

République Algérienne Démocratique et Populaire Ministère de  
l'Enseignement Supérieur et de la Recherche Scientifique

Université de Tébessa



Faculté des Sciences Exactes  
et Sciences de la Nature et de la Vie

Département des mathématiques et informatique

**Mémoire**  
Présenté en vue de l'obtention du diplôme de MASTER

Filière : (Mathématiques/Informatique)

Option : Système d'information

Par

Bouras Khaireddine

**ENHANCING KNOWLEDGE GRAPH  
COMPLETION USING TEXTUAL  
CONTENT**

Date de soutenance: 11/07/2021

Devant le jury

Dr.Benour Akrem	MCA	Université Larbi Tebessi	Président
Dr.Souahi Med Saleh	MCB	Université Larbi Tebessi	Examineur
Dr.Djeddai Ala	MCB	Université Larbi Tebessi	Encadreur

Année Universitaire: **2020/2021**



# ACKNOWLEDGMENTS

I THANK MY SUPERVISOR MR. DJEDDAI ALA FOR GUIDING ME  
THROUGHOUT MY WORK,

I ALSO THANK THE MEMBERS OF THE JURY, THE PRESIDENT  
MR.BENOUR AKREM. AND THE EXAMINER MR. SOUABI MED  
SALEH.FOR AGREEING TO DISCUSS OUR END OF STUDY  
PROJECT TO BENEFIT FROM THEIR EXPERIENCE AND  
ADVICE.

I WOULD LIKE TO THANK ALL MY TEACHERS IN  
THE DEPARTMENT OF MATHEMATICS AND COMPUTER  
SCIENCE.



# DEDICATION

TO MY PARENTS,  
TO MY FAMILY,  
TO MY FRIENDS.

# Abstract

---

Several Knowledge graphs like DBpedia, Freebase, Wordnet and others are far from complete. Thus, Knowledge Graph Completion is a task which has a main objective to complete these graphs with missing knowledge. Every knowledge graph is a set of triples like "Algiers capitalOf Algeria" where the first is the subject entity, the second is the relation and the last is the object entity. The principal tasks are the link prediction and the relation classification where the former predict the relation between two given entities and the latter classifies given triples with true or false. Approaches that use only the observed triples can give best results but they fail in case of unseen entities because the prediction models have trained with only existing triples. Therefore, new directions have been proposed to solve this problem and the main is using an external resource like text, because this later have rich contents.

# Résumé

---

Plusieurs graphiques de connaissances comme DBpedia, Freebase, Wordnet et d'autres sont loin d'être complets. Ainsi, l'achèvement du graphe de connaissances est une tâche qui a pour objectif principal de compléter ces graphes avec des connaissances manquantes. Chaque graphe de connaissances est un ensemble de triplets comme «Alger capitale d'Algérie» où le premier est l'entité sujet, le second est la relation et le dernier est l'entité objet. Les tâches principales sont la prédiction de lien et la classification des relations où la première prédire la relation entre deux entités données et la dernière classe des triplets donnés avec vrai ou faux. Les approches qui n'utilisent que les triplets observés peuvent donner les meilleurs résultats, mais elles échouent en cas d'entités invisibles car les modèles de prédiction se sont entraînés avec uniquement des triplets existants. Par conséquent, de nouvelles directions ont été proposées pour résoudre ce problème et la principale consiste à utiliser une ressource externe comme le texte, car ce dernier a un contenu riche.

# الملخص

العديد من الرسوم البيانية المعرفية مثل DBpedai و Freebase و Wordnet وغيرها بعيدة عن الاكتمال. وبالتالي ، فإن إكمال الرسم البياني المعرفي مهمة لها هدف رئيسي لإكمال هذه الرسوم البيانية مع المعرفة المفقودة. كل رسم بياني معرفي هو مجموعة ثلاثية مثل "الجزائر العاصمة الجزائر" حيث الأول هو الكيان الموضوع ، والثاني هو العلاقة والأخير هو كيان الكائن. المهام الرئيسية هي التنبؤ بالارتباط وتصنيف العلاقة حيث يتنبأ الأول بالعلاقة بين كيانين معينين ويصنف لاحقًا ثلاثيات معينة مع صواب أو خطأ. يمكن للمقاربات التي تستخدم فقط الثلاثيات التي تمت ملاحظتها أن تعطي أفضل النتائج ولكنها تفشل في حالة الكيانات غير المرئية لأن نماذج التنبؤ قد تم تدريبها باستخدام الثلاثيات الموجودة فقط. لذلك ، تم اقتراح اتجاهات جديدة لحل هذه المشكلة والأهم هو استخدام مورد خارجي مثل النص ، لأن هذا لاحقًا يحتوي على محتويات غنية.

# Summary

General introduction.....	42
---------------------------	----

## Chapter 1 : Knowledge graph completion

1.1 Introduction.....	4
1.2 Knowledge Representation Techniques .....	4
1.3 Graph .....	5
1.4 Knowledge graphs.....	6
1.5 Different Knowledge Graphs .....	7
1.6 Problems of Knowledge graphs .....	10
1.7 Knowledge graphs completion .....	10
1.7.1 Conception.....	10
1.7.2 Definition .....	11
1.7.3 Knowledge Embedding.....	12
1.7.4 Knowledge Graph Embedding .....	13
1.7.5 Knowledge Graph completion Classification .....	13
1.7.5.1 Closed environment KGC:.....	13
1.7.5.2 Open environment KGC: .....	13
1.8 Textual information .....	14
1.8.1 Raw texts.....	14
1.8.2 Textual Mentions .....	14
1.8.3 Entity description.....	14
1.9 Knowledge Graph Embeddings Applications.....	15
1.9.1 Abbreviation Disambiguation.....	15
1.9.2 Classifying Entities as Instances of a Class.....	15
1.9.3 Language Translation .....	15
1.9.4 Recommender Systems.....	15
1.9.5 Question Answering.....	16
1.10 Conclusion.....	16

## Chapter 2: Knowledge Graph Completion methods

2.1 Introduction.....	18
2.2 Knowledge Graph Completion Method Based on Translation Model .....	18
2.2.1 TransE .....	18
2.2.2 TransH.....	19
2.2.3 TransR .....	20
2.2.4 Comparison.....	21
2.3 Knowledge Graph Completion Method Based on Semantic Matching Model.....	21
2.3.1 TransW .....	21
2.3.2 TransC .....	22
2.4 Knowledge Graph Completion Method Based on Network Representation Learning .....	22
2.4.1 ConvE .....	22
2.4.2 ProjE .....	23
2.5 Embedding approaches with text data .....	24
2.5.1 Initialize the Entity Embedding.....	24
2.5.2 Augment the Structure-Based Kg Embedding.....	24
2.5.3 Joint Embedding of The Texts and Facts .....	25
2.6 Conclusion .....	26

## Chapter 3: Contribution

<b>3.1 Introduction</b> .....	<b>28</b>
<b>3.2 Word2Vec</b> .....	<b>28</b>
<b>3.2.1 Skip-Gram</b> .....	<b>28</b>
<b>3.2.2 CBOW</b> .....	<b>30</b>
<b>3.2.3 CBOW vs Skip-Gram</b> .....	<b>32</b>
<b>3.3 TransE</b> .....	<b>33</b>
<b>3.4 Problem study</b> .....	<b>33</b>
<b>3.5 Methodology</b> .....	<b>33</b>
<b>3.5.1 Proposed solution</b> .....	<b>33</b>
<b>3.5.2 Proