello specifico verranno considerati due modelli orientati alle classi: TransC-OWL, basato su TransC a cui si è iniettato conoscenza di fondo e TransM.

# Indice

**Sommario**

# Capitolo 1

# Introduzione

## 1.1 Contesto

La nozione di *Knowledge Graph* (nel seguito KG) è stata introdotta in letteratura sin dagli anni '70 [Sch73], mentre i primi tentativi di impiego nel campo dell'Intelligenza Artificiale risalgono al 1980 [Hog+20]. L'idea alla base dei KG è quella di utilizzare grafi per rappresentare i dati e le informazioni. Un'astrazione della conoscenza basata sulla struttura di grafo presenta diversi benefici aggiuntivi rispetto ad altri modelli quali il classico modello relazionale o i più recenti modelli adottati nel contesto NoSQL [TS22]. Essi forniscono una rappresentazione intuitiva in numerosi domini e gli archi riescono a catturare le relazioni tra entità anche di natura molto diversa tra loro. I grafi permettono di posticipare la definizione dello schema in modo tale da consentire direttamente ai dati di evolversi nella maniera più flessibile e dinamica possibile, proprietà indispensabile in un ambito come quello del *Web Semantico* nel quale la conoscenza viene trattata senza assunzioni di completezza, ossia si ragiona facendo l'*assunzione di mondo aperto* (OWA) [MP15] come in Logica.

I KG possono inoltre essere impiegati sia per definire la semantica delle informazioni ivi rappresentate mediante i formalismi utilizzati per la rappresentazione della conoscenza, come ad esempio le ontologie e sia per effettuare processi di ragionamento sulla semantica associata agli elementi contenuti nel KG [Hog+20]. Nell'ambito del Web Semantico, i KG vengono descritti mediante l'utilizzo di linguaggi formali, ciascuno dei quali può essere interpretato come un differente sottoinsieme delle

capacità espressive delle *Logiche Descrittive* [Baa+03], tra i più noti vi sono RDF, RDF-Schema e OWL [Ehi+20].

In particolare, il linguaggio RDF (Resource Description Framework) ha giocato un ruolo fondamentale nell'ambito dei cosiddetti *Linked Data*. Con questo termine si designa un insieme di principi e norme che mirano a rendere i dati strutturati disponibili e collegati sul Web [BHB09]. Nel 2006, Tim Berners-Lee, padre del World Wide Web, ha delineato i requisiti necessari affinché i dati pubblicati sul Web, utilizzando proprio RDF, possano diventare parte integrante di una vasta e unificata struttura dati globale, a cui ha dato il nome di *Web of Data*. Questa iniziativa è tesa verso l'obiettivo di rendere i dati accessibili e utilizzabili non solo da parte degli esseri umani, ma soprattutto da parte delle macchine, favorendo la condivisione globale della conoscenza puntando a un'interoperabilità di livello superiore, quello semantico.

Il termine Linked Open Data (LOD) viene utilizzato quando i dati sono pubblicati seguendo gli standard e le licenze aperte, permettendo così a chiunque di utilizzarli, modificarli e condividerli senza restrizioni. I KG, strutturati in modo semantico e basati su RDF, costituiscono una delle realizzazioni di questa visione. Essi rappresentano una vasta collezione di conoscenze interconnesse, che spaziano dai dati scientifici alle descrizioni di entità del mondo reale. La *Linked Open Data cloud* (Fig. 1.1) è una rete di KG interconnessi che abbraccia una vasta gamma di domini e settori. Questa enorme struttura dati continua a crescere costantemente, con nuovi dati che vengono pubblicati e collegati regolarmente.

## 1.2 Motivazioni

L'enorme mole di informazioni prodotte nell'ambito del Web Semantico e non solo, si pensi ad esempio ai Linked Open Data, risulta molto complessa da gestire a causa del volume dei dati e del rumore, dovuto alla presenza di informazioni errate rispetto al dominio di riferimento ma non riscontrabili come tali da meccanismi di ragionamento automatico

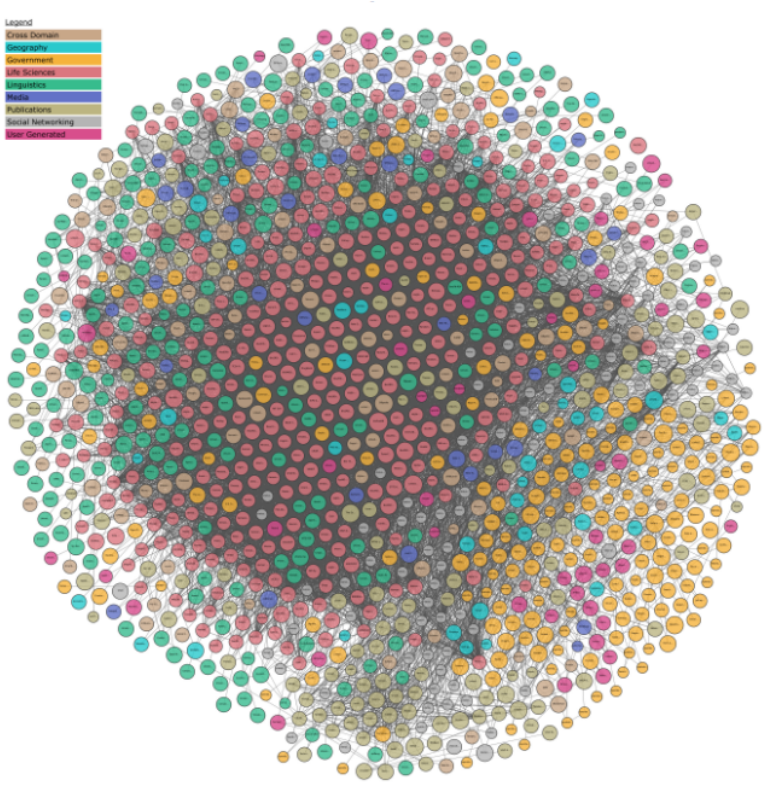



Figura 1.1: Rappresentazione grafica della Linked Open Data Cloud (anno 2023)

e dell'inerente incompletezza, dovuta alla mancanza di informazioni per una porzione significativa del KG.

L'attività volta a correggere varie problematiche associate ai KG (KG) è comunemente nota con il termine *Knowledge Graph refinement*. I metodi impiegati in questa pratica si differenziano in base agli obiettivi che perseguono, su come migliorano la completezza e/o la correttezza del KG, riducendo il rumore presente. Tra queste strategie, molte si concentrano sulle relazioni tra le entità tra cui la validazione e l'aggiornamento delle stesse. Ciò può comportare la rimozione di relazioni errate, l'aggiunta di relazioni mancanti o la modifica delle relazioni esistenti per renderle più accurate e coerenti. Un'altra tecnica consiste nella gestione dei tipi di entità, che comprende l'assegnazione di tipi alle entità che non hanno una classificazione, la correzione dei tipi errati e la scoperta di nuovi tipi di entità basandosi sui dati disponibili. Infine, un altro fattore rilevante riguarda la tipologia dei dati utilizzati, che possono provenire da fonti esterne o essere estratti direttamente dal KG stesso.

Per riuscire in tali compiti, alcune strategie si avvalgono dei servizi

di ragionamento che è possibile utilizzare nei KG, mentre altre si avvalgono della possibilità di addestrare modelli sul KG stesso. Recenti lavori [San19; Qua19; dQF21] hanno provato l'efficacia della combinazione delle due strategie appena citate, adoperando l'iniezione di conoscenza di fondo per incrementare le prestazioni di modelli classici quali TransE [Bor+13] e TransR [Lin+15].

In questo lavoro si cercherà di adattare metodi basati su una maggiore complessità di rappresentazione delle classi, come TransC [Lv+18] e metodi basati sull'iniezione di conoscenza, come TransOWL a modelli allo stato dell'arte, successivi a TransE e più performanti.

L'obiettivo è quello di verificare, con tali espansioni, in quale misura è possibile incrementare le prestazioni rispetto ai task di knowledge graph refinement, nello specifico di estrazione di conoscenza, che riguarda l'estrazione di informazioni o conoscenza da un KG, cioè identificare e estrarre nuove informazioni o relazioni che possono essere aggiunte al KG esistente (ad esempio sfruttando la geometria del modello per ricavare nuove regole logiche), e di Triple Classification, che invece riguarda la classificazione delle triple di conoscenza come vere o false. Un esempio potrebbe essere "*Tom è il padre di Jerry*" che viene classificata come *vera*, mentre "*Jerry è il padre di Tom*" dev'essere necessariamente *falsa*. La Triple Classification implica l'analisi delle triple per determinare se sono accurate o meno. L'obiettivo è classificare correttamente ogni tripla come vera (cioè corrispondente alla realtà) o falsa (cioè errata o non supportata dalle evidenze).

L'auspicio è, dunque, quello di dare prova della validità dei metodi per l'espansione dei modelli, affinché possano essere utilizzati per migliorare le prestazioni dei futuri modelli allo stato dell'arte.

## 1.3 Organizzazione della Tesi

Il presente lavoro è strutturato come segue:

Nel capitolo 2 in cui verranno introdotti i principali linguaggi utilizzati nell'ambito del Web Semantico e di conseguenza quindi nei KG.

Nel capitolo 3 verranno enunciati i principali modelli dalla quale prendono spunto TransC-OWL e TransM. Verranno esaminati alcuni dei principali lavori presenti in letteratura che hanno fornito l'ispirazione e il contesto per lo sviluppo dei modelli proposti in questa tesi.

Nel capitolo 4 delle estensioni proposte, verranno presentati i due modelli costruiti nel contesto di questa tesi. Saranno illustrate le architetture dei modelli, funzioni di score, di loss e ottimizzatori. Verranno inoltre fornite, ove necessario, basi matematiche per poter comprendere le scelte di progettazione del modello.

Nel capitolo 5 della valutazione sperimentale, verranno esposti i risultati ottenuti dai modelli proposti nel task di Triple Classification ed estrazione di regole. Verranno discussi i punteggi di accuracy, F1, precision e recall ottenuti dai modelli sui diversi dataset di valutazione utilizzati. Saranno confrontati i risultati con quelli di altri approcci presenti in letteratura per valutare l'efficacia e l'efficienza dei modelli proposti.

Nel capitolo finale verranno tratte le conclusioni di questo lavoro di tesi, riepilogando i risultati ottenuti nei modelli TransC-OWL e TransM.

# Capitolo 2

# Linguaggi nel Semantic Web

## 2.1 RDF/RDFS

Le informazioni contenute nel **Semantic Web** espresse mediante metadati necessitano di un'organizzazione strutturata e standardizzata per poter essere scambiate tra i sistemi e gli applicativi. **RDF** [Ehi+20] (*Resource Description Language*[1]) è il linguaggio preposto alla codifica di tali metadati. RDF organizza la conoscenza mediante documenti composti da insiemi di **triple**, che costituiscono l'*unità elementare* di rappresentazione dell'informazione. Le triple esprimono **relazioni** tra entità e condividono tutte la medesima struttura composta da *(soggetto, predicato, oggetto)*. Ad esempio, la frase "*Alan Turing è nato a Londra*" può essere codificata sotto forma di tripla come (`AlanTuring, natoA, Londra`), dove `Alan Turing` è il soggetto, `natoA` è il predicato e `Londra` è l'oggetto.

Per consentire l'*interoperabilità semantica* tra gli applicativi, è necessario identificare in maniera univoca ciascuna di queste entità, chiamate **risorse**, utilizzando gli **URI** (*Uniform Resource Identifiers*[2]). Inoltre, all'interno di una tripla è possibile utilizzare anche **valori associati a dati** come numeri, date, valori di verità e stringhe, chiamati **letterali** (*literals*), la cui interpretazione è data dal *datatype*, ovvero la classe di dati a cui fanno parte. Per rappresentare conoscenza esprimibile con strutture più complesse, come ad esempio "*Alan Turing ha dimostrato*

[1] `https://www.w3.org/RDF/`

[2] `https://www.w3.org/wiki/URI`

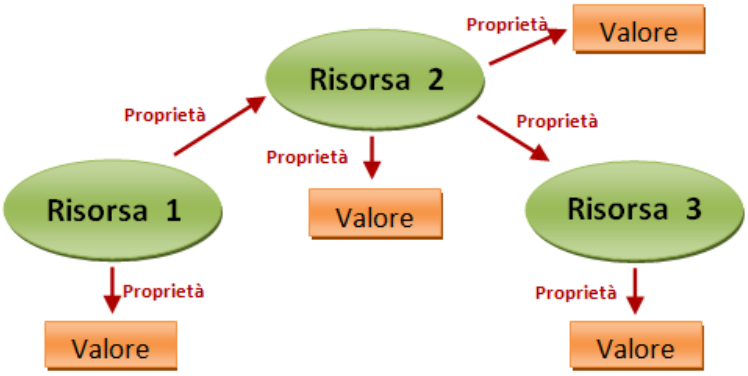



Figura 2.1: Struttura a grafo tipica di RDF e RDFS

*che la Macchina di Turing non risolve il problema dell'arresto*", è possibile utilizzare i **blank node**[3]: *nodi ausiliari*, sostituibili al soggetto o all'oggetto, impiegabili per rappresentare e riferirsi a ulteriori risorse. I documenti RDF, che sono costituiti da un insieme di triple, possono essere interpretati come **grafi orientati**, spiegando così la struttura a grafo (figura 2.1) dei **knowledge graph** e la loro capacità di rappresentare la conoscenza nel Web Semantico.

Tuttavia, l'*espressività di RDF* è limitata. Per rappresentare significati semantici più precisi, come la distinzione tra individui e classi, è necessario un linguaggio più evoluto chiamato **RDF-Schema** (*Resource Description Language Schema*[4], o RDFS) [Ehi+20]. L'importante novità introdotta da RDFS è la possibilità di definire dei **vocabolari**, che consentono di esprimere ciascuna risorsa in funzione di un'altra. Ad esempio, è possibile definire `Alan Turing` come un `Matematico`. Questa caratteristica, che consente di descrivere le relazioni semantiche in un dominio di interesse, rende RDFS un linguaggio adatto alla rappresentazione della conoscenza, anche definito *ontology language*.

Considerando la frase "*i matematici sono scienziati*", vogliamo implicitamente intendere che la classe dei `Matematici`, ovvero l'insieme di individui che studiano matematica, è inclusa nella classe degli scienzia-

[3] `https://www.w3.org/wiki/BlankNodes`

[4] `https://www.w3.org/TR/rdf-schema/`

ti, ovvero l'insieme di individui che studiano scienze. Le nuove proprietà di RDFS consentono di organizzare le classi e le relazioni in **strutture gerarchiche**, poiché le classi e le relazioni possono essere interpretate come insiemi di cui è possibile considerarne i relativi sottoinsiemi, esprimendo quindi *sottoclassi* e *sottorelazioni*. È inoltre possibile precisare le classi a cui devono appartenere le risorse soggetto e oggetto di una relazione mediante i rispettivi costrutti di `rdfs:domain` e `rdfs:range`. Ad esempio, nella frase "*Alan Turing è nato a Londra*", ci aspettiamo che il soggetto sia sempre un essere umano e l'oggetto un luogo geografico.

Tuttavia, RDFS non è sufficientemente espressivo per rappresentare forme di conoscenza più articolate. Sarà quindi necessario utilizzare un linguaggio diverso, come **OWL**.

## 2.2 OWL

OWL[5] (Web Ontology Language) [Ehi+20], sviluppato a partire dal 2004 dal W3C Web Ontology Working Group, è un linguaggio per esprimere conoscenza nel Semantic Web con una finalità specifica: costituire un compromesso tra l'espressività del linguaggio e l'efficienza in termini computazionali dei processi di inferenza. A partire dal 2009, è stata rilasciata una variante più espressiva di OWL, denominata OWL 2 (figura 2.2), che però non è necessario considerare in questo lavoro di tesi. Esistono diverse varianti, definite profili, del linguaggio OWL, denominate OWL Species [HKR10]:

- **OWL Full:** è la variante più espressiva, che generalmente può risultare indecidibile. L'incertezza è causata, in parte, dalla mancanza del vincolo di separazione dei tipi (*type separation*), il che determina che individui, classi e predicati possano essere indicati dallo stesso identificativo in proposizioni o statement differenti [HKR10].
- **OWL DL:** è la variante caratterizzata dalla decidibilità in termini computazionali, ovvero dato un problema di inferenza esiste sempre un algoritmo in grado di terminare. Per ottenere questa proprietà è necessario imporre delle restrizioni quali: la separazione e la dichiarazione dei tipi, restrizioni sulle relazioni concrete (o *concrete roles*, relazioni tra individui e tipi di dati) e sulle relazioni astratte (o *abstract roles*, relazioni tra individui ed individui) [HKR10].
- **OWL Lite:** inizialmente concepita come la variante più semplice, si è dimostrata equivalente in termini di complessità rispetto a OWL DL; viene quindi scarsamente presa in considerazione [HKR10].

In questo lavoro di tesi è stata considerata la variante **OWL DL**. Il linguaggio OWL offre diversi costrutti per esprimere proprietà tra individui, classi e relazioni. Prendiamo ad esempio la frase "le persone non sono libri". Con questa affermazione intendiamo esprimere che ogni indivi-

[5] `https://www.w3.org/OWL/`

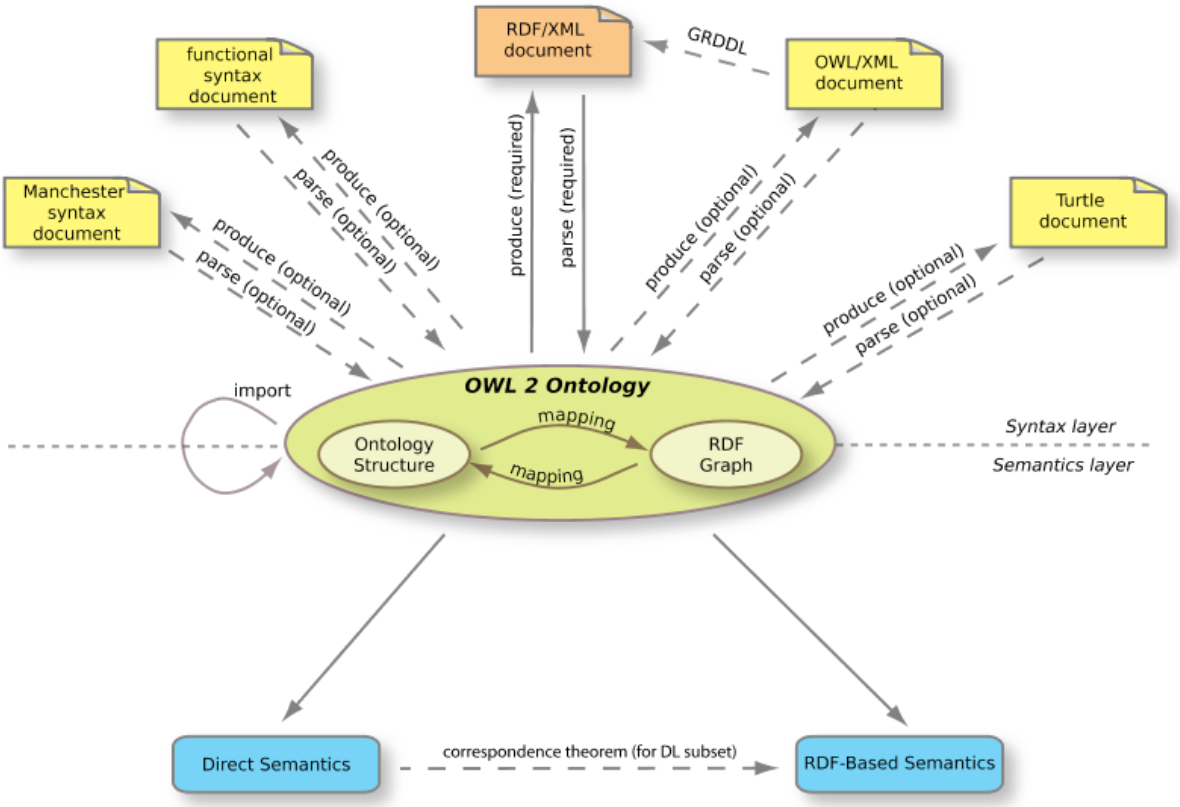



Figura 2.2: Struttura di OWL 2

duo definito come persona non può essere anche un oggetto inanimato come un libro. OWL ci consente di rappresentare questa informazione utilizzando il costrutto `owl:disjointWith`, che specifica che due classi non possono avere individui in comune. Possiamo quindi inferire, ad esempio, che "Alan Turing non è un libro". OWL consente anche di specificare ulteriori relazioni tra classi utilizzando i costrutti `owl:intersectionOf`, `owl:unionOf`, `owl:complementOf` e `owl:equivalentClass`. Ad esempio, con il costrutto `owl:intersectionOf` possiamo precisare che la classe `Donna` è costituita da individui che appartengono sia alla classe `Femmina` che alla classe `Persona`.

OWL consente anche di specificare proprietà particolari relative alle relazioni, chiamate predicati, che sono tutte istanze della classe `rdf:Property` e definiscono associazioni tra risorse soggetto e risorse oggetto nella forma di relazioni binarie. Oltre alla proprietà `rdfs:subProperty`, menzionata in precedenza, è possibile specificare altre proprietà utili come `owl:FunctionalProperty`, `owl:InverseOf`. La prima specifica che la relazione in questione, ad esempio `marito_di`,

si comporta come una funzione, cioè ad ogni risorsa soggetto è associata una sola risorsa oggetto. La seconda specifica che date due relazioni, ad esempio `figlio_di` ed `genitore_di`, sono l'inversa l'una dell'altra, il che significa che se c'è una relazione tra l'oggetto $x$ e l'oggetto $y$ secondo il predicato `figlio_di`, allora esiste una relazione tra l'oggetto y come soggetto e l'oggetto $x$ come oggetto secondo la relazione `genitore_di`. Due relazioni possono anche essere definite equivalenti mediante il costrutto `owl:equivalentProperty`, il che significa che mettono in relazione le stesse risorse in termini di estensione della proprietà, cioè gli insiemi di istanze della relazione definite come coppie (soggetto, oggetto). È anche possibile specificare che due risorse si riferiscono allo stesso oggetto utilizzando il costrutto `owl:sameAs`, che permette di collegare tra loro i KG e le informazioni che contengono, ad esempio nel contesto dei Linked Open Data.

Il linguaggio OWL si basa su una famiglia di linguaggi formali chiamati logiche descrittive (DL) per la rappresentazione della conoscenza. Le DL sono caratterizzate da diversi livelli di espressività e costituiscono un frammento della logica del primo ordine. I Knowledge Graph permettono di rappresentare le basi di conoscenza espresse tramite logiche descrittive, che sono composte da due componenti: ABox e TBox (figura 2.3). Il TBox contiene la conoscenza relativa alla terminologia o *terminological knowledge*, che è espressa tramite dichiarazioni che descrivono proprietà generali dei concetti. La forma elementare di dichiarazione è la definizione di un concetto, cioè la dichiarazione di un concetto in termini di concetti precedentemente definiti. Ad esempio, possiamo definire la classe `Uomo` come $\texttt{Uomo} \equiv \texttt{Persona} \sqcap \texttt{Maschio}$, il che significa che un'istanza della classe `Uomo`, come ad esempio `Alan Turing`, è un'istanza sia della classe `Persona` che della classe `Maschio`. È anche possibile effettuare dichiarazioni del tipo $\texttt{Uomo} \sqsubseteq \texttt{Persona}$, che indicano che tutte le istanze di `Uomo` sono anche istanze di `Persona`. Questo approccio consente di esprimere ogni concetto complesso come una sostituzione di concetti semplici o concetti atomici. Per garantire il successo di questo approccio, è necessario che vi sia una sola definizione per concetto e

che tali definizioni siano acicliche, cioè non devono presentare definizioni ricorsive.

Una delle principali attività nella costruzione di una terminologia è la classificazione, che consiste nell'organizzare i concetti all'interno di una gerarchia tassonomica e implica la verifica se un concetto è incluso in un altro (questa relazione è chiamata subsumption). Uno dei principali servizi di deduzione che può essere utilizzato nel TBox è l'implicazione logica, che verifica se una relazione generica (come la subsumption tra due concetti) può essere considerata una conseguenza logica delle dichiarazioni presenti nel TBox. L'ABox contiene tutte le affermazioni relative agli individui della base di conoscenza, ad esempio relazioni tra individui tramite un predicato o specificazioni della classe a cui un individuo appartiene. Per questo motivo, si dice che l'ABox contiene conoscenza asserzionale o *assertional knowledge*. Uno dei principali servizi di inferenza che può essere eseguito nell'ABox è il controllo di istanza, che verifica se un individuo è un'istanza di una determinata classe. Un altro servizio importante, che può essere espresso in termini di controllo di istanza, è la consistenza della knowledge base, che verifica se ogni concetto nella knowledge base ha almeno un individuo. Verificare se una knowledge base è soddisfacibile equivale a verificare se esiste un'interpretazione, e quindi un modello, in cui non ci siano contraddizioni.

Un importante servizio legato al ragionamento è la *class disjointness*, ovvero verificare se date due classi nella base di conoscenza esse siano disgiunte, cioè non vi sia nessun individuo che sia istanza di entrambe (fondamentale in questo lavoro di tesi).

Un'ultima importante puntualizzazione necessaria, che incide fortemente sui risultati dei processi di inferenza, è descrivere l'assunzione di mondo aperto (*Open World Assumption* - OWA) e l'assunzione di mondo chiuso (*Closed World Assumption* - CWA) [HKR10]. Quando la CWA è in vigore, tutte le affermazioni non presenti nella base di conoscenza sono da considerarsi false, mentre quando la OWA è in vigore, non è possibile esprimersi circa la veridicità di tali affermazioni mancanti.

Si consideri un semplice esempio: supponiamo di aver definito nella

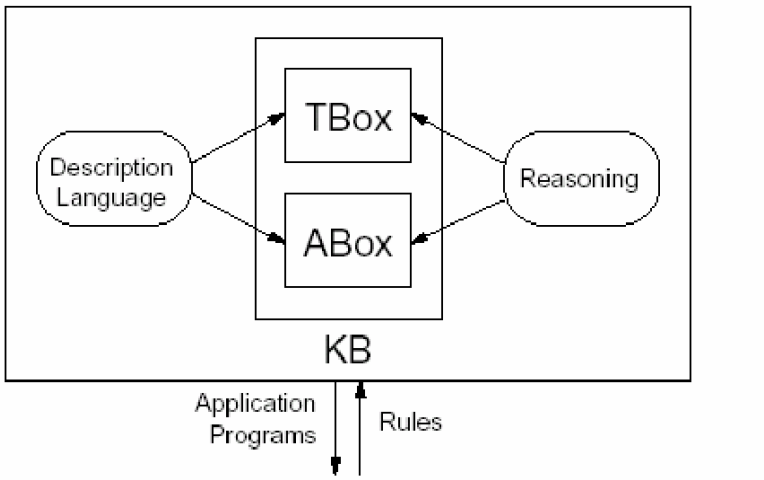



Figura 2.3: Logica descrittiva

nostra base di conoscenza i concetti di `Matematico` e `Filosofo`, e di sapere che "Alan Turing è un matematico", informazione esprimibile sotto forma di tripla con (`Alan Turing, typeOf, Matematico`). Se la CWA fosse in vigore, potremmo automaticamente dedurre che "Alan Turing non è un filosofo". Tuttavia, tale approccio risulta giustificato in sistemi quali, ad esempio, i modelli relazionali, ma nell'ambito dei KG e del Semantic Web tale approccio risulterebbe inappropriato. I KG peccano di completezza e sono in grado di mettere in collegamento le informazioni tra loro (basti pensare ai Linked Open Data), pertanto non si può escludere a priori che una data tripla risulti vera o falsa in quanto potrebbe essere semplicemente mancante o specificata in un altro KG.

Pertanto, la tripla (`Alan Turing, typeOf, Filosofo`), che risulta effettivamente vera, non può essere ritenuta né falsa né vera; il suo valore di verità semplicemente non è noto. Questo spiega il motivo per cui, in generale, nei KG e nel Semantic Web, l'assunzione in vigore è quella di mondo aperto. Rispetto ai modelli che verranno considerati in questo lavoro di tesi, sarà necessario generare delle triple che risultino false con la maggiore probabilità possibile. La spiegazione appena fornita circa la OWA rende evidente la complessità di tale compito che si è cercato di risolvere sfruttando l'espressività del linguaggio OWL e altre euristiche: ad esempio, se la nostra base di conoscenza specificasse che le classi

`Matematico` e `Filosofo` sono disgiunte, allora potremmo dedurre che la tripla (`Alan Turing, typeOf, Filosofo`) è certamente falsa.

# Capitolo 3

# Lavori Correlati

Per poter fare previsioni su nuove conoscenze a partire da quelle presenti nel Knowledge Graph, è fondamentale riuscire a modellare le correlazioni e le dipendenze che esistono tra le variabili del sistema, ovvero le entità e le relazioni che costituiscono il KG.

Gli Energy-Based Models (EBM), come descritti in [LCH06] [MdF16], offrono un approccio per affrontare problemi di apprendimento sia probabilistici che non, come la classificazione e la predizione. Questi modelli cercano di rispondere a domande come: "Dati certi valori per la variabile X, quali valori di Y sono più compatibili?" [LCH06]. Questo significa che il modello è progettato per fornire una singola risposta. Ad esempio, nel caso della classificazione, considerando la tripla (`Alan Turing`, `nato_a`, `Londra`), ci si aspetta che il modello dica se questa tripla è vera o falsa, ossia i valori di verità rappresentati dalla variabile Y. Allo stesso modo, nel caso della predizione, considerando la tripla (`Alan Turing`, `nato_a`, ?), ci si aspetta che il modello fornisca come risposta l'entità `Londra`.

Gli EBM modellano le dipendenze tra le variabili del sistema utilizzando funzioni scalari, anche chiamate funzioni energetiche. Queste funzioni rappresentano la similarità o la compatibilità tra le variabili. Data quindi una generica tripla, se vi è una forte compatibilità tra queste variabili, ovvero se la tripla rappresenta una conoscenza vera, la funzione energetica avrà valori bassi, altrimenti avrà valori alti [LCH06]. Gli EBM si articolano in due fasi principali: la fase di inferenza e la fase di apprendimento.

Nella fase di inferenza, data una configurazione di variabili, si effet-

tua una previsione o una classificazione trovando il valore associato alle variabili sconosciute che minimizza il valore della funzione energetica, rendendo le variabili coerenti con la conoscenza esistente. La fase di apprendimento, invece, consiste nel trovare la formulazione migliore per la funzione energetica, in modo tale che associ valori bassi a configurazioni compatibili, come le triple che rappresentano conoscenze vere, e valori alti a configurazioni incompatibili, come le triple che rappresentano conoscenze false [LCH06]. L'introduzione degli EBM e successivamente degli EBEM è stata motivata dalla possibilità di descrivere i modelli di Translational Distance, una specifica classe di modelli basati su Knowledge Graph Embedding a cui appartiene TransE.

## 3.1 TransE

TransE [Bor+13] è stato uno dei primi modelli basati su *Translational Distance Model*. Innanzitutto, essendo un modello che lavora sui knowledge graph, bisogna specificare che i nodi corrispondono alle entità mentre gli archi del tipo $(head, label, tail)$ denotati con $(h, l, t)$ indicano che esiste una relazione tra l'entità $h$ e l'entità $t$.

Ad ogni entità, pertanto, è associato un vettore multidimensionale in uno spazio vettoriale (spazio di embedding), così come anche per le relazioni viene associato un vettore multidimensionale. Nello specifico, per le relazioni viene associato un'operazione di traslazione.

Data una tripla, bisogna impiegare la traslazione associata alla relazione che mette in correlazione le due entità $h$ e $t$ (attraverso una funzione di score che varia di modello in modello), come in figura 3.1. È stata scelta la traslazione per due motivi:

1. La motivazione principale si basa sull'osservazione che le correlazioni di tipo gerarchico sono comuni nelle basi di conoscenza e richiedono un metodo efficace per essere modellate. Un approccio adatto potrebbe essere basato sulla traslazione vettoriale. Prendiamo ad esempio un albero come struttura dati adatta per rappresentare relazioni gerarchiche: in una disposizione bidimensionale, possiamo

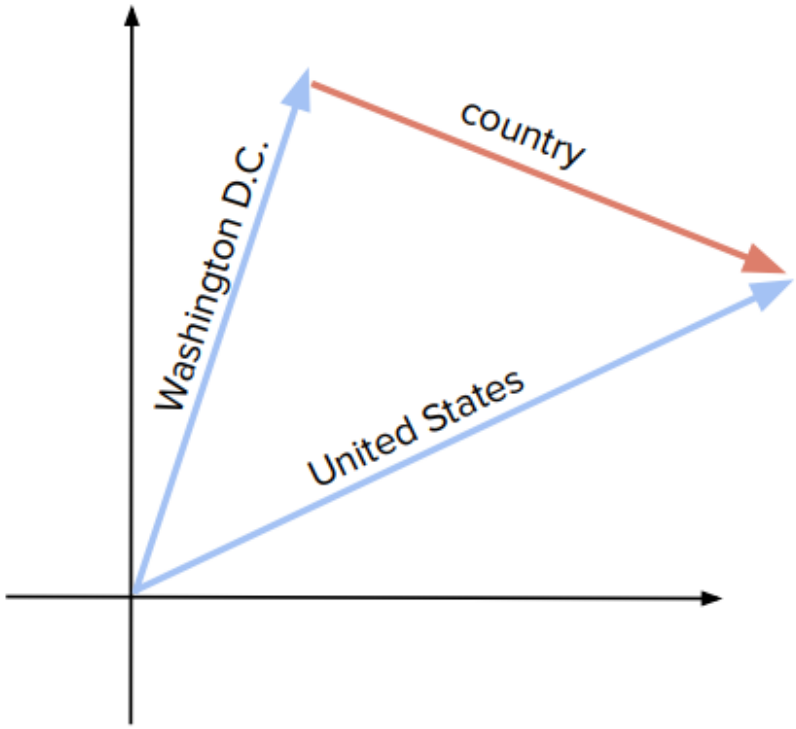



Figura 3.1: Esempio di come TransE rappresenta i modelli e le interazioni tra entità e relazioni nello spazio vettoriale (immagine presa da [Bia+20]).

associare a ciascun nodo il vettore di embedding di un'entità. Per correlare due nodi, potremmo utilizzare la traslazione vettoriale tra di essi, che si è dimostrata efficace.

2. Un'altra motivazione significativa è fornita da [Mik+13], che si concentra sulle relazioni 1:1. Queste relazioni vengono modellate attraverso traslazioni che mettono in relazione vettori di embedding associati a parole presenti in un testo non strutturato, rappresentando tipi diversi. Ad esempio, la relazione `nato a` collega la parola `Alan Turing`, che rappresenta una persona, alla parola `Londra`, che rappresenta una città secondo l'approccio di TransE. L'efficacia dimostrata da questo approccio suggerisce la possibilità di estenderlo anche alle relazioni e alle entità (caratterizzate da tipi differenti) presenti in numero molto maggiore nei Knowledge Graph [Hog+20].

Un generico KG $G$, essendo innanzitutto un grafo, può essere descritto

tramite una coppia di nodi ed archi $G = (E, R)$, con $E$ l'insieme dei nodi ed $R$ degli archi; tuttavia, $G$ può essere rappresentato come un insieme di triple (h,r,t) dove $h, t \in E$ sono chiamate entità di "testa" e di "coda" con $E$ insieme delle entità mentre $R$ insieme delle relazioni.

Ogni modello basato su Translational Model, di cui *TransE* risulta essere uno dei primi, adotta una propria funzione di score per mettere in relazione i vettori delle entità e relazioni tra di loro. Dal punto di vista matematico, quindi, in TransE si impone una relazione del tipo $h + r \approx t$ se la tripla $(h, r, t)$ è positiva. Qualora invece $(h, r, t)$ è falsa, allora $h + r \neq t$, cioè la somma tra $h$ ed $r$ non deve essere uguale a $t$ o vicino ad esso. La funzione di score adottata da TransE è pertanto:

$$f_r(h, t) = ||h + r - t||_2^2$$

La formula descrive la distanza euclidea tra il vettore somma $h + r$ e il vettore $t$ (grazie alla norma 2) e la si eleva al quadrato per enfatizzare maggiormente le distanze più elevate, quindi per triple errate.

**Apprendimento.** L'apprendimento del modello TransE passa dalla definizione di una funzione di loss $L$ da minimizzare. Per ottenere tale risultato è necessario optare per una particolare formulazione della funzione di loss $L$ basata su di uno specifico iperparametro $\gamma$ detto margine:

$$L = \sum_{(h,r,t)\in\Delta} \sum_{(h',r,'t)\in\Delta'} [\gamma + d(h + r, t) - d(h' + r, t')]_+$$

dove $[x]_+$ denota la parte positiva di x. La funzione di loss favorisce valori bassi di energia per le triple positive piuttosto che per le triple corrotte. L'ottimizzazione è effettuata grazie allo stochastic gradient descent [Rud16] nella modalità minibatch. Tale operazione di aggiornamento sui parametri viene iterata un certo numero di volte, ed ogni iterazione è definita epoch. Durante il processo di ottimizzazione, vi è il rischio che, nel tentativo di minimizzare la funzione di loss $L$, i valori

associati alle norme dei vettori di embedding vengano artificiosamente incrementati compromettendo il corretto addestramento di tali vettori [Bor+13]. Per evitare tale fenomeno quindi, si impongono dei vincoli su tali valori delle norme euclidee associati ai vettori di embedding delle entità (le relazioni risultano dunque escluse) nel seguente modo:

$$\forall e \in E, ||e||_2 \leq 1$$

dove E è l'insieme delle entità.

Rimangono comunque diversi limiti in tale approccio, poiché risulta eccessivamente elementare nel rappresentare relazioni più complesse di quelle di tipo 1:1 e in generale nel riuscire a rappresentare correttamente l'espressività e le proprietà che possono essere descritte da linguaggi formali come RDFS e OWL su entità e relazioni, come ad esempio la proprietà transitiva, simmetrica e riflessiva, per le relazioni.

## 3.2 KG2E

La rappresentazione di un KG eseguita da alcuni modelli come ad esempio TransE [Bor+13] o anche TransC [Lv+18], incorporano entità e relazioni in uno spazio vettoriale ottimizzando una funzione di perdita globale che garantisce che i punteggi delle triple positive siano superiori a quelli negative. Questi modelli, però, considerano sempre tutte le entità e le relazioni allo stesso modo e ignorano le loro incertezze.
KG2E [He+15], infatti, cerca di superare questa mancanza adottando un embedding basato sulla densità di probabilità per modellare esplicitamente la certezza delle entità e delle relazioni, apprendendo le rappresentazioni dei KG nello spazio delle distribuzioni gaussiane multidimensionali. Ogni entità/relazione è rappresentata da una distribuzione gaussiana, in cui la media indica la sua posizione e la covarianza (nello specifico covarianza diagonale) cerca di rappresentare la dimensione all'interno dello spazio vettoriale (come in figura 3.2).

KG2E sostiene che le incertezze nei KG possano essere influenzate da diversi fattori, tra cui lo squilibrio tra la testa e la coda delle relazioni,

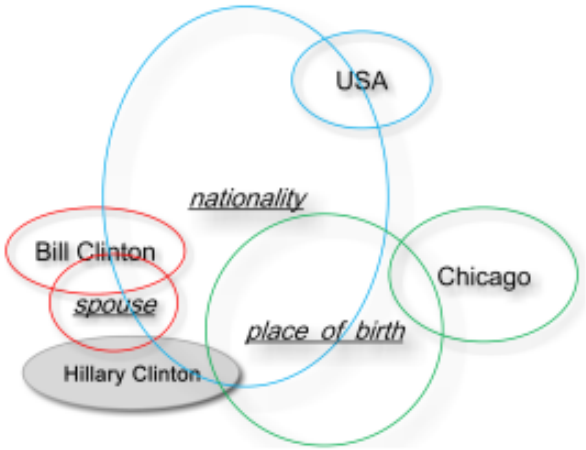



Figura 3.2: Illustrazione grafica delle medie e delle covarianze (diagonali) delle entità e relazioni in un embedding gaussiano. Immagine presa da [He+15].

il numero differente di triple collegate per diverse relazioni ed entità e l'ambiguità delle relazioni stesse, e così via. Ad esempio, ipotizziamo che un'entità che contiene meno triple abbia una maggiore incertezza e che una relazione che collega più triple con contesti più complessi abbia anche una maggiore incertezza. Inoltre, le diverse parti delle relazioni possono contenere un numero molto diverso di entità (come la parte della testa e della coda per genere o nazionalità) e le relazioni ad alta frequenza (ad esempio, la nazionalità) collegano più coppie di entità rispetto a quelle a bassa frequenza (ad esempio, la religione). Ciò indica che le variazioni di incertezza con diverse entità e relazioni in un KG variano enormemente ed è auspicabile considerare questo problema durante l'apprendimento delle rappresentazioni di un KG in uno spazio latente. La media di ogni vettore di embedding corrisponde alla posizione nello spazio vettoriale, mentre la matrice di covarianza indica l'incertezza con la quale si relaziona con le altre entità.

La funzione di loss è la seguente:

$$L = \sum_{(h,r,t)\in\Delta} \sum_{(h',r',t')\in\Delta'} [\gamma + \mathcal{E}(h,r,t) - \mathcal{E}(h',r',t')]_+$$

dove $[x]_+$ ha lo scopo di ottenere il massimo tra 0 e $x$, $\gamma$ è il margine che separa le triple positive dalle negative e $\mathcal{E}$ è la funzione di score utilizzata. L'ottimizzatore utilizzato è lo Stochastic Gradient Descent

[Rud16]. KG2E [He+15], tuttavia, utilizza due funzioni di score: la Kullback-Leibler divergence (KL) e la expected likelihood (EL).

La funzione di score utilizzata per la **KL** è la seguente:

$$\mathcal{E}(h,r,t) = \frac{1}{2}(D_{\text{KL}}(P_e, P_r) + D_{\text{KL}}(P_r, P_e))$$

dove $P_e \sim \mathcal{N}(\mu_h - \mu_t, \Sigma_h + \Sigma_t)$, cioè è la distribuzione di probabilità risultante da $h$ a $t$, mentre $P_r \sim \mathcal{N}(\mu_r, \Sigma_r)$. La KL divergence è un metodo per misurare la similarità di due distribuzioni di probabilità ed è *asimmetrica*.

Un'altra funzione di score utilizzata è la **expected likelihood**, dove la formula è la seguente:

$$\mathcal{E}(h,r,t) = \log \mathcal{E}(P_e, P_r)$$

Nello specifico, l'expected likelihood di KG2E [He+15] ha fornito uno spunto interessante per la funzione di score delle `typeOf` nel modello TransM. Il modello KG2E [He+15] è riuscito ad avere prestazioni migliori allo stato dell'arte rispetto a molti modelli.

## 3.3 Modelli Orientati alle Classi

I Knowledge Graph Embeddings orientati alle classi sono una variante dei modelli di embedding utilizzati per rappresentare le relazioni tra entità in un KG. Questi modelli cercano di catturare le relazioni gerarchiche e le proprietà delle classi nel KG. In un KG, le entità e le relazioni sono spesso organizzate in gerarchie, in cui le classi rappresentano concetti più ampi e generici, mentre le istanze rappresentano esempi specifici di tali concetti. Ad esempio, nel dominio dei film, si potrebbe avere una classe `Film` e diverse istanze di film come `Star Wars` e `Il Padrino`. Le relazioni tra le entità possono essere di diversi tipi, come `genere di`, `diretto da` o `ha come attore principale`. L'utilizzo di knowledge graph embeddings orientati alle classi può portare a una migliore rappresentazione delle relazioni gerarchiche e delle proprietà delle classi nel

KG. Ciò può essere utile in nei task di triple classification, cioè identificare le triple vere o false e nel task dell'estrazione di conoscenza ovvero derivare regole logiche dal modello. Esistono numerosi modelli orientati alle classi [WQW21], ma in questo lavoro di tesi ci si sofferma su quelli più vicini in quanto a task di triple classification ed estrazione di conoscenza.

### 3.3.1 TransC

Lo scopo di TransC [Lv+18] è quello di suddividere le entità in concetti e instanze per riuscire a rappresentare la relazione di transitività `isA`. L'idea alla base è ispirata al modo in cui le persone organizzano i concetti all'interno della mente, ovvero in modo gerarchico e soprattutto le istanze di un concetto alla quale appartengono dovrebbero trovarsi abbastanza vicini ad esso ([PTG12]). Per far ciò, per ogni concetto $c \in C$ si associa una sfera $s(p, m)$ dove $p \in \mathbb{R}^k$ mentre $m$ è il centro e raggio della sfera. Per ogni istanza e relazione, invece, si associa un vettore di embedding. Innanzitutto, si definiscono tre differenti tipologie di triple all'interno della KB, ciascuna con una differente score function:

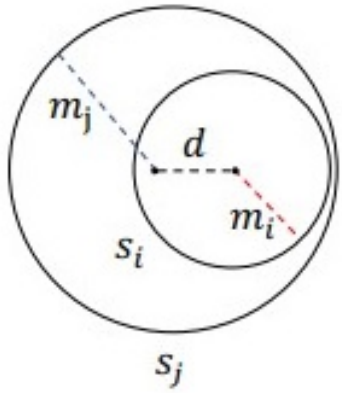
1. **Tripla `typeOf`:** data una tripla `typeOf` $(i, r_e, c)$ vera, il soggetto $i$ dovrebbe trovarsi all'interno della sfera $s$ per rappresentare la relazione di `typeOf` tra $i$ e $c$. Tuttavia, potrebbe capitare un'altra posizione cui $i$ è al di fuori della sfera (ma questo succede solo quando la rappresentazione embedding non è ancora stata ottimizzata a sufficienza). La score function è definita come:

$$f_e(i, c) = ||i - p||_2 - m$$

2. **Tripla `subClassOf`:** data una tripla `subClassOf` $(c_i, r_c, c_j)$ vera, in questo caso si definisce innanzitutto la distanza tra i due centri delle sfere come:

$$d = ||p_i - p_j||_2$$

In questo caso, **la sfera $s_i$ dovrebbe trovarsi all'interno della**




$(a)\ d < |m_i - m_j| \wedge m_i < m_j\ (GOAL)$

Figura 3.3

**sfera** $s_j$ per rappresentare la relazione di `subClassOf` tra $c_i$ e $c_j$, ovvero come nella Figura 3.3.

Ci possono essere tre casi differenti:

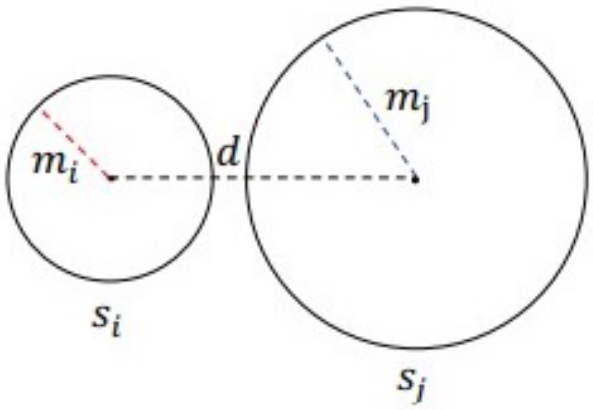
(a) $s_i$ **non interseca** $s_j$**:** in questo caso la rappresentazione embedding deve essere ottimizzata ulteriormente, cioè le due sfere devono avvicinarsi ancor di più. La score function, pertanto sarà (vedere Figura 3.4):

$$f_c(c_i, c_j) = d + m_i - m_j$$

(b) $s_i$ **interseca** $s_j$**:** identico al caso a (cfr. Fig. 3.5):

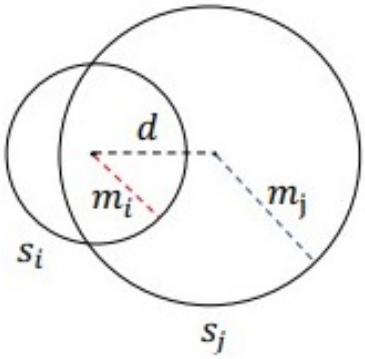
(c) $s_j$ **è all'interno di** $s_i$**:** è esattamente il contrario del nostro target (cioè avere $s_i$ all'interno di $s_j$). In questo caso basta ridurre quindi $m_j$ e aumentare $m_i$. La loss function, pertanto sarà (vedere Figura 3.6):

$$f_c(c_i, c_j) = m_i - m_j$$




(b) $d \geq |m_i + m_j|$

Figura 3.4



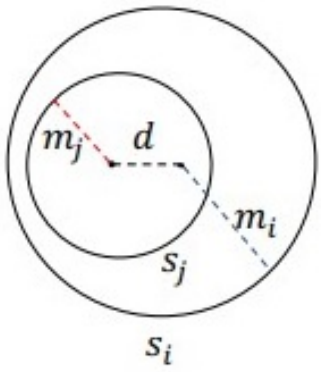

(c) $|m_i - m_j| \leq d < |m_i + m_j|$

Figura 3.5

3. **Tripla relazionale:** per ciascuna tripla relazionale generica $(h, r, t)$, TransC cercherà di imparare le rappresentazioni dei vettori $h$, $t$, $r \in \mathbb{R}^k$ sia per le istanze e sia per le relazioni. In questo caso la funzione score è identica a TransE, cioè:

$$f_r(h, t) = ||h + r - t||_2^2$$

La funzione di loss è calcolata come combinazione delle tre funzioni di score. Per poter quindi effettuare una nuova predizione, supponiamo di avere due triple $(i, r_e, c_i)$ e $(c_i, r_c, c_j)$ entrambe vere. Questo significa che $i$ si trova all'interno di $s_i$ e a sua volta $s_i$ si trova all'interno di $s_j$. Grazie a questa informazione, ne deriva facilmente che anche $i$ si trova




$$(d)\ d < |m_i - m_j| \wedge m_i \geq m_j$$

Figura 3.6

all'interno di $s_j$, cioè che esisterà molto probabilmente anche la tripla vera $(i, r_e, c_j)$.

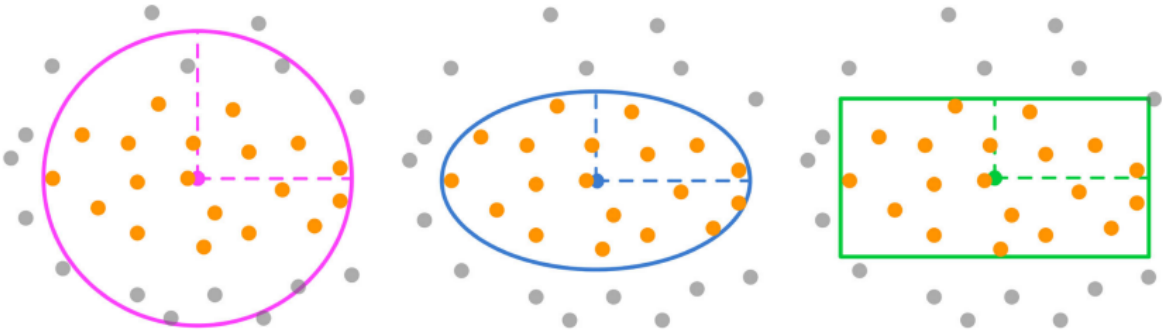



Figura 3.7: Le regioni chiuse da diversi colori rappresentano differenti embedding della stessa classe, mentre i punti arancioni rappresentano le istanze della classe. Da notare come la distribuzione dei punti è la stessa in tutti e tre gli embedding, ma la regione delimitata dalla sfera fa sì che vengano incluse anche entità non istanze della classe. Immagine fornita da [Yu+23].

### 3.3.2 TransEllipsoid

TransEllipsoid [Yu+23] cerca di superare l'isotropia[1] geometrica dell'ipersfera derivante da TransC [Lv+18] proponendo un modello anisotropico[2] orientato alle classi la cui forma geometrica è quella di un ellissoide.
Innanzitutto, l'isotropia è un concetto che si riferisce alla proprietà di uniformità o omogeneità di un oggetto o di un sistema nelle diverse direzioni, cioè non dipende dalla direzione in cui viene osservato o misurato. In fisica, ad esempio, l'isotropia è spesso associata alla simmetria rispetto alle direzioni nello spazio. Ad esempio, una (iper)sfera è isotropica, mentre un (iper)ellissoide è anisotropico.

La Figura 3.7 è rappresentativa del problema che modelli come TransC [Lv+18] non riescono a risolvere, cioè come già spiegato l'isotropia. Una differente geometria associata alla classe (come l'ellisse o un rettangolo) invece è in grado di avere una precisione migliore e quindi in generale migliori prestazioni.

Per ciascuna classe $c \in \mathbf{C}$, TransEllipsoid associa una regione

[1] https://it.wikipedia.org/wiki/Isotropia

[2] https://it.wikipedia.org/wiki/Anisotropia

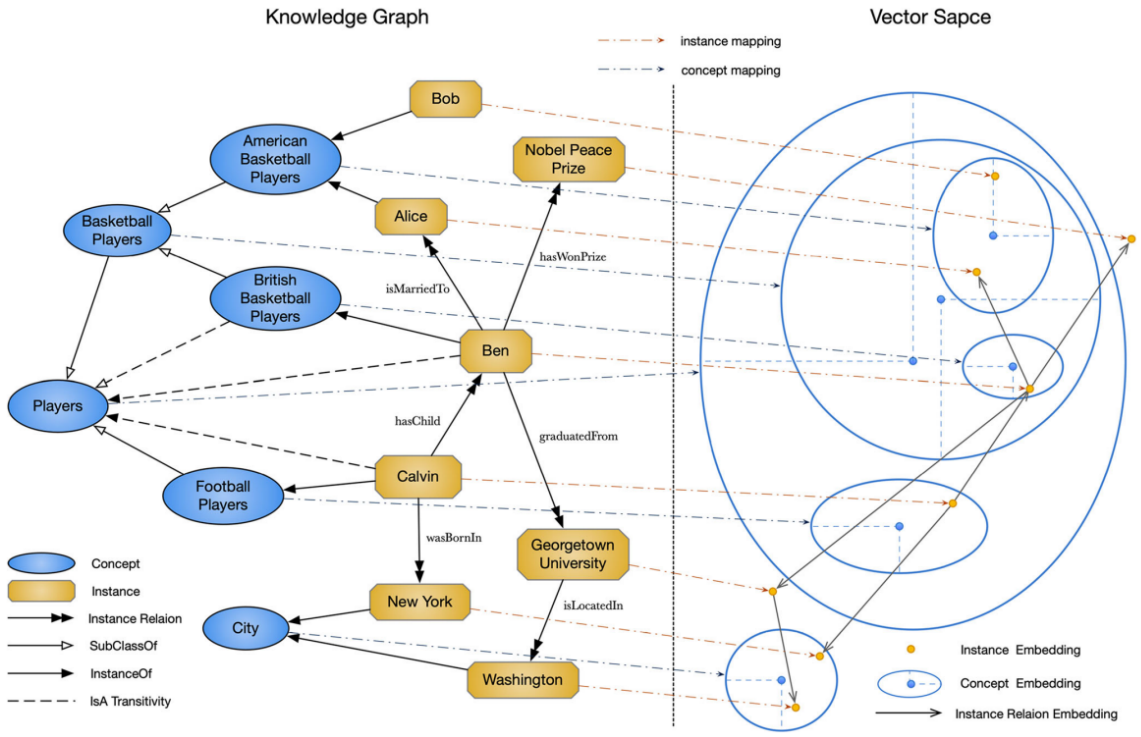



Figura 3.8: Un esempio di istanze e classi in un KG e di come gli ellissoidi inglobano le istanze. Immagine fornita da [Yu+23].

ellissoidale $G_o(c, b)$ (si veda Fig. 3.8 e Fig. 3.9) come segue:

$$G_o(c,b) = \left\{ (x_1, x_2, ..., x_k) \middle| \sum_{j=1}^{k} \left( \frac{x_j - c_j}{b'_j} \right)^2 \leq r_c^2 \right\}$$

dove $k$ è la dimensione dell'embedding, $r_c$ è il learning rate, $c$ è il centro dell'ellissoide, $b$ è la lunghezza lungo l'asse corrispondente. Questa formula può anche essere vista come un ellissoide trasformato in una sfera dove il centro è $(\frac{c_1}{b'_1}, \frac{c_2}{b'_2}, \frac{c_3}{b'_3}, ..., \frac{c_k}{b'_k})$ e il raggio è $r_c$. Di seguito le funzioni di score utilizzate per le differenti triple:

1. **Tripla `typeOf`:** data una tripla `typeOf` $(i, r_e, c)$ vera, il soggetto $i$ dovrebbe trovarsi all'interno dell'embedding $G_o(c, b)$ per rappresentare la relazione di `typeOf` tra $i$ e $c$. La loss function è definita come:

$$\mathbf{f}_{re}(i,c) = \left[ \sum_{j=1}^{k} \left( \frac{i_j - c_j}{b'_j} \right)^2 - r_c^2 \right]_+$$

2. **Tripla `subClassOf`:** data una tripla `subClassOf` $(c_m, r_c, c_n)$ vera, la

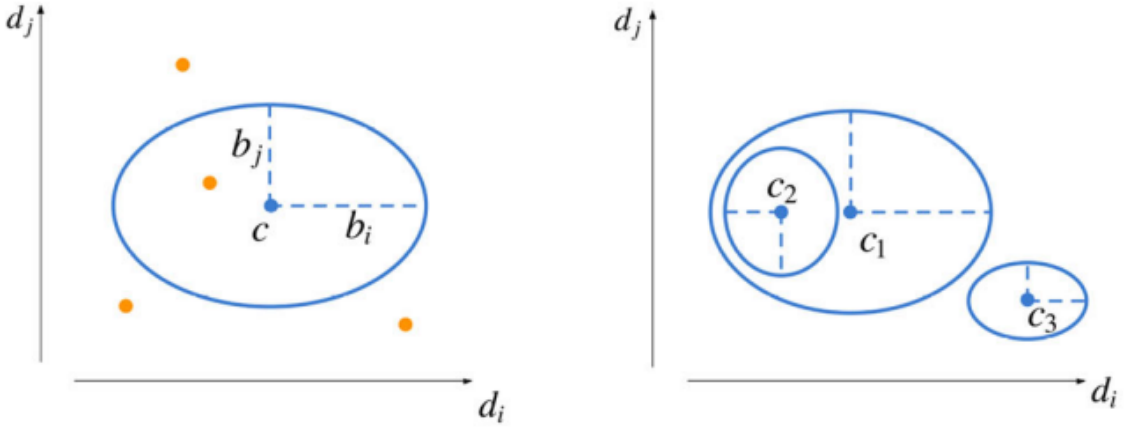



Figura 3.9: Ogni punto arancione rappresenta un istanza dell'embedding. Ciascuna regione chiusa da un punto e linea blu rappresenta una classe. Immagine fornita da [Yu+23].

regione dell'embedding $G_{om}(c_m, b_n)$ deve essere inclusa nella regione $G_{on}(c_n, b_n)$, cioè $G_{om} \subset G_{on}$. La funzione di score pertanto è:

$$\mathbf{f}_{rc}(c_m, c_n) = \left[ \sum_{j=1}^{k} \left( \frac{c_{mj}}{b'_{mj}} - \frac{c_{nj}}{b'_{nj}} \right)^2 + r_{cm}^2 - r_{cn}^2 \right]_+$$

Se $\mathbf{f}_{rc}(c_m, c_n) = 0$, allora la relazione `subClassOf` è soddisfatta.

3. **Tripla relazionale**: identica a TransE [Bor+13].

La funzione di loss, similmente a TransC [Lv+18], viene compilata come combinazione lineare delle tre funzioni di loss associate alle triple `typeOf`, `subClassOf` e relazionali. Grazie alla rappresentazione più complessa dell'embedding ellissoidale, TransEllipsoid [Yu+23] è riuscito a superare TransC [Lv+18] nei task di triple classification e link prediction.

### 3.3.3 KEC

Una classe (o concetto) rappresenta una risorsa fondamentale per la comprensione umana. I concetti comuni sono fondamentali e molto generici, giocando un ruolo cruciale nell'apprendimento umano. Pertanto, l'utilizzo delle informazioni derivate dai concetti può portare vantaggi significativi, anche in termini di risultati, nel contesto della costruzione di un Knowledge Graph (KG). Ad esempio, Microsoft Concept Graph [Che+15] (Figura 3.10) associa le entità a diversi concetti semantici, con una probabilità che dipende dal contesto. Ad esempio, l'entità `mela` può essere associata a `frutta`, `azienda`, `brand` o `cibo`, ma la probabilità di tali associazioni varia a seconda del contesto in cui viene utilizzata. A differenza degli embedding basati sul testo, gli embedding basati sui concetti sono più generici e non dipendono dal contesto. Ad esempio, se il contesto riguarda la tecnologia, gli embedding basati sul testo potrebbero facilmente associare `mela` a `azienda` o `Apple` in quanto il testo contiene frequenti menzioni di `iPhone`, `hardware` e così via. Tuttavia, in questo caso, sarebbe difficile parlare della mela come `frutto`.

Al contrario, i concetti associati alla parola `mela` sono molteplici e includono contesti diversi, ciascuno con una probabilità associata. Il modello KEC (Knowledge Graph Embedding with Concepts) [GSL19] combina la correlazione delle triple simboliche con un grafo di classi, eseguendo il processo di embedding nello spazio delle classi. In particolare, la probabilità di una tripla viene misurata proiettando il vettore di loss nello spazio delle classi come un iperpiano che rappresenta la rilevanza delle classi per le entità.

Per imparare le rappresentazioni delle classi ed entità in modo tale da catturare la relazione semantica tra di essi, KEC utilizza il modello skip-gram [Mik+13], ovvero un modello che cerca di generare le rappresentazioni delle parole che sono adeguate nel predire il loro contesto. Per imparare simultaneamente l'embedding delle entità e dei concetti, è stato utilizzato un metodo che combina le informazioni sul contesto attraverso delle parole in modo da predire le parole di contesto, ovvero ad esempio: se $e_t$ è l'entità target, allora i suoi concetti $(c_1, c_2, ..., c_k)$

| Instance | Concept | Relations |
|---|---|---|
| apple | fruit | 6315 |
| apple | company | 4353 |
| apple | food | 1152 |
| apple | brand | 764 |
| apple | fresh fruit | 750 |
| apple | fruit tree | 568 |
| apple | crop | 483 |
| apple | corporation | 280 |
| apple | manufacturer | 279 |
| apple | firm | 257 |

Figura 3.10: `apple` in Microsoft Concept Graph [Che+15]. Immagine fornita da [GSL19].

saranno combinati con $e_t$ per predire il contesto delle entità. Per ciascuna coppia ($e_t$, $e_c$), skip-gram associa quindi la probabilità che $e_c$ sia il contesto di $e_t$ o di $c_i$ usando la funzione softmax:

$$P(e_c|e_t) = \frac{\exp(e_c \cdot e_t)}{\sum_{e \in E} \exp(e \cdot e_t)}$$

$$P(e_c|c_i) = \frac{\exp(e_c \cdot c_i)}{\sum_{e \in E} \exp(e \cdot c_i)}$$

dove $e_t$, $e_c$ e $c_i$ sono le rappresentazioni delle entità target, l'entità contesto e la classe dell'entità target rispettivamente. KEC, quindi, definisce la log probabilità da massimizzare come:

$$L = \frac{1}{|D|} \sum_{(e_c, e_t) \in D} \log P(e_c|e_t) + \sum_{c_i \in C(e_t)} \log P(e_c|c_i)$$

dove $D$ è l'insieme delle coppie entità target-contesto mentre $C(e_t)$ denota l'insieme delle classi associate ad $e_t$.

Per caratterizzare una forte relazione tra le triple e le informazioni delle classi si costruisce innanzitutto un iperpiano con il vettore normale $c$ come:

$$c = C(e_h, e_t) = \frac{e_h - e_t}{||e_h - e_t||_2^2}$$

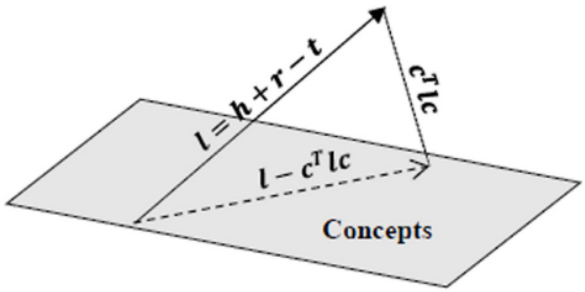



Figura 3.11: Vettore loss proiettato nell'iperpiano delle classi. Immagine fornita da [GSL19].

La funzione di score utilizzata è pertanto:

$$f_r(h,t) = -\lambda||l - c^T lc||_2^2 + ||l||_2^2$$

dove $\lambda$ è un parametro che bilancia i due addendi; $l$ invece, come in TransE è $l = h + r - t$. La spiegazione geometrica è illustrata in figura 3.11. La funzione di loss utilizzata è identica a quella di TransE.

Inizialmente, il modello KEC ha la capacità di individuare la rilevanza concettuale. Lo scopo è proiettare la differenza tra la testa dell'entità tradotta $h'$ e la coda dell'entità $t$ sull'iperpiano concettuale. Questo si basa sul principio che se una linea giace su un iperpiano, tutti i punti di quella linea giacciono anche sull'iperpiano. Pertanto, le entità che hanno concetti rilevanti si trovano su un iperpiano coerente, e il vettore di differenza $h' - t$ si trova attorno all'iperpiano. Questo significa che una tripla corrotta si trova lontano dall'iperpiano, il che comporta una grande loss. D'altra parte, anche se la loss di una tripla corretta è significativa in termini di embedding, potrebbe essere inferiore (o migliore) dopo essere stata proiettata sull'iperpiano. Questo implica che le informazioni concettuali aumentano la rilevanza concettuale, migliorando l'embedding.

Ad esempio, se si considera il fatto (`Christopher Plummer, nazionalità, Canada`), è difficile dedurne la possibilità solo dall'embedding. In base alla previsione dei collegamenti con il modello TransE, viene classificato come un fatto impossibile (al 11.831º posto su 14.951). Tuttavia, in un grafo concettuale, `Christopher Plummer` è associato ai concetti di `Attore`

e `Star`, mentre `Canada` è associato ai concetti di `Paese` e `Nazione`. È evidente che esiste una relazione `nazionalità` tra i concetti delle due entità. Questo significa che le due entità sono rilevanti nello spazio semantico dei concetti. Come previsto, dopo la proiezione concettuale nel modello KEC, la tripla si posiziona al 49º posto su 14.951, il che la rende più plausibile.

In secondo luogo, il modello KEC offre un'espressione semantica precisa. Nel modello TransE, se i vettori di loss delle triple hanno la stessa lunghezza, vengono considerati equivalenti ed è difficile distinguerli. Tuttavia, il modello KEC incorpora la semantica concettuale per migliorare la capacità discriminante. In particolare, i vettori di loss con la stessa lunghezza vengono proiettati sugli iperpiani concettuali corrispondenti come risultato della loss, consentendo una divisione ragionevole delle perdite. Ad esempio, considerando una query che chiede quale regista ha realizzato il film `WALL-E`, ci sono due entità candidate: la risposta corretta `Andrew Stanton` e la risposta negativa `James Cameron`. Le perdite delle due triple sono quasi uguali nell'incorporazione della conoscenza, quindi è difficile distinguerle. Tuttavia, nello spazio concettuale, sia `WALL-E` che `Andrew Stanton` sono molto rilevanti per il concetto `Pixar`, mentre `James Cameron` non ha questo concetto. Di conseguenza, sia la query che la risposta corretta si trovano sull'iperpiano `Pixar`, mentre la query e la risposta negativa non si trovano nell'iperpiano concettuale associato corrispondente. Ciò significa che la loss proiettata della risposta corretta potrebbe essere molto inferiore rispetto a quella della risposta errata, semplificando la discriminazione.

KEC è riuscito ad apportare miglioramenti ai modelli base allo stato dell'arte come TransE nei task di link prediction e triple classification.

# Capitolo 4

# Estensioni Proposte

## 4.1 TransC-OWL

In questa tesi di di laurea, viene presentato il modello TransC-OWL, che rappresenta un'evoluzione del modello TransC [Lv+18]. Il modello TransC-OWL mantiene la stessa struttura di base di TransC, ma migliora l'efficacia introducendo elementi di TransOWL. Quest'ultimo sfrutta le potenzialità dei linguaggi formali RDFS e OWL per generare nuova conoscenza di base tramite l'utilizzo di un **reasoner**. La conoscenza generata viene quindi iniettata nel processo di addestramento del modello, al fine di migliorarne le prestazioni e l'accuratezza.

### 4.1.1 Iniezione di Conoscenza di Fondo

I linguaggi formali come RDFS e OWL consentono di descrivere le proprietà che caratterizzano le entità e le relazioni contenute in un KG. I metodi basati sulla regolarizzazione degli embedding mediante assiomi, come TransE, provano come l'utilizzo di tale conoscenza di fondo possa determinare un incremento nelle prestazioni grazie all'impiego di specifici vincoli nella funzione di loss per determinate entità e relazioni.

In TransOWL [San19] sono stati formulati specifici vincoli energetici per ciascun assioma considerato, che veicolano le modalità con cui i vettori di embedding vengono appresi durante l'addestramento. TransOWL ha dunque cercato di espandere il lavoro di TransE, generando un modello caratterizzato da due principali componenti volti all'iniezione di conoscenza di fondo all'interno del modello TransE durante la fase di addestramento:

1. **Reasoner:** L'impiego di un reasoner è motivato dalla necessità di riuscire a generare triple corrotte che risultino certamente negative, evitando dunque i falsi negativi, per poter attuare un addestramento più efficace. Il ragionamento effettuato durante la corruzione delle triple sfrutta gli assiomi, specificati mediante RDFS e OWL, definiti `subdomain`, `range`, `disjointWith`, `functionalProperty` sia per generare veri negativi sia per rilevare ed evitare falsi negativi.

2. **Iniezione di conoscenza di fondo:** Un insieme di assiomi differenti rispetto a quelli utilizzati dal reasoner sono impiegati per la definizione di vincoli energetici considerati in fase di addestramento affinché i vettori associati alle entità e alle relazioni interessate da tali assiomi rispecchino determinate proprietà. Gli assiomi in questione sono `equivalentClass`, `equivalentProperty`, `inverseOf`, `typeOf` e `subClassOf` (le ultime due derivanti dal modello **TransC**).

### 4.1.2 Funzione di Loss e Ottimizzatore

La funzione di loss di TransC-OWL adatta quelle di TransOWL a TransC tramite combinazione lineare di loss specifiche:

$$
\begin{aligned}
L = & \sum_{(h,r,t)\in\Delta} \sum_{(h',t)\in\Delta'} [\gamma_e + f_e(h,t) - f_e(h',t')]_+ \\
& + \sum_{(h,typeOf,t)\in\Delta} \sum_{(h',\texttt{typeOf},t')\in\Delta'} [\gamma_t + f_t(h,t) - f_t(h',t')]_+ \\
& + \sum_{(h,\texttt{subClassOf},t)\in\Delta} \sum_{(h',\texttt{subClassOf},t')\in\Delta'} [\gamma_s + f_s(h,t) - f_s(h',t')]_+ \\
& + \sum_{(h,\texttt{inverseOf},t)\in\Delta} \sum_{(h',\texttt{inverseOf},t')\in\Delta'} [\gamma_e + f_i(h,t) - f_i(h',t')]_+ \\
& + \sum_{(h,\texttt{equivProp},t)\in\Delta} \sum_{(h',\texttt{equivProp},t')\in\Delta'} [\gamma_e + f_q(h,t) - f_q(h',t')]_+ \\
& + \sum_{(h,\texttt{equivClass},t)\in\Delta} \sum_{(h',\texttt{equivClass},t')\in\Delta'} [\gamma_t + f_t(h,t) - f_t(h',t')]_+
\end{aligned}
$$

dove $\gamma_e$, $\gamma_t$, $\gamma_s$ sono i margini attribuiti a ciascuna funzione, ovvero possono essere visti come una misura di quanto i fatti previsti dal modello si allontanino dai fatti reali del knowledge graph.

Minimizzare il margine aiuta a migliorare l'accuratezza e la coerenza delle previsioni del modello, garantendo che le entità e le relazioni coinvolte in un fatto del knowledge graph siano rappresentate in modo coerente nello spazio vettoriale. Nello specifico, le prime tre funzioni di loss (triple relazionali, `typeOf` e `subClassOf`) derivano direttamente dal modello TransC, le ultime tre invece da TransOWL.

La funzione di loss $L$ di TransC-OWL è stata presentata come una funzione da minimizzare per trovare dei parametri, dunque dei vettori di embedding, che siano in grado di rappresentare nel modo più preciso possibile la conoscenza espressa all'interno di un KG. Il problema di trovare il minimo di una funzione, il che equivale a trovare un punto di minimo avente come dimensione tutti i parametri associati agli embedding, può dunque essere risolto mediante l'utilizzo di diversi approcci, tuttavia in TransC-OWL è stato considerata solo la *Discesa del Gradiente* (GD) [Rud16].

Questi ultimi, infatti, effettuano, ad ogni iterazione dell'algoritmo, l'aggiornamento dei parametri del modello nella direzione opposta a quella dettata dal gradiente calcolato sulla funzione di loss $\Delta_\theta L(\theta)$; il calcolo della variazione che interessa ogni singolo parametro, graficamente rappresentabile come lo spostamento che un punto effettua sulla superficie della funzione $L$, è stabilito dalla tipologia di ottimizzatore impiegato e da iperparametri quali il learning rate $\eta$. Per minimizzare la funzione $L$, è necessario innanzitutto calcolarne il gradiente:

$$g_t = \Delta L$$

Successivamente, ciascun parametro verrà aggiornato di conseguenza in questo modo:

$$\Delta_t = -\theta g_t$$

dove $\Delta_t$ rappresenta la variazione relativa al parametro $\theta$ all'istante $t$ dell'iterazione durante la fase di addestramento, $g_t$ rappresenta la funzione gradiente all'istante $t$, ed è un iperparametro globale fissato, comunemente chiamato learning rate.

Nel contesto dei *Translational Distance Model*, le entità e le relazioni dei grafi non sono distribuite uniformemente. Di conseguenza, le entità meno frequenti verranno addestrate meno spesso, ma con le stesse variazioni dei parametri rispetto agli altri embedding più frequenti.

Nella fase di addestramento bisogna anche scegliere il giusto compromesso nel numero di iterazioni (epoche) $T$ dell'algoritmo GD nella variante stocastica (SGD) [Rud16]. In generale, un numero maggiore di iterazioni permette al modello di apprendere più a fondo la struttura del KG e migliorare la rappresentazione degli embedding. Tuttavia, un numero eccessivamente elevato di iterazioni potrebbe comportare un tempo di addestramento più lungo e un rischio di overfitting, in cui il modello si adatta eccessivamente ai dati di addestramento, ma non generalizzando bene su nuovi dati. Per evitare il rischio di overfitting, sono state utilizzate tecniche di regolarizzazione come la norma sui parametri del modello durante l'ottimizzazione. Queste tecniche penalizzano i valori dei parametri più elevati, riducendo così la complessità del modello e favorendo una migliore generalizzazione.

L'ottimizzazione dei parametri in TransC-OWL richiede un'attenta messa a punto degli iperparametri, tra cui il learning rate, la regolarizzazione e il numero di iterazioni. Per poter migliorare ancor di più la scelta degli iperparametri è possibile utilizzare tecniche di ricerca come la grid search o la ricerca casuale per individuare la combinazione ottimale di valori che massimizzi le prestazioni del modello sul set di validazione (non utilizzati però in TransC-OWL).

## 4.2 TransM

Prendendo spunto da TransC e TransC-OWL, in questo lavoro di tesi si propone anche TransM (nome che prende origine dalla funzione di score della relazione `subClassOf`). Si tratta di un modello orientato maggiormente alle classi, simile ai due precedenti, ma con maggiori ambizioni nella modellazione delle classi e delle relazioni tra di esse. Per questo motivo in questa sezione ci si concentrerà maggiormente sulle

classi, tralasciando invece le relazioni `non typeOf` (identiche a TransE e TransC).

### 4.2.1 Classi e Distribuzioni di Probabilità

A differenza di TransC, che associava a ciascuna classe un'ipersfera, le classi modellate in TransM sono rappresentate come iperellissoidi (vedi fig. 4.1). Pertanto, è necessario introdurre il concetto di matrice di covarianza, poiché essa consente di modellare una classe tramite un iperellisoide.

La matrice di covarianza è una matrice quadrata simmetrica che contiene le covarianze tra le diverse variabili di un dataset. La covarianza è una misura statistica che esprime la relazione di dipendenza lineare tra due variabili casuali. Nella matrice di covarianza, gli elementi sulla diagonale principale rappresentano le varianze delle rispettive variabili, mentre gli elementi al di fuori della diagonale rappresentano le covarianze tra le coppie di variabili. La covarianza tra due variabili $X$ e $Y$ è calcolata come la media del prodotto dei loro scarti dalla media.

La matrice di covarianza può essere rappresentata come:

$$\mathbf{C} = \begin{bmatrix} \mathrm{Cov}(X_1, X_1) & \mathrm{Cov}(X_1, X_2) & \dots & \mathrm{Cov}(X_1, X_n) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Cov}(X_2, X_2) & \dots & \mathrm{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(X_n, X_1) & \mathrm{Cov}(X_n, X_2) & \dots & \mathrm{Cov}(X_n, X_n) \end{bmatrix}$$

dove $\mathrm{Cov}(X_i, X_j)$ rappresenta la covarianza tra le variabili $X_i$ e $X_j$.

In TransM, tuttavia, viene utilizzata una forma particolare di matrice di covarianza, cioè la matrice di covarianza diagonale.

La matrice di covarianza diagonale è una matrice di covarianza in cui tutti gli elementi al di fuori della diagonale principale sono nulli. In altre parole, tutte le covarianze tra le variabili sono zero e la matrice contiene solo le varianze delle variabili stesse.

La matrice di covarianza diagonale può essere rappresentata come:

$$\mathbf{C} = \begin{bmatrix} \mathrm{Var}(X_1) & 0 & \dots & 0 \\ 0 & \mathrm{Var}(X_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathrm{Var}(X_n) \end{bmatrix}$$

dove $\mathrm{Var}(X_i)$ rappresenta la varianza della variabile $X_i$.

Quando la matrice di covarianza è diagonale, significa che le variabili non sono correlate tra loro. Questo implica che non esiste una relazione lineare significativa tra le variabili e ogni variabile è considerata indipendente dalle altre.

Dal punto di vista geometrico, la differenza principale tra una matrice di covarianza generale e una matrice di covarianza diagonale risiede nella forma e nell'orientamento dell'ellissoide di covarianza.

In una matrice di covarianza generale, la forma e l'orientamento dell'ellissoide dipendono dalle covarianze tra le variabili. Ad esempio, se due variabili sono positivamente correlate, l'ellissoide di covarianza sarà allungato lungo l'asse corrispondente a quelle due variabili, mentre se sono negativamente correlate, l'ellissoide sarà compresso lungo quell'asse. D'altra parte, una matrice di covarianza diagonale implica che le variabili sono statisticamente indipendenti e non influenzano reciprocamente le loro varianze. Di conseguenza, l'ellissoide di covarianza diventa

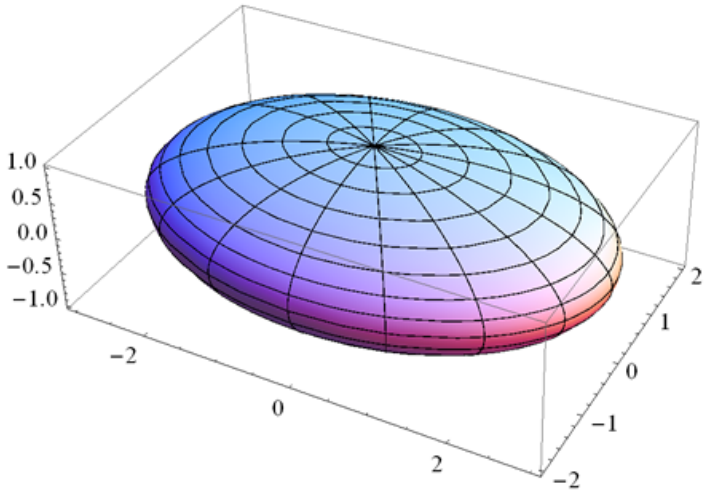



Figura 4.1: Raffigurazione di un ellissoide tridimensionale. Immagine presa da wikipedia.

una serie di ellissoidi, una per ogni variabile. Questi ellissoidi hanno i loro assi principali allineati con gli assi del sistema di coordinate.

I motivi che hanno spinto l'utilizzo di una matrice di covarianza diagonale piuttosto che una completa sono molteplici:

1. **Riduzione della complessità computazionale**: Una matrice di covarianza generale richiede il calcolo (molto oneroso in algoritmi come SGD) di tutte le covarianze tra le variabili, che crescono in modo quadratico rispetto al numero di variabili. Utilizzando una matrice di covarianza diagonale, tutte le covarianze tra le variabili sono nulle, riducendo di conseguenza la complessità computazionale.

2. **Riduzione del rischio di overfitting**: L'utilizzo di una matrice di covarianza diagonale assume l'assenza di correlazioni tra le variabili. Questo può essere vantaggioso per evitare problemi di overfitting, cioè eliminando le correlazioni, si riduce la possibilità che il modello impari relazioni spurie o ridondanti tra le variabili.

3. **Trattamento delle variabili indipendenti**: Molte classi nel dataset possono essere considerate indipendenti tra loro, senza alcuna relazione significativa. Utilizzando una matrice di covarianza diagonale, si assume questa indipendenza e si semplifica la modellazione delle variabili come caratteristiche indipendenti. L'unica relazione esistente in questo caso è quella della `subClassOf`.

4. L'ultimo ma principale motivo della scelta di una matrice di covarianza diagonale è l'**efficienza nella rappresentazione delle distribuzioni multivariate**: la matrice di covarianza diagonale può essere utilizzata per rappresentare distribuzioni multivariate in modo più semplice e spesso efficiente rispetto a una matrice di covarianza generale.

### 4.2.2 Distribuzione Normale Multivariata

La distribuzione normale multivariata (vedi fig. 4.2) è una distribuzione di probabilità che coinvolge più variabili casuali. La sua formula può

essere espressa come:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)} \tag{4.1}$$

oppure

$$\log f(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) - \frac{1}{2}\log|\mathbf{\Sigma}| - \frac{d}{2}\log(2\pi) \tag{4.2}$$

dove:

- $\mathbf{x}$ rappresenta un vettore casuale $d$-dimensionale, cioè $\mathbf{x} = [x_1, x_2, \ldots, x_d]$,
- $\mu$ è un vettore $d$-dimensionale che rappresenta il vettore delle medie delle variabili casuali,
- $\mathbf{\Sigma}$ è una matrice di covarianza $d \times d$ che rappresenta le covarianze tra le variabili casuali,
- $|\mathbf{\Sigma}|$ indica il determinante della matrice di covarianza $\mathbf{\Sigma}$,

La formula descrive la densità di probabilità $f(\mathbf{x})$ di ottenere il vettore casuale $\mathbf{x}$ dalla distribuzione normale multivariata. La densità di probabilità dipende dalle medie ($\mu$) e dalle covarianze ($\mathbf{\Sigma}$) delle variabili casuali. Un fattore di normalizzazione viene applicato per garantire che l'area sottesa dalla curva sia pari a 1.

In sintesi, l'innovativo metodo proposto da TransM per modellare le classi e le relazioni tra di esse richiede quindi:

1. un vettore $\mu$ per ogni classe di dimensione $d$;
2. una vettore $\mathbf{\Sigma}$ per ogni classe di dimensione $d$ (essendo una matrice diagonale si può salvare semplicemente un vettore);
3. una funzione di score per la relazione `typeOf`;
4. una funzione di score per la relazione `subClassOf`;

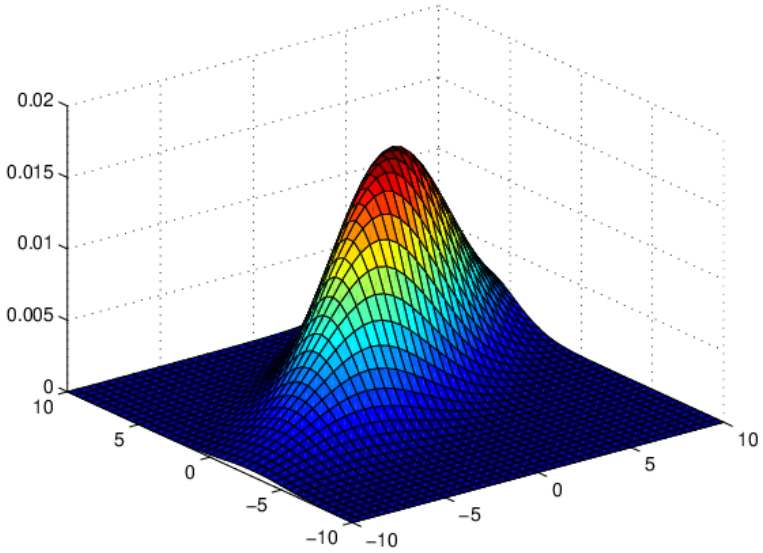



Figura 4.2: Raffigurazione geometrica della distribuzione normale multivariata.

### 4.2.3 Funzioni di Score

In TransM, come già accennato in precedenza, è necessario capire in che modo le relazioni `typeOf` e `subClassOf` vengono modellate attraverso la distribuzione normale multivariata (vedi 4.2.2).

#### Funzione di score typeOf

Dal punto di vista geometrico, lo scopo della funzione di score della relazione `typeOf`, data un'entità $h$ e una classe $c$, è quella di far inglobare l'entità $h$ dall'iperellissoide associato alla classe $c$. Ogni entità che ricade all'interno dell'iperellisoide sarà istanza di quella classe. Dal punto di vista probabilistico, la funzione di score della relazione `typeOf` può essere interpretata come una misura di probabilità o di densità per determinare se un'entità $h$ è un'istanza di una classe $c$. Nello specifico si è utilizzata la **log probabilità** per due motivi principali:

1. **Evita problemi di underflow**: nell'ambito di una distribuzione normale multivariata, la probabilità che un'entità sia associata a una classe specifica può teoricamente variare da 0 a 1. Tuttavia, cercare di ottenere un valore di probabilità alto è estremamente improbabile, soprattutto nel contesto di KG in cui si lavora con dimensioni degli embedding elevate. La probabilità massima associata a un'entità che appartiene a una classe specifica si verifica quando l'entità si

trova esattamente al centro della distribuzione (quando il vettore $x$ coincide con $\boldsymbol{\mu}$). In questo caso, la probabilità può raggiungere il valore massimo, che quasi sempre è un valore inferiore a $0,1$. Nei calcoli con numeri a virgola mobile, la moltiplicazione di numeri molto piccoli può portare a un underflow, ovvero alla perdita di precisione dovuta all'eccessiva approssimazione a zero e rendendo quindi difficoltoso il calcolo del gradiente da parte dell'ottimizzatore. Usando la log probabilità, i prodotti vengono trasformati in somme, evitando così problemi di underflow.

2. **Stabilizza l'ottimizzazione**: Durante l'addestramento di modelli probabilistici, l'ottimizzazione dei parametri può essere più stabile e più rapida utilizzando la log probabilità come funzione obiettivo. Ciò è dovuto alla natura concava della funzione logaritmo, che semplifica il processo di ottimizzazione.

La funzione di score della relazione `typeOf` è quindi:

$$f(\mathbf{h}, \mathbf{c}) = logf(\mathbf{h}) \tag{4.3}$$

dove:

- $\mathbf{h}$ è il vettore associato all'entità $h$.
- $\mathbf{c}$ è la struttura associata alla classe $c$ (vettore $\mu$ e vettore $\mathbf{\Sigma}$).
- $logf(\mathbf{h})$ è la formula della log probabilità della distribuzione normale multivariata, come spiegata nella formula 4.2.

### Funzione di Score subClassOf

Dal punto di vista geometrico, lo scopo della funzione di score della relazione `subClassOf`, data una classe $c_1$ e una classe $c_2$, è quella di far inglobare l'iperellissoide associato alla classe $c_1$ dall'iperellissoide associato alla classe $c_2$. Questa è una relazione antisimmetrica, in quanto se $c_1$ è sottoclasse di $c_2$ sicuramente il contrario sarà falso.

Tali motivazioni hanno motivato l'utilizzo della distanza di Mahalanobis, una misura di dissimilarità tra due vettori che tiene conto delle

correlazioni tra le variabili. Dati due vettori $x$ e $y$:

$$D(x, y) = \sqrt{(x - y)^T \cdot S^{-1} \cdot (x - y)}$$

dove $D(x, y)$ rappresenta la distanza di Mahalanobis tra i vettori $x$ e $y$, $(x - y)^T$ indica la trasposta della differenza tra $x$ e $y$, e $S^{-1}$ è la matrice inversa della matrice di covarianza $S$. Tuttavia, se la matrice di covarianza è diagonale, come in TransM, la formula si semplifica in:

$$D(x, y) = \sqrt{\sum_{i=1}^{n} \left( \frac{x_i - y_i}{\sigma_i} \right)^2}$$

Si immagini di avere un insieme di punti nello spazio a più dimensioni e si voglia determinare la probabilità che un nuovo punto appartenga a quell'insieme. L'idea intuitiva è che se il nuovo punto è vicino al centro di massa degli altri punti, è più probabile che appartenga all'insieme.

Inoltre, è importante considerare se l'insieme si estende su una piccola o grande distanza, in modo da poter valutare se una certa distanza dal centro è più o meno significativa. Un approccio semplice consiste nel calcolare la deviazione standard dei campioni rispetto al centro di massa. Se la distanza tra il nuovo punto e il centro di massa è inferiore a una deviazione standard, possiamo concludere che è molto probabile che il nuovo punto appartenga all'insieme. Al contrario, se la distanza è maggiore, è meno probabile che il punto appartenga all'insieme. Questo approccio si basa sull'assunzione che i campioni siano distribuiti all'interno di una sfera intorno al centro di massa. Tuttavia, se la distribuzione non è sferica (ad esempio ellissoidale), ci aspettiamo che la probabilità che il nuovo punto appartenga all'insieme dipenda non solo dalla distanza dal centro di massa, ma anche dalla direzione. In direzioni in cui l'ellissoide è più schiacciato, il nuovo punto deve essere più vicino per essere considerato appartenente all'insieme, mentre in direzioni in cui l'ellissoide è più allungato, il punto può trovarsi anche a distanze maggiori. La distanza di Mahalanobis, dunque, è semplicemente la distanza del punto in esame dal centro delle masse normalizzata rispetto all'ampiezza dell'ellissoide nella direzione del punto in esame.

Il problema della Mahalanobis che però non riesce a risolvere è la stima di quanto l'ellissoide $c_1$ venga inglobato dall'ellissoide $c_2$, informazione fondamentale nella relazione `subClassOf`. A questo scopo si è deciso di calcolare il volume dei due ellissoidi tramite la seguente formula:

$$\mathtt{V} = \frac{1}{2} \cdot n \cdot \log(2\pi) + \frac{1}{2} \cdot \sum_{i=1}^{n} \log(\Sigma)$$

Il calcolo del volume degli ellissoidi è basato sul determinante della matrice di covarianza e tiene conto delle deviazioni standard delle variabili rappresentate dagli elementi diagonali della matrice. Questo calcolo ci fornisce una misura della dimensione relativa degli ellissoidi associati agli insiemi di dati.

Grazie alle seguenti formule è stato possibile definire la funzione di score della relazione `subClassOf`:

$$f(\mathbf{c_1}, \mathbf{c_2}) = -\alpha D(c_1, c_2) - (V(c_1) - V(c_2)) \tag{4.4}$$

dove:

- $\alpha$ è un parametro positivo scelto in modo tale da avere entrambi gli addendi alla stessa scala (da notare che dipende dalla dimensione dell'embedding).
- $\mathbf{c_1}$ è la struttura associata alla classe $c1$ (vettore $\mu$ e vettore $\mathbf{\Sigma}$).
- $\mathbf{c_2}$ è la struttura associata alla classe $c2$ (vettore $\mu$ e vettore $\mathbf{\Sigma}$).
- $D(c_1, c_2)$ è la distanza di Mahalanobis calcolata da $c1$ a $c2$.
- $V(c_1)$ è il volume calcolato per la classe $c1$.
- $V(c_2)$ è il volume calcolato per la classe $c2$.

Ricapitolando, se $\mathbf{c_1}$ è sottoclasse di $\mathbf{c_2}$:

$$-\alpha D(c_1, c_2) - (V(c_1) - V(c_2))$$

Essendo $\mathbf{c_1}$ inglobato da $\mathbf{c_2}$ in quanto sua sottoclasse, il valore senza segno $\alpha D(c_1, c_2)$ sarà minore rispetto a $\alpha D(c_2, c_1)$ in quanto se il punto

$c_1$ (nello specifico $\mu$ di $c_1$) è all'interno dell'ellissoide di $c_2$, significa che si trova nelle regioni di alta probabilità della distribuzione dei dati. Poiché la distanza di Mahalanobis considera le informazioni sulla varianza e la correlazione delle variabili, la sua misura sarà minore quando il punto $c_1$ è all'interno dell'ellissoide, indicando una maggiore somiglianza. Esisterà quindi la seguente relazione:

$$0 < -\alpha D(c_2, c_1) < -\alpha D(c_1, c_2)$$

Oltre ciò, anche $-(V(c_1) - V(c_2))$ sarà maggiore rispetto a $-(V(c_2) - V(c_1))$, cioè esisterà la seguente relazione:

$$0 < -(V(c_2) - V(c_1)) < -(V(c_1) - V(c_2))$$

In questo caso, quindi, la probabilità sarà più alta se $c_1$ ha un volume minore rispetto a $c_2$ con $c_1$ sottoclasse di $c_2$. Ne segue pertanto che:

$$f(\mathbf{c_2}, \mathbf{c_1}) < f(\mathbf{c_1}, \mathbf{c_2}) < 0$$

Quindi, la probabilità associata alla tripla vera $(\mathbf{c_1}, \texttt{subClassOf}, \mathbf{c_2})$ sarà più alta rispetto alla tripla falsa $(\mathbf{c_2}, \texttt{subClassOf}, \mathbf{c_1})$ tenendo in considerazione sia distanza di Mahalanobis che differenza tra i volumi.

### 4.2.4 Funzione di Loss e Ottimizzatore

La funzione di loss $L$ di TransM è così definita:

$$\begin{aligned} L = &\lambda_1 \sum_{(h,r,t)\in\Delta} \sum_{(h',t)\in\Delta'} (\gamma + f_r(h,t) - f_r(h',t')) \\ &+ \lambda_2 \sum_{(h,\texttt{typeOf},t)\in\Delta} \sum_{(h',t)\in\Delta'} (\gamma + f_t(h,t) - f_t(h',t')) \\ &+ \lambda_3 \sum_{(h,\texttt{subClassOf},t)\in\Delta} \sum_{(h',t)\in\Delta'} (\gamma + f_s(h,t) - f_s(h',t')) \end{aligned} \tag{4.5}$$

dove $\gamma$ viene anche chiamato margine: quest'ultimo può essere interpretato come una misura di quanto i fatti previsti dal modello si allontanino dai fatti reali del knowledge graph. Minimizzare il margine aiuta

a migliorare l'accuratezza e la coerenza delle previsioni del modello, garantendo che le entità e le relazioni coinvolte in un fatto del knowledge graph siano rappresentate in modo coerente nello spazio vettoriale. I coefficienti $\lambda_1, \lambda_2$ e $\lambda_3$ sono coefficienti che regolano il peso di ciascuna funzione di loss. È stato utilizzato, inoltre, il regolarizzatore $L2$ per controllare l'overfitting e migliorare la capacità di generalizzazione del modello. Il regolarizzatore $L2$ introduce una penalità nella funzione di costo del modello in base alla norma $L2$ dei pesi. Applicando il regolarizzatore $L2$, i pesi del modello vengono penalizzati proporzionalmente al loro valore quadratico. L'ottimizzazione dei parametri in TransM richiede un'attenta analisi, tra cui soprattutto i coefficienti di regolarizzazione $\lambda_1, \lambda_2$ e $\lambda_3$. A questo scopo è stata implementata la grid search per determinare la miglior combinazione di essi.

L'ottimizzatore scelto in TransM è Adagrad [DHS11]. Il motivo di ciò è da attribuire all'obiettivo principale che Adagrad cerca di risolvere, cioè il problema dell'adattamento del tasso di apprendimento (learning rate) per ciascun parametro del modello. In molti casi, i gradienti dei diversi parametri possono variare significativamente in termini di scala, portando a una convergenza lenta o instabile (caso molto probabile in TransM). Adagrad risolve questo problema aggiornando il tasso di apprendimento in modo adattivo per ogni parametro, tenendo conto della storia dei gradienti passati. L'algoritmo Adagrad calcola e mantiene una media accumulativa dei gradienti al quadrato per ciascun parametro durante l'addestramento. Questa media accumulativa viene utilizzata per adattare il tasso di apprendimento per ogni parametro nel corso del tempo. Durante l'aggiornamento dei pesi, Adagrad divide il tasso di apprendimento per la radice quadrata della somma dei gradienti al quadrato accumulati per il parametro corrispondente. Questo significa che i parametri che hanno avuto gradienti maggiori nel corso dell'addestramento avranno un tasso di apprendimento ridotto, mentre i parametri con gradienti minori avranno un tasso di apprendimento aumentato. In questo modo, Adagrad si adatta automaticamente alla scala dei gradienti per ogni parametro, consentendo una convergenza più rapida e stabile.

# Capitolo 5

# Valutazione Sperimentale

## 5.1 Triple Classification

Il task di Triple Classification giudica se una tripla è corretta o meno. Questa può essere una tripla relazionale, una tripla `typeOf` oppure una tripla `subClassOf` (come è stato valutato nel lavoro di TransC [Lv+18], ma non in questa tesi). Per questo task sono necessarie anche le triple negative, pertanto oltre al set contenente le triple vere, è stato costruito per ogni dataset un set contenente le triple negative. Nel caso di TransC [Lv+18], TransC-OWL e TransM, viene impostato per ciascuna relazione un threshold $\delta$r. Quindi, data una tripla appartenente al test set, se la funzione di loss calcolata su di essa è minore di $\delta$r verrà classificata come una tripla positiva, altrimenti come una tripla negativa. Il parametro $\delta$r viene ottenuto minimizzando l'FPR sul valid set.

Sono stati addestrati i modelli TransC, TransC-OWL e TransM su diversi dataset ([Fär+18]) e, valutando i risultati in diverse epoche (a causa di problemi di overfitting), si è scelto di inserire solamente i risultati migliori avuti in una tra le epoche.

Nella classificazione, potremo distinguere le seguenti tipologie di triple:

- **TP - True Positive**. Triple che rappresentano conoscenza veritiera e che sono state classificate correttamente come vere.
- **TN - True Negative**. Triple che rappresentano conoscenza falsa e che sono state classificate correttamente come false.

- **FP - False Positive**. Triple che rappresentano conoscenza falsa e che sono state classificate erroneamente come vere.
- **FN - False Negative**. Triple che rappresentano conoscenza veritiera e che sono state classificate erroneamente come false.

Mediante il conteggio di queste quantità, sarà possibile calcolare le seguenti metriche impiegate per valutare le prestazioni dei modelli:

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FN+FP}}$$

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

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

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Eseguendo una valutazione complessiva, TransC[Lv+18] mantiene performance molto alte in quasi ogni dataset sperimentato nel task di triple classification `typeOf` grazie all'innovativa rappresentazione dei concetti nei suoi vettori di embedding. Inoltre, la variante TransC-OWL è riuscita a migliorare in alcuni casi le prestazioni di TransC[Lv+18] nelle triple `typeOf` grazie all'iniezione di conoscenza a discapito di un leggero peggioramento nel task di triple classification `non typeOf`. Il modello innovativo TransM, invece, risulta essere generalmente il migliore anche rispetto ai modelli precedenti (TransC e TransC-OWL inclusi).

### 5.1.1 Nell

Il dataset su cui si sono avuti i migliori risultati è stato Nell[1]. Nell è un dataset costruito dalla Carnegie Mellon University ricercando direttamente all'interno del web tramite un agente intelligente chiamato **N**ever-**E**nding **L**anguage **L**earner. Lo scopo principale di Nell è quindi quello di estrarre conoscenza partendo da informazioni non strutturate (come sono le pagine web) ed inoltre, una volta fatto ciò, migliorare la comprensione delle informazioni estratte in modo tale da estrarre maggior conoscenza.
In questo lavoro di tesi, in realtà, è stato utilizzato un sottoinsieme di Nell (a causa della grossa dimensione del dataset Nell), ovvero NELL2RDF-vanilla [2] il quale non contiene tutte le proprietà che i modelli sviluppati possono sfruttare. In particolare, Nell contiene un'abbondante numero di triple `subClassOf` per ciascun concetto e un ridotto numero di triple `typeOf` per ciascuna istanza (tabella 5.1).

Prendendo in considerazione le triple `non typeOf` (tabella 5.2), le prestazioni avute sono molto simili agli altri modelli tranne che nel recall in cui risulta essere il peggiore tra tutti.
Prendendo in considerazione, invece, le triple `typeOf`, il modello TransC-OWL risulta essere il secondo migliore in tutte le metriche dopo TransM. Questi due modelli beneficiano particolarmente da questo dataset in quanto ha un'abbondante presenza di assiomi `subClassOf`,

[1] http://rtw.ml.cmu.edu/rtw/

[2] http://nell-ld.telecom-st-etienne.fr/

| **Triple** | **Numero** |
|---|---|
| #Instance | 68620 |
| #Concept | 1187 |
| #Relation | 272 |
| #Train (Relational Triple) | 1,13[3] |
| #Train (`typeOf` Triple) | 0,57[4] |
| #Train (`subClassOf` Triple) | 1,06[5] |

Tabella 5.1: Numero di triple per il dataset NELL

| Nell | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | No typeOf | | | | typeOf | | | |
| **Modello** | **Acc.** | **F1** | **P** | **R** | **Acc.** | **F1** | **P** | **R** |
| TransE | 0.733 | 0.720 | 0.755 | 0.691 | 0.626 | 0.422 | 0.276 | 0.900 |
| TransOWL | 0.675 | 0.673 | 0.677 | 0.671 | 0.819 | 0.506 | 0.430 | 0.615 |
| TransR | 0.758 | 0.730 | 0.843 | 0.644 | 0.803 | 0.417 | 0.452 | 0.388 |
| TransROWL | 0.745 | 0.720 | 0.835 | 0.633 | 0.763 | 0.488 | 0.334 | 0.911 |
| TransROWL$^R$ | 0.743 | 0.715 | 0.845 | 0.621 | 0.760 | 0.474 | 0.337 | 0.804 |
| TransROWL-HRS | 0.749 | 0.736 | 0.847 | 0.651 | 0.823 | 0.567 | 0.421 | 0.871 |
| TransC | **0.845** | **0.822** | 0.966 | **0.716** | 0.596 | 0.691 | 0.559 | 0.908 |
| TransC-OWL | 0.668 | 0.565 | 0.820 | 0.432 | 0.672 | 0.620 | 0.735 | 0.537 |
| TransM | 0.777 | 0.808 | **0.991** | 0.682 | **0.981** | **0.979** | **0.984** | **0.993** |

Tabella 5.2: Risultati ottenuti nel task di Triple Classification per il dataset NELL

unita a un ridotto numero di triple `typeOf`. A differenza degli altri modelli che hanno messo in evidenza un forte limite di apprendimento (con una precision estremamente bassa), TransM in particolare riesce a sopperire all'incompletezza delle triple `typeOf` del training set sfruttando la conoscenza fornita dalle triple `subClassOf`.

### 5.1.2 DBpedia

DBpedia[Leh+15] è un progetto nato nel 2007 con lo scopo di estrarre informazioni strutturate da Wikipedia e pubblicarle sul Web come Linked Open Data in formato RDF. Il primo set di dati disponibile pubblicamente è stato pubblicato nel 2007. Gli articoli di Wikipedia sono costituiti principalmente da testo libero, ma includono anche informazioni strutturate incorporate negli articoli. Queste informazioni strutturate vengono estratte e inserite in un set di dati uniforme che può essere interrogato. La versione 2016-04 del set di dati DBpedia descrive 6,0 milioni di entità, di cui 5,2 milioni sono classificate in un'ontologia coerente, inclusi 1,5 milioni di persone, 810.000 posti, 135.000 album musicali, 106.000 film, 20.000 videogiochi, 275.000 organizzazioni, 301.000 specie e 5.000 malattie. DBpedia utilizza il Resource Description Framework (RDF) per rappresentare le informazioni estratte e consiste di 9,5 miliardi di triple RDF, di cui 1,3 miliardi sono stati estratti dall'edizione inglese di Wikipedia e 5,0 miliardi da edizioni in altre lingue. Il linguaggio di mappatura DBpedia è stato sviluppato per aiutare a mappare le relazioni su un'ontologia in modo tale da ridurre il numero di sinonimi.

A causa della grande diversità di relazioni in uso su Wikipedia, il processo di sviluppo e miglioramento di queste mappature è stato aperto ai contributi pubblici. Per esigenze computazionali, sono stati considerati due sottoinsiemi di DBpedia denominati DBpedia100K[Din+18] e DBpedia15K[Liu+19].

Nel caso di DBpedia15K in cui, a differenza del dataset Nell, è presente un abbondante numero di triple `typeOf` nel training set (tabella 5.3), ogni modello riesce ad avere buone performance in quasi tutte le metriche (tabella 5.4). Infatti, lo scopo di DBpedia15K è quello di verificare il comportamento del modello in condizioni normali. Nonostante ciò, ad esclusione di TransC e TransC-OWL, i restanti modelli hanno scarse prestazioni nella recall delle triple non `typeOf` (così come anche nel dataset DBpediaYAGO). Per quanto riguarda il task di triple classification `typeOf`, TransC-OWL è superiore in ogni metrica rispetto a TransC. Anche TransM riesce ad avere prestazioni molto buone in ogni metrica, risultando il migliore nell'accuracy delle triple `non typeOf`.

Nel caso di DBpedia100K il discorso sul triple classification `non typeOf` è analogo rispetto a DBpedia15K. Per quanto riguarda il task di triple classification `typeOf` (tabella 5.5) è leggermente differente rispetto a DBpedia15K. Qui infatti, il numero medio di relazioni `typeOf` per ogni istanza nel training set è circa un quarto rispetto a DBpedia15K. Questo rende più difficoltoso il task di triple classification `typeOf` negli altri modelli ad esclusione di TransC, TransC-OWL e TransM. Le performance avute in questi tre modelli, in questo caso, sono molto simili tra loro e non c'è un miglioramento netto di TransC-OWL rispetto a TransC (come avvenuto ad esempio in Nell e DBpedia15K). Una possibile spiegazione è data dalla minor conoscenza esterna disponibile

| **Triple** | **DBpedia15K** | **DBpedia100K** |
|---|---|---|
| #Instance | 15258 | 100001 |
| #Concept | 1179 | 1179 |
| #Relation | 278 | 322 |
| #Train (Relational Triple)[6] | 3,95 | 3,1 |
| #Train (`typeOf` Triple)[7] | 4,45 | 1,09 |
| #Train (`subClassOf` Triple)[8] | 8,37 | 8,37 |

Tabella 5.3: Numero di triple per il dataset DBpedia15K e DBpedia100K

| DBpedia15K | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | No typeOf | | | | typeOf | | | |
| **Modello** | **Acc.** | **F1** | **P** | **R** | **Acc.** | **F1** | **P** | **R** |
| TransE | 0.663 | 0.577 | 0.991 | 0.407 | 0.899 | 0.860 | 0.781 | 0.958 |
| TransOWL | 0.658 | 0.572 | 0.967 | 0.407 | 0.975 | 0.960 | 0.990 | 0.933 |
| TransR | 0.655 | 0.560 | **0.998** | 0.390 | 0.978 | 0.966 | 0.979 | 0.954 |
| TransROWL | 0.652 | 0.555 | 0.997 | 0.385 | 0.985 | **0.992** | **0.999** | **0.987** |
| TransROWL$^R$ | 0.648 | 0.547 | **0.998** | 0.377 | 0.981 | 0.970 | 0.969 | 0.972 |
| TransROWL-HRS | 0.634 | 0.521 | **0.998** | 0.353 | **0.988** | 0.987 | 0.997 | 0.978 |
| TransC | 0.696 | 0.816 | 0.722 | 0.940 | 0.942 | 0.942 | 0.937 | 0.948 |
| TransC-OWL | 0.698 | **0.818** | 0.722 | **0.946** | 0.967 | 0.966 | 0.976 | 0.957 |
| TransM | **0.712** | 0.761 | 0.945 | 0.638 | 0.953 | 0.974 | 0.965 | 0.984 |

Tabella 5.4: Risultati ottenuti nel task di Triple Classification per il dataset DBpedia15K

in DBpedia100K: ad esempio le triple `inverseOf` in DBpedia100K sono mancanti, mentre ad esempio in Nell erano abbondanti. Questo non ha permesso di avere gli stessi miglioramenti avuti nel task di triple classification `typeOf`. In questo caso TransM risulta il migliore nella recall delle triple `typeOf`.

### 5.1.3 DBpediaYago

YAGO[9] (**Y**et **A**nother **G**reat **O**ntology) è un ontologia open source estratta automaticamente da fonti come Wikipedia. YAGO è stato impiegato per estendere e rendere più completa la conoscenza descritta in

[9]https://yago-knowledge.org/

| DBpedia100K | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | No typeOf | | | | typeOf | | | |
| **Modello** | **Acc.** | **F1** | **P** | **R** | **Acc.** | **F1** | **P** | **R** |
| TransE | 0.742 | 0.560 | 0.993 | 0.390 | 0.958 | 0.781 | 0.667 | 0.943 |
| TransOWL | 0.714 | 0.513 | 0.901 | 0.359 | 0.980 | 0.869 | 0.908 | 0.835 |
| TransR | 0.721 | 0.528 | **0.998** | 0.359 | 0.983 | 0.899 | 0.890 | 0.910 |
| TransROWL | 0.744 | 0.562 | **0.998** | 0.392 | **0.987** | 0.928 | 0.965 | 0.895 |
| TransROWL$^R$ | 0.732 | 0.533 | **0.998** | 0.364 | 0.981 | 0.943 | 0.951 | 0.936 |
| TransROWL-HRS | 0.714 | 0.484 | **0.998** | 0.320 | 0.985 | 0.921 | 0.900 | 0.945 |
| TransC | **0.797** | **0.796** | 0.979 | **0.672** | 0.960 | **0.958** | **0.984** | 0.935 |
| TransC-OWL | 0.746 | 0.734 | 0.966 | 0.592 | 0.956 | 0.954 | 0.981 | 0.930 |
| TransM | 0.735 | 0.716 | 0.973 | 0.567 | 0.944 | 0.950 | 0.910 | **0.993** |

Tabella 5.5: Risultati ottenuti nel task di Triple Classification per il dataset DBpedia100K

| Triple | Numero |
|---|---|
| #Instance | 81694 |
| #Concept | 248799 |
| #Relation | 316 |
| #Train (Relational Triple) | 1,55[10] |
| #Train (`typeOf` Triple) | 0,90[11] |
| #Train (`subClassOf` Triple) | 1,00[12] |

Tabella 5.6: Numero di triple per il dataset DBpediaYAGO

| DBpediaYago | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | No typeOf | | | | typeOf | | | |
| **Modello** | **Acc.** | **F1** | **P** | **R** | **Acc.** | **F1** | **P** | **R** |
| TransE | 0.654 | 0.582 | 0.914 | 0.428 | **0.962** | 0.900 | 0.969 | 0.841 |
| TransOWL | 0.692 | 0.589 | 0.887 | 0.441 | 0.931 | 0.801 | 0.961 | 0.688 |
| TransR | 0.644 | 0.457 | 0.964 | 0.300 | 0.844 | 0.391 | 0.946 | 0.247 |
| TransROWL | 0.649 | 0.466 | 0.968 | 0.307 | 0.905 | 0.700 | 0.973 | 0.547 |
| TransROWL$^R$ | 0.636 | 0.432 | **0.981** | 0.277 | 0.854 | 0.455 | 0.953 | 0.299 |
| TransROWL-HRS | 0.647 | 0.461 | 0.975 | 0.302 | 0.950 | 0.861 | **0.986** | 0.765 |
| TransC | 0.655 | 0.674 | 0.946 | 0.524 | 0.934 | **0.931** | 0.969 | 0.896 |
| TransC-OWL | 0.592 | 0.581 | 0.972 | 0.415 | 0.923 | 0.922 | 0.939 | 0.907 |
| TransM | **0.744** | **0.786** | 0.862 | **0.724** | 0.930 | 0.929 | 0.951 | **0.909** |

Tabella 5.7: Risultati ottenuti nel task di Triple Classification per il dataset DBpediaYAGO

DBpedia15K e sfruttando i numerosi link che collegano i dataset DBpedia [Leh+15] e YAGO tra loro è stato possibile realizzare un nuovo dataset, denominato DBpediaYAGO. Le prestazioni di un modello risentono particolarmente del numero di triple su cui hanno la possibilità di addestrarsi. Alla luce di tale fenomeno, mediante DBpediaYAGO, sarà possibile verificare quanto effettivamente i modelli sviluppati, e i loro miglioramenti in termini prestazionali, risentano di un KG caratterizzato da maggiore completezza o meno.

Nel caso di un dataset affetto da minore incompletezza (Tab. 5.6) come DBpediaYAGO, i risultati sono mediamente buoni in ogni modello. In particolare, per quanto riguarda il task di triple classification `no typeOf` (tabella 5.7) il discorso è analogo rispetto a DBpedia15K e DBpedia100K; TransM riesce in generale ad avere le prestazioni migliori in molte metriche (soprattutto `no typeOf`).

## 5.2 Estrazione di Regole

In molte applicazioni dei knowledge graph (specialmente in ambito medico), è importante avere una comprensione di come i concetti si relazionano tra di loro in modo tale da dare una spiegazione delle predizioni. Per far ciò è necessario associare una logica al modello utilizzato per poi poter rispondere alle query dell'utente. Tuttavia, in relazione ai knowledge graph embeddings (come il modello TransC-OWL), l'explainability ha una difficile interpretazione, ovvero: mentre i knowledge graph sono in genere direttamente interpretabili in termini di relazioni dirette tra le entità, gli embeddings perdono questa proprietà a causa dello spazio dei vettori a bassa dimensionalità in cui lavorano ([Bia+20]). L'explainability passa quindi dalla definizione di metodi che supportano il ragionamento mediante la logica, proprio perché quest'ultima offre un paradigma che sostiene il ragionamento e le sue inferenze sono giustificabili e verificabili utilizzando assiomi logici.

Lo scopo è quindi quello di proporre dei modelli di embeddings che siano in grado di modellare gli assiomi logici all'interno dello spazio vettoriale. Tuttavia, gli approcci più basilari, come TransE non sono in grado ad esempio di modellare la relazione di simmetria a causa della funzione di score scelta. I modelli TransM e TransC-OWL sono in grado di rappresentare, sfruttando il modello traslazionale derivante da TransE, le relazioni inverse, antisimmetriche e composte. Grazie invece alla geometria associata alle classi (ipersfere ed iperellissoidi), è possibile estrarre la relazione transitiva di `typeOf` e `subClassOf` oltre che regole di disgiunzione tra classi. Un difetto di TransE e derivati è l'impossibilità di estrarre relazioni fondamentali per l'interpretazione di una predizione come quelle simmetriche, indicata in lavori quali [Tro+16; TC15; GML15] come una tra le tre più importanti insieme alla relazione inversa e composta.

Le metriche utilizzate per la valutazione delle regole estratte sono diverse rispetto a quelle utilizzate per la classificazione (anche se poi nella sostanza sono molto simili).

Data una regola $X \implies Y$:

1. **Supporto**: Il supporto è un'indicazione di quanto frequentemente un itemset appare nel dataset [WWT99], dove un itemset sono i valori assunti da $X$ e $Y$. Il supporto è dunque:

$$\text{support}(X \implies Y) = P(X \cap Y) = \frac{\#\{triple\ con\ X\ e\ Y\}}{\#\{triple\ totali\}}$$

2. **Confidenza**: La confidenza è la percentuale delle triple che soddisfano $X$ e anche $Y$ [WWT99]. È pertanto il rapporto tra le triple che contengono sia $X$ che $Y$ sul totale di triple che contengono X. La confidenza, in realtà, può essere interpretata come una stima della probabilità condizionata $P[Y|X]$. Quindi:

$$\begin{aligned} \text{conf}(X \implies Y) &= P[Y|X] \\ &= \frac{\text{support}(X \cap Y)}{\text{support}(X)} = \frac{\#\{triple\ con\ X\ e\ Y\}}{\#\{triple\ con\ X\}} \end{aligned}$$

3. **Lift**: Il lift è una misura utilizzata nelle regole di associazione per determinare la forza della relazione tra due oggetti in un dataset. In particolare, il lift misura quanto più probabile è l'occorrenza di entrambi gli oggetti rispetto a quanto ci si aspetterebbe se fossero indipendenti l'uno dall'altro. Il lift è dunque un modo per valutare l'importanza di una regola di associazione e può servire per identificare le relazioni significative. La formula è dunque:

$$\text{lift}(X \implies Y) = \frac{\text{support}(X \cap Y)}{\text{support}(X) \cdot \text{support}(Y)}$$

4. **P(X → Y)**: È la misura di quanto è probabile che l'implicazione sia vera. Se sappiamo che $Y$ è vero, allora l'implicazione è vera, indipendentemente dalla probabilità di $X$. In questo caso, la probabilità dell'implicazione è 1. Se sappiamo che $Y$ è falso, allora l'implicazione è vera solo se $X$ è falsa. In questo caso, la probabilità dell'implicazione è 0. Tuttavia, se non sappiamo se $Y$ è vero o falso,

allora la probabilità dell'implicazione dipenderà dalla probabilità a priori di $X$ e $Y$, e dal loro legame causale.

### 5.2.1 Relazioni Simmetriche e Antisimmetriche

Una relazione r è **simmetrica** se $\forall x, y$:

$$r(x, y) \Rightarrow r(y, x)$$

Una relazione r è **antisimmetrica** se $\forall x, y$:

$$r(x, y) \Rightarrow \neg r(y, x)$$

I modello TransC-OWL e TransM, come già spiegato, sono in grado di apprendere solamente le relazioni antisimmetriche e non simmetriche a causa della identica rappresentazione di TransE nel rappresentare entità e relazioni. Ad esempio, se si ha una evidente relazione simmetrica come `sposato_con` e due individui come `Mamma` e `Papà` le cui due triple (`Mamma`, `sposato_con`, `Papà`) e (`Papà`, `sposato_con`, `Mamma`) sono entrambe vere, a causa della funzione di score di TransC-OWL e TransM, che è la medesima di TransE nel caso di triple relazionali, si avrà che:

1. (`Mamma, sposato_con, Papà`) vera quindi:

$$\texttt{Mamma} + \texttt{sposato\_con} \approx \texttt{Papà}$$

2. (`Papà, sposato_con, Mamma`) vera quindi:

$$\texttt{Papà} + \texttt{sposato\_con} \approx \texttt{Mamma}$$

Tuttavia questo provoca un assurdo perché, sostituendo ad esempio il vettore `Mamma` della prima equazione con la seconda equazione si avrebbe che:

$$\texttt{Papà} + \texttt{sposato\_con} + \texttt{sposato\_con} \approx \texttt{Papà}$$

$$\texttt{Papà} + \texttt{sposato\_con} + \texttt{sposato\_con} - \texttt{Papà} \approx 0$$

$$\texttt{sposato\_con} + \texttt{sposato\_con} \approx 0$$

$$\texttt{sposato\_con} \approx -\texttt{sposato\_con}$$

Quindi questo significa che la relazione simmetrica in TransC-OWL, TransM e in TransE è possibile impararla solo se il vettore relazione è uguale al suo inverso e cioè solo se il vettore è nullo, il che renderebbe inutile la rappresentazione stessa della relazione in quanto non sarebbe in grado di apprendere nulla.

Diversamente è il caso di una relazione antisimmetrica, che invece è possibile impararla. Ad esempio se si ha una relazione antisimmetrica come `figlio_di` e due individui come `Nonno` e `Mamma` la cui tripla (`Mamma`, `figlio_di`, `Nonno`) e (`Nonno, figlio_di, Mamma`) falsa, si avrà che:

1. (`Mamma, figlio_di, Nonno`) vera quindi:

$$\texttt{Mamma} + \texttt{figlio\_di} \approx \texttt{Nonno}$$

2. (`Nonno, figlio_di, Mamma`) falsa quindi:

$$\texttt{Nonno} + \texttt{figlio\_di} \not\approx \texttt{Mamma}$$

In questo caso la situazione è diversa perché se si addiziona il vettore `figlio_di` al vettore `Mamma`, il vettore risultante sarà vicino al vettore `Nonno`, come aspettato, però se si addiziona lo stesso vettore `figlio_di` al vettore `Nonno`, questo non sarà vicino al vettore `Mamma` il che non provoca problemi come nel caso precedente della simmetria e quindi questo conclude che la relazione antisimmetrica è imparabile in TransE e derivati come TransC-OWL e TransM.
Per verificare quali fossero quindi le relazioni antisimmetriche (o quelle relazioni che più si avvicinano ad esserlo) si è proceduti eseguendo la somma tra il vettore relazione con se stesso (ad esempio $r_1 + r_1$) e calcolando la distanza tra il vettore risultate e il vettore nullo: maggiore

| **Dataset** | **Relazione antisimmetrica** |
|---|---|
| DBpediaYago | `datore`<br>`colonna_sonora`<br>`fondazione`<br>`città`<br>`luogo_di_riposo`<br>... |
| DBpedia15K | `ex_squadra`<br>`parlato_in`<br>`lingua`<br>`etnia`<br>... |
| DBpedia100K | `santuario_maggiore`<br>`progetto_significante`<br>`luogo_di_riposo`<br>`creatore`<br>`luogo_di_nascita`<br>... |
| Nell | `ha_sede_nel_paese`<br>`stazione_televisiva_in_città`<br>`vestiti_realizzati_da_piante`<br>`persona_nata_in_città`<br>`suona_strumento`<br>... |

Tabella 5.8: Risultati ottenuti nell'estrazione di regole antisimmetriche

sarà la distanza e maggiore sarà la probabilità che la relazione sia antisimmetrica. Sono state prese in considerazione solamente le relazioni che hanno riportato le distanze maggiori.

Le relazioni trovate sono evidentemente antisimmetriche (vedere tabella 5.8), ad esempio prendendo la relazione `colonna_sonora`:

$$\forall x, y \ \texttt{colonna\_sonora}(x, y) \Rightarrow \neg \texttt{colonna\_sonora}(y, x)$$

ovvero, ad esempio:

$$(\texttt{La maledizione della prima luna, colonna\_sonora, Pirati dei caraibi}) \ vera$$

$$(\texttt{Pirati dei Caraibi, colonna\_sonora, La maledizione della prima luna}) \ falsa$$

| **N. antisimmetriche totali** | **N. antisimmetriche trovate** | **Probabilità** $P[E] = \frac{trovate}{tot.}$ | **Probabilità** $P[E] = \frac{\sum_{n=1}^{trovate} dist_n}{trovate}$ |
|---|---|---|---|
| 240 | 206 | 85,8% | 89,0% |
| 40 | 36 | 90,0% | 93,7% |

Tabella 5.9: Valutazione delle regole antisimmetriche sul dataset Nell

Sono state valutate, inoltre, le relazioni antisimmetriche sul dataset Nell in quanto l'ontologia era l'unica che metteva a disposizione il costrutto `ASymmetricProperty`. Su un totale di 270 relazioni, ben 240 sono antisimmetriche, quindi si è proceduti in questa maniera:

1. Sono state ordinate in ordine decrescente le relazioni in base alla distanza con il vettore nullo (come spiegato sopra);
2. È stato valutato prima il comportamento su tutte le 240 relazioni antisimmetriche;
3. Dopodiché è stato valutato il comportamento sulle relazioni che nel training set avevano il maggior numero di esempi per capire meglio come il modello si comportasse quando la conoscenza è abbondante (su un totale di 40 relazioni antisimmetriche);

Sul totale di 240 relazioni antisimmetriche ben 206 sono state trovate (circa l'86%), ovvero: dopo aver ordinato in ordine decrescente le relazioni in base alla distanza con il vettore nullo si è tagliato esattamente alla relazione numero 240 in modo tale da controllare quante tra le prime 240 "trovate" come vere dal modello fossero realmente esatte. Questa metodologia in realtà è simile alla metrica Hits@k. Si è voluto anche verificare la probabilità che il modello sia in grado di trovare una relazione antisimmetrica esatta in base alla distanza con il vettore nullo, cioè:

1. Come prima cosa è stato applicato il min max scaler sulle distanze associate a ciascuna relazione in modo tale da avere un range compreso tra 0 e 1, dove 0 significa che la relazione è identica al vettore nullo mentre 1 è il massimo tra tutte le distanze delle relazioni;

2. È stata calcolata la probabilità effettuando la media tra le distanze ($dist_n$ nella formula) delle relazioni realmente antisimmetriche.

Sul totale di 40 relazioni antisimmetriche (meglio imparate dal modello perché addestrate su più esempi) invece ben 36 sono state trovate (90%) con una probabilità di circa 94% se si considerano invece le distanze con il vettore nullo.

### 5.2.2 Relazioni Inverse

Una relazione $r_1$ è **inversa** a $r_2$ se $\forall x, y$:

$$r_2(x, y) \Rightarrow r_1(y, x)$$

Il modello TransE e derivati sono in grado di imparare le relazioni inverse. Ad esempio se si hanno due relazione inverse come `figlio_di` e `genitore_di` e due individui come `Genitore` e `Figlio` si avrà che:

1. (`Figlio`, `figlio_di`, `Genitore`) vera quindi:

$$\texttt{Figlio} + \texttt{figlio\_di} \approx \texttt{Genitore}$$

2. (`Genitore`, `genitore_di`, `Figlio`) vera quindi:

$$\texttt{Genitore} + \texttt{genitore\_di} \approx \texttt{Figlio}$$

Molto similmente al caso della relazione simmetrica si ha che:

$$\texttt{Figlio} + \texttt{figlio\_di} + \texttt{genitore\_di} \approx \texttt{Figlio}$$

$$\cancel{\texttt{Figlio} +} \texttt{figlio\_di} + \texttt{genitore\_di} \cancel{- \texttt{Figlio}} \approx 0$$

$$\texttt{figlio\_di} + \texttt{genitore\_di} \approx 0$$

$$\texttt{figlio\_di} \approx -\texttt{genitore\_di}$$

Quindi, affinché le due relazioni siano inverse, anche i vettori ad esse associate devono essere inversi rispetto all'addizione. È importante no-

| Dataset | Regola | Prob. |
|---|---|---|
| DBpediaYAGO | `influenzato` ≡ `influenzato_da`$^-$<br>`mappa` ≡ `pittura`$^-$ | >95%<br>38% |
| DBpedia15K | `influenzato` ≡ `influenzato_da`$^-$ | >95% |
| DBpedia100K | - | - |
| Nell | - | - |

Tabella 5.10: Risultati ottenuti nell'estrazione di regole inverse solo per TransC-OWL

tare che se si dimostra che $r_1$ è inversa rispetto a $r_2$, non c'è bisogno di dimostrare anche il viceversa perché una relazione inversa è ovviamente anche simmetrica.

Per verificare quali fossero quindi le relazioni inverse (o quelle relazioni che più si avvicinano ad esserlo) si è proceduti esattamente eseguendo la somma tra i due vettori relazione e calcolando la distanza tra il vettore risultate e il vettore nullo (esattamente come nel caso della relazione antisimmetrica): minore sarà la distanza e maggiore sarà la probabilità che i due vettori siano tra di loro inversi. Sono state prese in considerazione solamente le relazioni che hanno riportato una distanza minore rispetto a una certa soglia $\delta r = 2$. La probabilità che esse siano inverse è stata calcolata come:

$$P = (\delta r - dist)\frac{100}{\delta r}$$

in modo tale che una distanza pari a 0 venga etichettata come probabilità certa (100%), mentre una distanza pari a 2 è stata etichettata come 0%. Solamente due relazioni hanno avuto un probabilità maggiore al 50% (che è l'unica esatta), quindi le altre sono state scartate.

In Tab. 5.10 le relazioni inverse selezionate con le relative metriche.

Solamente nel dataset DBpediaYAGO e DBpedia15K era presente una vera e propria relazione inversa, ovvero la `influenzato` con la `influenzato_da` che il modello TransC-OWL in particolare è stato in grado di riconoscere con un'alta affidabilità (che tendeva sempre a crescere con l'aumentare del numero di epoche, fino ad arrivare ad una probabilità vicina al 99% rischiando però di far andare il modello in overfitting in altre relazioni). I modelli TransC e TransM, inve-

ce, non sono stati in grado di apprendere la relazione inversa che c'è tra `influenzato` e `influenzato_da` a causa della mancata iniezione di conoscenza delle relazioni `inverseOf` che invece è stata effettuata in TransC-OWL.

### 5.2.3 Disgiunzione tra Classi

Una classe $c_1$ è **disgiunta** con $c_2$ se $\forall x$ individui:

$$(x, \texttt{typeOf}, c_1) \Rightarrow \neg(x, \texttt{typeOf}, c_2)$$

$$c_1 \sqcap c_2 = \bot, c_1 \neq c2$$

e viceversa, ovvero $\forall x$ individui:

$$(x, \texttt{typeOf}, c_2) \Rightarrow \neg(x, \texttt{typeOf}, c_1)$$

$$c_2 \sqcap c_1 = \bot, c_2 \neq c1$$

In sostanza, due classi sono disgiunte tra di loro se nessuna istanza tra le due è in comune, pertanto la loro intersezione sarà vuota. Questa relazione di disgiunzione è più interessante rispetto alla relazione inversa, perché sicuramente sono assiomi maggiormente presenti all'interno dei vari dataset ed anche perché è una relazione che spesso non viene aggiunta all'interno delle ontologie perché data come scontata, provocando alcune volte conclusioni errate da parte delle macchine e se aggiunte potrebbe ridurre notevolmente le derivazioni logiche. Un metodo proposto in [RdF21; Riz+17] consiste nel derivare gli assiomi di disgiunzione partendo direttamente dalle ontologie sfruttando l'induzione di cluster tree terminologici, cioè alberi logici in cui ogni nodo è un cluster di individui che rappresentano un sotto-concetto. La crescita di questi alberi si basa sulla tecnica divide-et-impera che assegna alla radice uno dei concetti selezionati mediante una euristica basata sulla minimizzazione del rischio di intersezione tra i sotto-concetti.
Il metodo proposto in questo lavoro di tesi, invece, sfrutta direttamente l'embedding (e nello specifico la geometria) risultante dall'addestramen-

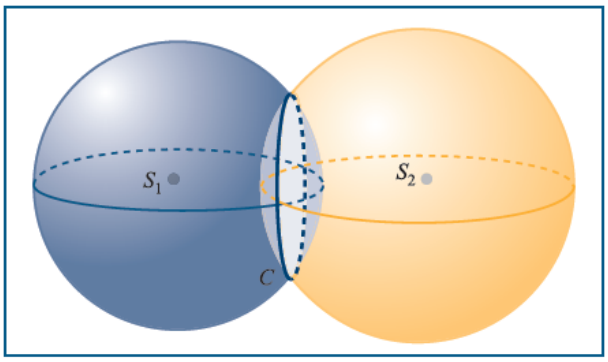



Figura 5.1: Intersezione tra due ipersfere

to per ricavare gli assiomi di disgiunzione tra classi. Il modello TransC-OWL, essendo orientato alle classi, si è dimostrato in grado di risolvere questo problema con ottimi risultati sfruttando direttamente la geometria del modello. Come spiegato già nella sezione di TransC [Lv+18], ad ogni classe viene associata una ipersfera all'interno dello spazio di embedding in modo che se una determinata entità è istanza di una classe, allora deve trovarsi all'interno del volume dell'ipersfera. Logicamente, se due classi hanno delle istanze in comune, questo significa che l'intersezione tra le due ipersfere associate alle due classi non è vuota, come in figura 5.1.

Il problema dell'intersezione tra classi, dunque, è stato ricondotto al problema dell'intersezione tra ipersfere (Figura 5.2), dove la formula per verificare se l'intersezione è vuota (noti $s_1$ ed $s_2$ come i centri delle due ipersfere, $r_1$ ed $r_2$ come i due raggi associati) è:

$$d > |r_1 + r_2|$$

dove $d = ||s_1 - s_2||_2$ ovvero la distanza tra i due centri delle ipersfere. L'unico problema, sfruttando direttamente la geometria del modello, potrebbe essere la difficoltà nel rappresentare correttamente le classi con l'ipersfera adatta nel caso in cui la conoscenza su alcune classi è minima, ovvero l'ipersfera imparata potrebbe non essere propriamente corretta e quindi di conseguenza avere risultati errati nella predizione di classi disgiunte (tipico della OWA in un ambito come il sematic web).

Per questo motivo è da considerare maggiormente la valutazione su quel-

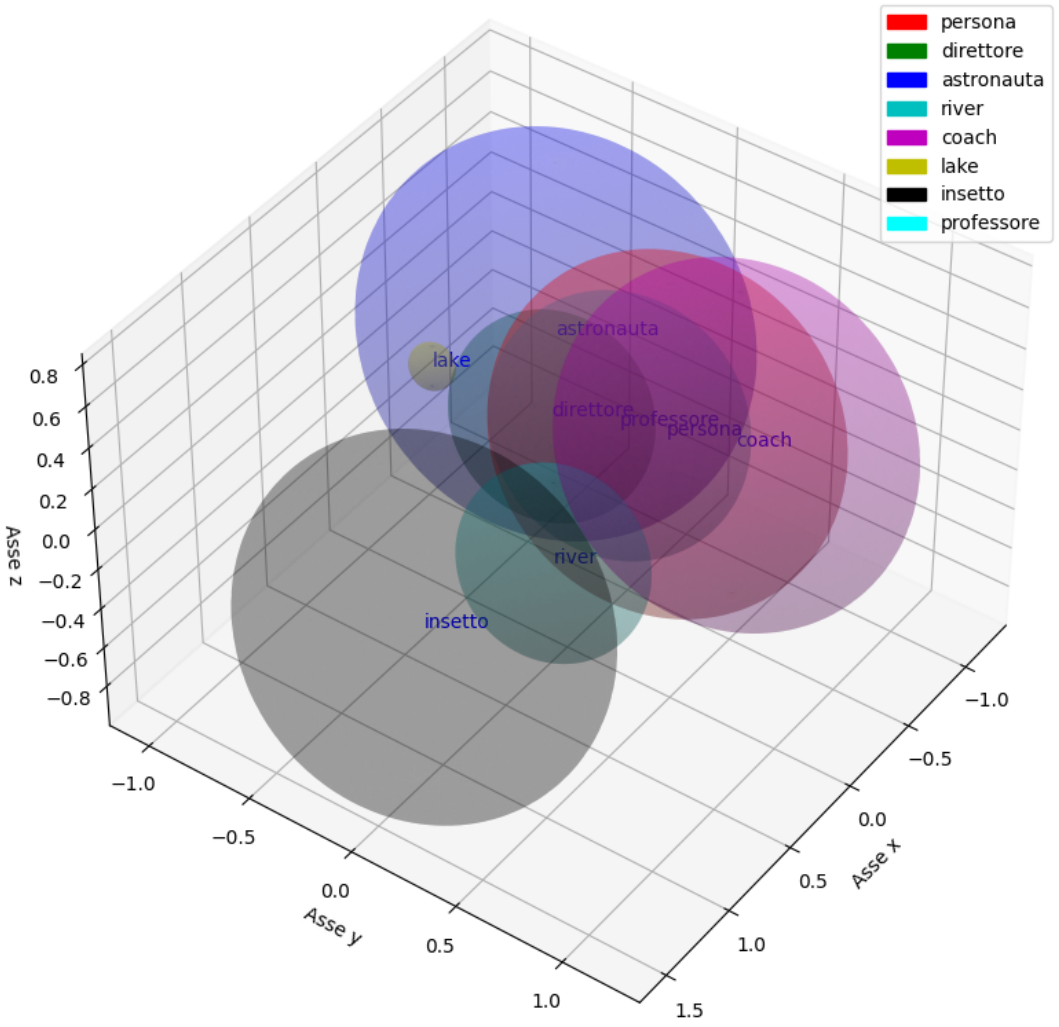



Figura 5.2: Visualizzazione in 3D delle ipersfere (TransC-OWL) associate alle classi tramite algoritmo PCA.

le classi che sono state imparate con un numero sufficiente di esempi (attraverso le relazioni `typeOf` e `subClassOf`). La regola di disgiunzione tra classi è pertanto:

$$c_1 \sqcap c_2 = \bot, c_1 \neq c_2$$

La differenza tra confidenza e probabilità dell'implicazione è la seguente:

1. La **confidenza** (o probabilità condizionata), ovvero $P(\neg c_2|c_1)$ è la probabilità che una determinata entità non sia istanza di $c_2$ sapendo che invece è istanza di $c_1$ (istanze non in comune sul totale di istanze di $c_1$).

| Dataset | Modello | Epoca | P | R | Conf. | Lift | $\sigma$ Lift |
|---|---|---|---|---|---|---|---|
| Nell | TransC | 100 | 0,985 | 0,157 | 0,994 | 1.994 | 0,531 |
| | | 500 | 0,988 | 0,686 | 0,995 | 1.995 | 0,509 |
| | | 1000 | 0,987 | **0,987** | 0,995 | 1,995 | 0,510 |
| | TransC-OWL | 100 | 0,989 | 0,169 | 0,983 | 1,983 | 0,460 |
| | | 500 | **0,990** | 0,721 | **0,997** | 1,997 | 0,511 |
| | | 1000 | 0,989 | 0,982 | **0,997** | 1,997 | 0,510 |
| | TransM | 100 | 0,977 | 0,499 | 0,990 | **2,030** | 0,516 |
| | | 500 | 0,977 | 0,499 | 0,991 | 2,007 | 0,507 |
| | | 1000 | 0,977 | 0,503 | 0,991 | 2,019 | 0,507 |
| DBpedia100K | TransC | 100 | 0,757 | 0,540 | 0,999 | 1,984 | 0,468 |
| | | 500 | 0,788 | **0,933** | **0,999** | **1,986** | 0,484 |
| | TransC-OWL | 100 | 0,791 | 0,535 | 0,999 | 1,981 | 0,457 |
| | | 500 | **0,784** | 0,929 | **0,999** | 1,986 | 0,481 |
| | TransM | 100 | 0,777 | 0,496 | 0,958 | 1,983 | 0,482 |
| | | 500 | 0,784 | 0,497 | 0,968 | 1,902 | 0,482 |
| DBpedia15K | TransC | 100 | 0,897 | 0,587 | **0,999** | 1,997 | 0,501 |
| | | 500 | 0,885 | **0,990** | **0,999** | 1,996 | 0,486 |
| | TransC-OWL | 100 | **0,899** | 0,602 | **0,999** | 1,986 | 0,512 |
| | | 500 | 0,885 | 0,986 | **0,999** | 1,996 | 0,486 |
| | TransM | 100 | 0,883 | 0,499 | 0,996 | **2,035** | 0,497 |
| | | 500 | 0.885 | 0,498 | 0,993 | 1,983 | 0,483 |

Tabella 5.11: Risultati ottenuti nell'estrazione di regole di disgiunzione tra classi per i vari dataset testati

2. La **probabilità dell'implicazione** è

$$P(c_1 \rightarrow \neg c_2) = P(\neg c_1 \vee \neg c_2) \overset{\text{De Morgan}}{=} 1 - P(c_1 \wedge c_2)$$

ossia proprio la probabilità della disgiunzione tra classi, ovvero la probabilità che una determinata istanza non sia in comune tra le due classi.

Prima di tutto si è tenuto conto della **gerarchia delle classi**, ovvero assiomi definiti già a livello di super-classi sono ridondanti a livello di sotto-classi, ad es. la regola $\texttt{Persona} \sqcap \texttt{Oggetti inanimati} = \bot$ è sicuramente vera, quindi varrà anche per tutte le sottoclassi come $\texttt{Maschio} \sqcap \texttt{Oggetti inanimati} = \bot$. È stato valutato il comportamento su diverse epoche per tener conto del caso in cui le ipersfere associate a ciascuna classe non siano completamente esatte (per simulare il comportamento tipico della OWA). Il modello risulta essere molto conservativo (come si nota dai risultati in tabella 5.11), ovvero la precision rimane costante ad ogni epoca, quindi le regole trovate sono sicuramente esatte.

Le regole trovate grazie alla geometria di TransC-OWL e TransM, inoltre, sono corrette perché la confidenza tende al 100% (così come la prob. dell'implicazione). Il lift ritrovato, invece, è molto vicino a 2. Questo significa che le regole trovate sono comunque positive, cioè $c_1$ influenza $\neg c_2$ anche se non in modo forte. Questo è coerente con la regola di disgiunzione tra classi perché le classi che rientrano in una regola sono per la maggior parte indipendenti tra di loro (lift basso), ad esempio la regola di disgiunzione:

$$\texttt{insetto} \sqcap \texttt{gioco} = \bot$$

è vera perché nessuna istanza di `insetto` è a sua volta istanza di `gioco` e viceversa ma è anche vero che le due classi sono indipendenti tra di loro. Diversamente invece una regola del tipo:

$$\texttt{maschio} \sqcap \texttt{femmina} = \bot$$

è anch'essa vera ma il lift è necessariamente più alto perché c'è una forte ed evidente relazione tra le due classi. Per questo motivo si è voluto verificare se il modello fosse in grado di trovare le regole con un maggiore lift (tabella 5.12):

Com'è possibile notare in tabella 5.12, le classi con una maggiore relazione vengono identificate correttamente dal modello.

Per quanto riguarda il modello TransM, il discorso è diverso anche a

| **Regola** |
|---|
| $\texttt{insetto} \sqcap \texttt{persona} = \bot$ |
| $\texttt{invertebrato} \sqcap \texttt{persona} = \bot$ |
| $\texttt{parte corpo} \sqcap \texttt{persona} = \bot$ |
| $\texttt{artropodo} \sqcap \texttt{persona} = \bot$ |
| $\texttt{vertebrato} \sqcap \texttt{persona} = \bot$ |
| $\texttt{animale} \sqcap \texttt{persona} = \bot$ |
| $\texttt{aracnide} \sqcap \texttt{persona} = \bot$ |
| $\texttt{batterio} \sqcap \texttt{persona} = \bot$ |
| ... |

Tabella 5.12: Regole di disgiunzione tra classi con lift maggiore

causa della natura differente con la quale le classi vengono modellate. Il problema dell'intersezione tra classi, in questo caso, è stato ricondotto al problema dell'intersezione tra iperellissoidi, dove il calcolo è sicuramente più complicato rispetto al caso precedente. Come enunciato in [AG03], si definiscano innanzitutto i due iperellissoidi come:

$$E_A = \{x : (x-a)^T A(x-a) \leq 1\}$$

$$E_B = \{x : (x-b)^T B(x-b) \leq 1\}$$

dove $A$, $B$ sono le matrici di covarianza mentre $a$, $b$ sono le medie degli ellissoidi. Per poter definire l'intersezione tra $E_A$ e $E_B$, bisogna definire innanzitutto una funzione $K : (0,1) \to \mathbb{R}$, come spiegato in [GH12]:

$$K := 1 - (b-a)^T \left(\frac{1}{1-s}A^{-1} + \frac{1}{s}B^{-1}\right)^{-1} (b-a)$$

Affinché $E_A$ si intersechi con $E_B$:

$$E_A \cap E_B = \{\} \text{ sse } K(s) < 0 \text{ per qualche } s \in (0,1)$$

Pertanto, si può verificare se due ellissoidi collidono (figura 5.3) trovando il valore di $s$ che minimizza $K$ nell'intervallo $(0,1)$. Per far ciò, si sono utilizzate librerie in python già convalidate come scipy.

Si è voluto anche valutare il comportamento rispetto all'assunzione di **disgiunzione forte**, ossia sottoclassi sibling di una stessa classe implicitamente considerate come disgiunte. È stato valutato sul dataset Nell con i seguenti risultati:

In linea di massima, com'è possibile notare in tabella 5.13, anche in questo caso (disgiunzione forte) la maggior parte delle regole sono esatte, tuttavia a causa del problema dell'incompletezza del dataset sono state dedotte le regole $\texttt{direttore} \sqcap \texttt{allenatore} = \bot$, $\texttt{direttore} \sqcap \texttt{atleta} = \bot$, $\texttt{professore} \sqcap \texttt{allenatore} = \bot$ e $\texttt{professore} \sqcap \texttt{atleta} = \bot$ che in realtà non sono disgiunte perché quasi sicuramente esisterà un individuo che è ad esempio sia `allenatore` che `professore`. Le più interessan-

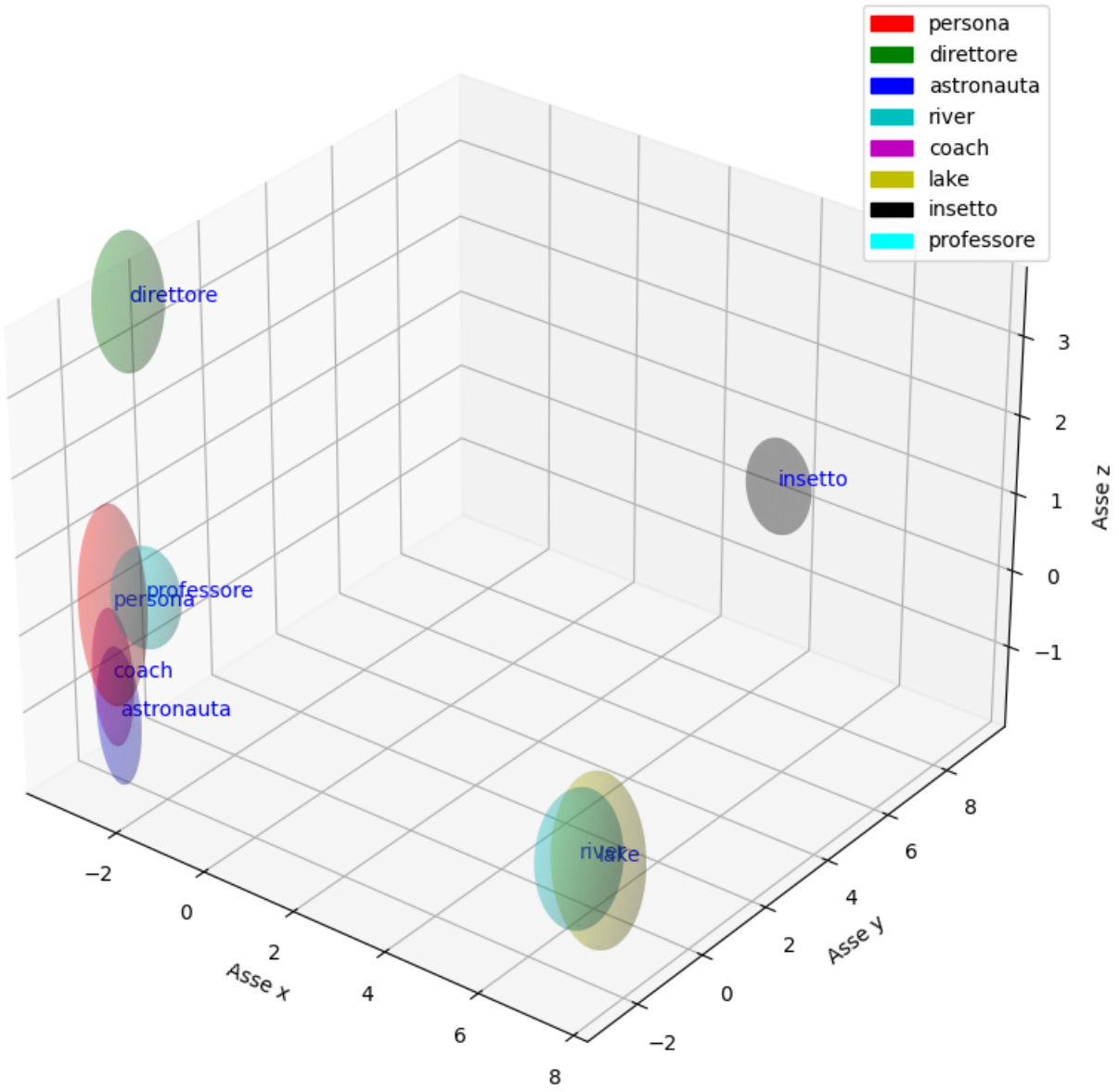



Figura 5.3: Visualizzazione in 3D degli ellissoidi (TransM) associati alle classi tramite algoritmo PCA. Notare come persona si trovi al centro tra professore, astronauta e coach. `Lago` e `fiume` sono abbastanza vicini tra loro, mentre `insetto` è lontano tra tutti.

ti, invece sono quelle riguardanti la classe `parte corpo` che può essere molto utile in ambito medico oppure anche quelle riguardanti la classe `specie`.

Il lift in questo caso è superiore rispetto alla media il che significa che le classi che compaiono tra le regole di disgiunzione forte trovate hanno una relazione maggiore rispetto alle altre regole di disgiunzione.

| Regola | Classe madre |
|---|---|
| sito web $\sqcap$ palazzo $= \bot$ | locazione |
| zoo $\sqcap$ area sci $= \bot$ | attrazione |
| direttore $\sqcap$ allenatore $= \bot$<br>direttore $\sqcap$ atleta $= \bot$<br>astronauta $\sqcap$ allenatore $= \bot$<br>astronauta $\sqcap$ atleta $= \bot$<br>professore $\sqcap$ allenatore $= \bot$<br>professore $\sqcap$ atleta $= \bot$ | persona<br>persona<br>persona<br>persona<br>persona<br>persona |
| montagna $\sqcap$ parco $= \bot$<br>montagna $\sqcap$ fiume $= \bot$<br>lago $\sqcap$ parco $= \bot$<br>lago $\sqcap$ fiume $= \bot$ | geolocalizzabile<br>geolocalizzabile<br>geolocalizzabile<br>geolocalizzabile |
| vena $\sqcap$ tessuto cerebrale $= \bot$<br>muscolo $\sqcap$ osso $= \bot$<br>muscolo $\sqcap$ tessuto cerebrale $= \bot$ | parte corpo<br>parte corpo<br>parte corpo |
| virus $\sqcap$ animale $= \bot$<br>batterio $\sqcap$ animale $= \bot$ | specie<br>specie |
| motore automobile $\sqcap$ veicolo $= \bot$ | prodotto |
| ... | ... |

Tabella 5.13: Regole di disgiunzione tra classi sorelle nel dataset Nell

### 5.2.4 Relazioni Composte

Una relazione $r_1$ è **composta** con una relazione $r_2$ ed $r_3$ se $\forall x, y, z$:

$$r_2(x, y) \wedge r_3(y, z) \Rightarrow r_1(x, z)$$

Il modello TransE e derivati sono in grado di imparare le relazioni composte, seppur con maggior difficoltà rispetto alle relazioni precedenti. Questo è dovuto, per ovvi motivi, al fatto che sono necessari più calcoli matematici in quanto sono incluse ben 3 relazioni, facendo aumentare la probabilità di errore. Se si prendono ad esempio tre relazioni tra di loro composte come `marito_di`, `figlio_di`, `genero_di` e tre entità come `Papà`, `Mamma`, `Nonno`, si avrà che:

1. (`Papà`, `marito_di`, `Mamma`) vera quindi:

$$\texttt{Papà} + \texttt{marito\_di} \approx \texttt{Mamma}$$

2. (Mamma, figlio_di, Nonno) vera quindi:

$$\texttt{Mamma} + \texttt{figlio\_di} \approx \texttt{Nonno}$$

3. (Papà, genero_di, Nonno) vera quindi:

$$\texttt{Papà} + \texttt{genero\_di} \approx \texttt{Nonno}$$

Quindi, eseguendo la prima sostituzione si ha che:

$$\texttt{Mamma} + \texttt{figlio\_di} \approx \texttt{Papà} + \texttt{genero\_di}$$

Ed eseguendo anche la seconda sostituzione si ha che:

$$\texttt{Papà} + \texttt{marito\_di} + \texttt{figlio\_di} \approx \texttt{Papà} + \texttt{genero\_di}$$

$$\cancel{\texttt{Papà} +} \texttt{marito\_di} + \texttt{figlio\_di} \cancel{- \texttt{Papà}} - \texttt{genero\_di} \approx 0$$

$$\texttt{marito\_di} + \texttt{figlio\_di} \approx \texttt{genero\_di}$$

Pertanto, affinché le tre relazioni siano composte, è necessario che $r_1 + r_2 = r_3$. Potrebbe però verificarsi un problema nel caso in cui il modello non riesca a imparare nulla su determinate relazioni (e quindi il loro vettore associato risulterebbe simile al vettore nullo). In questo caso, se il vettore nullo è ad esempio $r_1$ si avrebbe che $r_2 = r_3$, oppure se il vettore nullo è $r_2$ si avrebbe che $r_1 = r_3$ oppure se il vettore nullo è $r_3$ si avrebbe che $r_1 = -r_2$. In tutti e tre i casi non si ha una relazione composta, quindi per evitare ciò si è deciso di escludere dall'estrazione di regole tutte quelle relazioni che per l'appunto hanno un vettore simile al vettore nullo.
Per verificare quali fossero quindi le relazioni composte (o quelle relazioni che più si avvicinano ad esserlo) si è proceduti esattamente eseguendo la somma tra i due vettori $r_1$ e $r_2$ calcolando la distanza tra il vettore risultate e il vettore $r_3$: minore sarà la distanza e maggiore sarà la probabilità che le tre relazioni siano tra di loro composte. Sono state prese in considerazione solamente le relazioni che hanno riportato una

| Dataset | Relazione composta | Prob. |
|---|---|---|
| DBpediaYAGO | provincia + nata_in ⇒ origina_da | 67% |
| | provincia + morta_in ⇒ origina_da | 66% |
| | governatore + morto_in ⇒ vive_in | 64% |
| | provincia + nata_in ⇒ morta_in | 63% |
| | governatore + orgina_da ⇒ vive_in | 62% |
| | ... | |
| DBpedia15K | foce_fiume + confluenza_sorgente ⇒ confluenza_montagna | 64% |
| | serie_televisiva + confluenza_sorgente ⇒ costruzione_significante | 61% |
| | ... | |
| DBpedia100K | gestione_montagna + governo_montagna ⇒ gestione_posto | 62% |
| | direttore + governo_montagna ⇒ scrittore | 61% |
| | gestione_posto + governo_paese ⇒ gestione_montagna | 58% |
| | ... | |
| Nell | organizzazione + studente ⇒ studente_diplomato | 36% |
| | organizzazione + atleta ⇒ squadra | 30% |
| | ... | |

Tabella 5.14: Risultati ottenuti nell'estrazione di regole composte

distanza minore rispetto a una certa soglia $\delta r = 4$. La probabilità che esse siano composte è stata calcolata come:

$$P = (\delta r - dist)\frac{100}{\delta r}$$

in modo tale che una distanza pari a 0 venga etichettata come probabilità certa (100%), mentre una distanza pari a 4 è stata etichettata come 0%.

In Tab. 5.14 le relazioni composte selezionate con le relative metriche.

I modelli TransC-OWL e TransM si sono quindi dimostrati capace di apprendere le relazioni composte in tutti i dataset, anche se sono state selezionate le regole composte solo nel dataset DBpedia15K in quanto sono le più evidenti.

| Dataset | Relazione composta | Conf. | Lift |
|---|---|---|---|
| DBpedia15K | `fiume + sorgente_montagna ⇒ luogo_sorgente` | 100% | 41943.5 |
| | `fiume + luogo_sorgente ⇒ origine` | 100% | 41943.5 |
| | `fiume + bocca_sorgente ⇒ luogo_sorgente` | 100% | 41943.5 |
| | `fiume + bocca_montagna ⇒ origine` | 100% | 41943.5 |
| | `vice_presidente + morto_in ⇒ luogo_di_riposo` | 100% | 360.03 |
| | `network + broadcast_network ⇒ formato` | 100% | 198.314 |
| DBpediaYAGO | - | - | - |
| DBpedia100K | - | - | - |
| Nell | - | - | - |

Tabella 5.15: Risultati ottenuti nell'estrazione di regole composte

### 5.2.5 Relazioni Transitive

Una relazione $r_1$ è **transitiva** se $\forall x, y, z$:

$$r_1(x, y) \wedge r_1(y, z) \Rightarrow r_1(x, z)$$

Si può affermare che una relazione transitiva sia un caso particolare della relazione composta dove $r_2 = r_1$ e $r_3 = r_1$. Si è preferito differenziarle perché in TransC-OWL e TransM hanno obiettivi differenti, cioè:

1. La relazione composta riguarda le triple relazionali, quindi lavora sui vettori entità e relazione.

2. La relazione transitiva riguarda le due relazioni speciali `typeOf` e `subClassOf`, quindi lavora sul vettore entità e vettore dei concetti derivante da TransC.

Ad esempio se si hanno un'entità come `Di Caprio` e due classi come `Attore` e `Persona` si avrà che (seguendo la funzione di score di TransC):

1. (`Di Caprio`, `typeOf`, `Attore`) vera quindi:

$$||\texttt{Di Caprio} - \texttt{Attore}||_2 < r(\texttt{Attore})$$

2. (Attore, subClassOf, Persona) vera quindi:

$$||\texttt{Attore}-\texttt{Persona}||_2 < r(\texttt{Persona})-r(\texttt{Attore}) \; se \; r(\texttt{Attore}) < r(\texttt{Persona})$$

L'idea di base è che quindi, se Di Caprio si trova all'interno della sfera Attore e Attore si trova all'interno della sfera Persona, allora anche Di Caprio si troverà all'interno della sfera Persona. Dal punto di vista matematico:

$$||\texttt{Di Caprio}-\texttt{Attore}||_2 < r(\texttt{Attore}) < -||\texttt{Attore}-\texttt{Persona}||_2+r(\texttt{Persona})$$

$$||\texttt{Di Caprio} - \texttt{Attore}||_2 < -||\texttt{Attore} - \texttt{Persona}||_2 + r(\texttt{Persona})$$

$$||\texttt{Di Caprio} - \texttt{Attore}||_2 + ||\texttt{Attore} - \texttt{Persona}||_2 < +r(\texttt{Persona})$$

$$||\texttt{Di Caprio} - \texttt{Attore} + \texttt{Attore} - \texttt{Persona}||_2 < r(\texttt{Persona})$$

$$||\texttt{Di Caprio} - \texttt{Persona}||_2 < r(\texttt{Persona})$$

Si è dimostrata quindi la valenza della transitività di typeOf nel modello TransC in quanto partendo da (Di Caprio, typeOf, Attore) e (Attore, subClassOf, Persona) $\Rightarrow$ (Di Caprio, typeOf, Persona).

In modo molto simile è dimostrabile anche la transitività della relazione subClassOf. Avendo ad esempio tre classi come Attore, Persona, Homo Sapiens, si avrà che:

1. (Attore, subClassOf, Persona) vera quindi (se $r(\texttt{Attore}) < r(\texttt{Persona})$):

$$||\texttt{Attore} - \texttt{Persona}||_2 < r(\texttt{Persona}) - r(\texttt{Attore})$$

2. (Persona, subClassOf, Homo Sapiens) vera quindi (se $r(\texttt{Persona}) < r(\texttt{Homo Sapiens})$):

$$||\texttt{Persona} - \texttt{Homo Sapiens}||_2 < r(\texttt{Homo Sapiens}) - r(\texttt{Persona})$$

| Dataset | Predizione | Spiegazione |
|---|---|---|
| DBpediaYago | Cliff Richard ∈ Cantante | Cliff Richard ⊑ Cantante Rock ⊓ Cantante Rock ⊑ Cantante |
| | AthleticFC ∈ Squadra<br>... | AthleticFC ⊑ Squadra inglese ⊓ Squadra inglese ⊑ Squadra<br>... |
| Nell | Rachel Miner ∈ Persona | Rachel Miner ⊑ Attore ⊓ Attore ⊑ Persona |
| | Torta mele ∈ Cibo<br>... | Torta mele ⊑ Cucinata ⊓ Cucinata ⊑ Cibo<br>... |
| DBpedia15K | David Crane ∈ Governatore<br>... | David Crane ⊑ Persona ⊓ Persona ⊑ Governatore<br>... |
| DBpedia100K | Squadra ⊑ Organizzazione | Squadra ⊑ Posto ⊓ Posto ⊑ Organizzazione |
| | Conflitto militare ⊑ Evento sociale<br>... | Conflitto militare ⊑ Evento ⊓ Evento ⊑ Evento sociale<br>... |

Tabella 5.16: Risultati ottenuti nell'estrazione di regole transitive

Anche in questo caso, l'idea di base è che se **Attore** si trova all'interno della sfera **Persona** e **Persona** si trova all'interno della sfera **Homo Sapiens**, allora anche **Attore** si troverà all'interno della sfera **Homo Sapiens**. Dal punto di vista matematico:

$$||\texttt{Attore} - \texttt{Persona}||_2 < r(\texttt{Persona}) - r(\texttt{Attore})$$

$$||\texttt{Attore} - \texttt{Persona}||_2 + r(\texttt{Attore}) < r(\texttt{Persona})$$

$$||\texttt{Attore} - \texttt{Persona}||_2 + ||\texttt{Persona} - \texttt{Homo S.}||_{2+} < r(\texttt{Homo S.}) - r(\texttt{Attore})$$

$$||\texttt{Attore} - \cancel{\texttt{Persona} + \texttt{Persona}} - \texttt{Homo S.}||_{2+} < r(\texttt{Homo S.}) - r(\texttt{Attore})$$

$$||\texttt{Attore} - \texttt{Homo S.}||_{2+} < r(\texttt{Homo S.}) - r(\texttt{Attore})$$

La relazione di transitività è molto utile per spiegare una predizione. In Tab. 5.16 sono riportate alcune spiegazioni date dal modello TransC-OWL.

I modelli TransC-OWL e TransM si sono dimostrati adatti anche nell'imparare la transitività delle relazioni `typeOf` e `subClassOf`.

Un problema di questa rappresentazione (con la sfera ed ellissoide) è che però, nelle classi che nella gerarchia si trovano più in alto possono essere erroneamente attribuite superclassi di altre classi minori che non c'entrano nulla l'una con l'altra. Questo perché, il raggio della sfera associata alla classe più importante (o i raggi associati ad ogni ellissoide), ad esempio come successo in tutti i dataset con la classe `Persona`, è talmente grande da inglobare erroneamente altre classi non legate tra di loro.

# Conclusione

Si conclude qui questo lavoro di tesi sulla costruzione di modelli traslazionali orientati alle classi per l'estrazione di conoscenza e *triple classification*. Si è passati dal presentare lo stato dell'arte, tra cui i principali modelli correlati che hanno dato origine ai lavori di questa tesi, alla presentazione dei linguaggi RDF, RDFS e OWL. Dopo aver illustrato i due lavori di questa tesi, TransC-OWL e TransM, sono stati presentati i risultati ottenuti con entrambi, ovvero:

- l'iniezione di conoscenza in TransC-OWL, in alcuni casi, si è dimostrata efficace ed ha migliorato le prestazioni di TransC [Lv+18] nel task di triple classification;
- l'innovativa modellazione delle classi di TransM con distribuzioni gaussiane multivariate ha permesso di avere tra le migliori prestazioni nel task di triple classification `typeOf` e a volte anche `non typeOf`;
- grazie alla rappresentazione maggiormente orientata alle classi è stato possibile estrarre regole come la disgiunzione tra classi e regole transitive `typeOf` e `subClassOf` in entrambi i modelli riuscendo nell'intento di dare una spiegazione di una predizione;
- la regola più interessante che è stata trovata da entrambi i modelli riguardano la disgiunzione tra classi. Nello specifico TransM è riuscito ad ottenere in alcuni casi anche regole con un lift superiore a 2, indicando come l'associazione tra $c_1$ e $c_2$ nella regola $c_1 \sqcap c_2 = \bot$ sia due volte più forte di quanto ci si aspetterebbe se fossero indipendenti tra loro.

Questo lavoro di tesi apre diverse prospettive allo sviluppo di ulteriori modelli orientati alle classi, anche come espansione del modello TransM, tra cui:

- considerare un modello basato su misture di gaussiane [BF20] piuttosto che come semplice distribuzione multivariata potrebbe rivelarsi l'ideale per risolvere il problema di classi definite come unione di altre classi disgiunte tra loro (ovvero a livello geometrico una classe potrebbe essere definita come unione di altri iperellissoidi che non si intersecano), task che TransM non è in grado attualmente di risolvere.

- effettuare l'iniezione di conoscenza nel modello TransM potrebbe rappresentare un ottima possibilità per migliorare ulteriormente le prestazioni del modello;

- la valutazione di entrambi i modelli nel task di Link Prediction per i vari dataset (trascurata in questo lavoro di tesi) potrebbe ulteriormente rafforzare la validità e l'efficacia, fornendo una valutazione comparativa delle prestazioni rispetto agli altri modelli e consentendo una migliore comprensione delle limitazioni e delle aree di miglioramento per futuri sviluppi.

In conclusione, la modellazione delle classi nei KGE utilizzando un approccio basato su ipervettori (nello specifico ipersfere e iperellissoidi) ha dimostrato di essere efficace nel catturare le relazioni complesse tra le classi aprendo prospettive per ulteriori sviluppi futuri nell'estrazione di conoscenza e miglioramenti nel task di triple classification.

# Ringraziamenti

Desidero innanzitutto esprimere i miei sinceri ringraziamenti al Professore Nicola Fanizzi, relatore di questa tesi, per il suo prezioso sostegno, la sua disponibilità e la pazienza dimostrata. I suoi insegnamenti hanno avuto un impatto significativo sulla mia crescita come studente e mi hanno preparato per un futuro professionale migliore.

Desidero poi rivolgere un sentito ringraziamento alla mia famiglia che mi ha sostenuto in ogni fase del mio percorso universitario. Il loro costante supporto, affetto e incoraggiamento sono stati fondamentali per il mio successo accademico.

Desidero esprimere la mia gratitudine a tutti i miei amici e compagni di corso, in particolare a Stefan, Domenico e Tonio, per il tempo prezioso che abbiamo trascorso insieme. I momenti condivisi e il supporto reciproco hanno reso questa esperienza universitaria davvero speciale e indimenticabile.

Infine, un ringraziamento speciale è dedicato soprattutto a me stesso. Desidero riconoscere l'impegno e la determinazione che ho dimostrato lungo questo percorso. Sono orgoglioso di questo primo obiettivo personale e che sia soprattutto il primo di tanti.

# Bibliografia

[AG03] S. Alfano e M. L. Greer. «Determining If Two Solid Ellipsoids Intersect». In: *Journal of Guidance, Control, and Dynamics* 26.1 (2003), pp. 106–110. DOI: 10.2514/2.5020. eprint: https://doi.org/10.2514/2.5020.

[Baa+03] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi e P. F. Patel-Schneider, cur. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003. ISBN: 0-521-78176-0.

[BF20] N. Bouguila e W. Fan. *Mixture Models and Applications*. 2020. ISBN: 978-3-030-23875-9. DOI: 10.1007/978-3-030-23876-6.

[BHB09] C. Bizer, T. Heath e T. Berners-Lee. «Linked Data - The Story So Far». In: *Int. J. Semantic Web Inf. Syst.* 5.3 (2009), pp. 1–22. DOI: 10.4018/jswis.2009081901.

[Bia+20] F. Bianchi, G. Rossiello, L. Costabello, M. Palmonari e P. Minervini. «Knowledge Graph Embeddings and Explainable AI». In: *CoRR* abs/2004.14843 (2020). arXiv: 2004.14843. URL: https://arxiv.org/abs/2004.14843.

[Bor+13] A. Bordes, N. Usunier, A. Garcıéa-Durán, J. Weston e O. Yakhnenko. «Translating Embeddings for Modeling Multi-relational Data». In: *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*. A cura di C. J. C. Burges, L. Bottou, Z. Ghahramani e K. Q. Weinberger. 2013, pp. 2787–2795. URL: https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

[Che+15] J. Cheng, Z. Wang, J. Wen, J. Yan e Z. Chen. «Contextual Text Understanding in Distributional Semantic Space». In:

*Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. A cura di J. Bailey, A. Moffat, C. C. Aggarwal, M. de Rijke, R. Kumar, V. Murdock, T. K. Sellis e J. X. Yu. ACM, 2015, pp. 133–142. DOI: 10.1145/2806416.2806517.

[DHS11] J. C. Duchi, E. Hazan e Y. Singer. «Adaptive Subgradient Methods for Online Learning and Stochastic Optimization». In: *J. Mach. Learn. Res.* 12 (2011), pp. 2121–2159. DOI: 10.5555/1953048.2021068. URL: https://dl.acm.org/doi/10.5555/1953048.2021068.

[Din+18] B. Ding, Q. Wang, B. Wang e L. Guo. «Improving Knowledge Graph Embedding Using Simple Constraints». In: *CoRR* abs/1805.02408 (2018). arXiv: 1805.02408. URL: http://arxiv.org/abs/1805.02408.

[dQF21] C. d'Amato, N. F. Quatraro e N. Fanizzi. «Embedding Models for Knowledge Graphs Induced by Clusters of Relations and Background Knowledge». In: *Inductive Logic Programming - 30th International Conference, ILP 2021, Virtual Event, October 25-27, 2021, Proceedings*. A cura di N. Katzouris e A. Artikis. Vol. 13191. Lecture Notes in Computer Science. Springer, 2021, pp. 1–16. DOI: 10.1007/978-3-030-97454-1_1.

[Ehi+20] K. E. Ehimwenma, P. Crowther, M. D. Beer e S. A. Sharji. «An SQL Domain Ontology Learning for Analyzing Hierarchies of Structures in Pre-Learning Assessment Agents». In: *SN Comput. Sci.* 1.6 (2020), p. 335. DOI: 10.1007/s42979-020-00338-1.

[Fär+18] M. Färber, F. Bartscherer, C. Menne e A. Rettinger. «Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO». In: *Semantic Web* 9.1 (2018), pp. 77–129. DOI: 10.3233/SW-170275.

[GH12] I. Gilitschenski e U. D. Hanebeck. «A robust computational test for overlap of two arbitrary-dimensional ellipsoids in fault-detection of Kalman filters». In: *15th International*

*Conference on Information Fusion, FUSION 2012, Singapore, July 9-12, 2012*. IEEE, 2012, pp. 396–401. URL: https://ieeexplore.ieee.org/document/6289830/.

[GML15] K. Guu, J. Miller e P. Liang. «Traversing Knowledge Graphs in Vector Space». In: *CoRR* abs/1506.01094 (2015). arXiv: 1506.01094. URL: http://arxiv.org/abs/1506.01094.

[GSL19] N. Guan, D. Song e L. Liao. «Knowledge graph embedding with concepts». In: *Knowl. Based Syst.* 164 (2019), pp. 38–44. DOI: 10.1016/j.knosys.2018.10.008.

[He+15] S. He, K. Liu, G. Ji e J. Zhao. «Learning to Represent Knowledge Graphs with Gaussian Embedding». In: *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. A cura di J. Bailey, A. Moffat, C. C. Aggarwal, M. de Rijke, R. Kumar, V. Murdock, T. K. Sellis e J. X. Yu. ACM, 2015, pp. 623–632. DOI: 10.1145/2806416.2806502.

[HKR10] P. Hitzler, M. Krötzsch e S. Rudolph. *Foundations of Semantic Web Technologies*. Chapman e Hall/CRC Press, 2010. ISBN: 9781420090505. URL: http://www.semantic-web-book.org/.

[Hog+20] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab e A. Zimmermann. «Knowledge Graphs». In: *CoRR* abs/2003.02320 (2020). arXiv: 2003.02320. URL: https://arxiv.org/abs/2003.02320.

[LCH06] Y. Lecun, S. Chopra e R. Hadsell. «A tutorial on energy-based learning». In: *Predicting Structured Data*. MIT Press, 2006.

[Leh+15] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer e C. Bizer. «DBpedia - A large-scale, multilingual kno-

wledge base extracted from Wikipedia». In: *Semantic Web* 6.2 (2015), pp. 167–195. DOI: 10.3233/SW-140134.

[Lin+15] Y. Lin, Z. Liu, M. Sun, Y. Liu e X. Zhu. «Learning Entity and Relation Embeddings for Knowledge Graph Completion». In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. A cura di B. Bonet e S. Koenig. AAAI Press, 2015, pp. 2181–2187. URL: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9571.

[Liu+19] Y. Liu, H. Li, A. Garcıéa-Durán, M. Niepert, D. Oñoro-Rubio e D. S. Rosenblum. «MMKG: Multi-Modal Knowledge Graphs». In: *CoRR* abs/1903.05485 (2019). arXiv: 1903.05485. URL: http://arxiv.org/abs/1903.05485.

[Lv+18] X. Lv, L. Hou, J. Li e Z. Liu. «Differentiating Concepts and Instances for Knowledge Graph Embedding». In: *CoRR* abs/1811.04588 (2018). arXiv: 1811.04588. URL: http://arxiv.org/abs/1811.04588.

[MdF16] P. Minervini, C. d'Amato e N. Fanizzi. «Efficient energy-based embedding models for link prediction in knowledge graphs». In: *J. Intell. Inf. Syst.* 47.1 (2016), pp. 91–109. DOI: 10.1007/s10844-016-0414-7.

[Mik+13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado e J. Dean. «Distributed Representations of Words and Phrases and their Compositionality». In: *CoRR* abs/1310.4546 (2013). arXiv: 1310.4546. URL: http://arxiv.org/abs/1310.4546.

[MP15] P. Moore e H. V. Pham. «On Context and the Open World Assumption». In: *29th IEEE International Conference on Advanced Information Networking and Applications Workshops, AINA 2015 Workshops, Gwangju, South Korea, March 24-27, 2015*. A cura di L. Barolli, M. Takizawa, F. Xhafa, T. Enokido e J. H. Park. IEEE Computer Society, 2015, pp. 387–392. DOI: 10.1109/WAINA.2015.7.

[PTG12] S. T. Piantadosi, J. B. Tenenbaum e N. D. Goodman. «Bootstrapping in a language of thought: A formal model of

numerical concept learning». In: *Cognition* 123.2 (2012), pp. 199–217. ISSN: 0010-0277. DOI: https://doi.org/10.1016/j.cognition.2011.11.005. URL: https://www.sciencedirect.com/science/article/pii/S0010027711002769.

[Qua19] N. F. Quatraro. «TransROWL-HRS: Un approccio combinato per l'iniezione di conoscenza di fondo e ragionamento nella costruzione di embedding di knowledge graph». Tesi di laurea. Università degli studi di Bari Aldo Moro, 2019.

[RdF21] G. Rizzo, C. d'Amato e N. Fanizzi. «An unsupervised approach to disjointness learning based on terminological cluster trees». In: *Semantic Web* 12.3 (2021), pp. 423–447. DOI: 10.3233/SW-200391.

[Riz+17] G. Rizzo, C. d'Amato, N. Fanizzi e F. Esposito. «Terminological Cluster Trees for Disjointness Axiom Discovery». In: *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I*. Vol. 10249. Lecture Notes in Computer Science. 2017, pp. 184–201. DOI: 10.1007/978-3-319-58068-5_12.

[Rud16] S. Ruder. «An overview of gradient descent optimization algorithms». In: *CoRR* abs/1609.04747 (2016). arXiv: 1609.04747. URL: http://arxiv.org/abs/1609.04747.

[San19] G. Sansaro. «TransOWL: iniezione di conoscenza di fondo nella costruzione di embedding da knowledge graph». Tesi di laurea. Università degli studi di Bari Aldo Moro, 2019.

[Sch73] E. W. Schneider. «Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis». 1973. URL: https://eric.ed.gov/?id=ED088424.

[TC15] K. Toutanova e D. Chen. «Observed versus latent features for knowledge base and text inference». In: *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015*. A cura di A. Allauzen, E. Grefenstette, K. M. Her-

mann, H. Larochelle e S. W. Yih. Association for Computational Linguistics, 2015, pp. 57–66. DOI: 10.18653/v1/W15-4007.

[Tro+16] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier e G. Bouchard. «Complex Embeddings for Simple Link Prediction». In: *CoRR* abs/1606.06357 (2016). arXiv: 1606.06357. URL: http://arxiv.org/abs/1606.06357.

[TS22] I. Tiddi e S. Schlobach. «Knowledge graphs as tools for explainable machine learning: A survey». In: *Artif. Intell.* 302 (2022), p. 103627. DOI: 10.1016/j.artint.2021.103627.

[WQW21] M. Wang, L. Qiu e X. Wang. «A Survey on Knowledge Graph Embeddings for Link Prediction». In: *Symmetry* 13.3 (2021), p. 485. DOI: 10.3390/sym13030485.

[WWT99] P. C. Wong, P. Whitney e J. J. Thomas. «Visualizing Association Rules for Text Mining». In: *IEEE Symposium on Information Visualization 1999 (INFOVIS'99), San Francisco, California, USA, October 24-29, 1999*. IEEE Computer Society, 1999, pp. 120–123. DOI: 10.1109/INFVIS.1999.801866.

[Yu+23] J. Yu, C. Zhang, Z. Hu, Y. Ji, D. Fu e X. Wang. «Geometry-based anisotropy representation learning of concepts for knowledge graph embedding». In: *Applied Intelligence* (2023), pp. 1–22. DOI: 10.1007/s10489-023-04528-1.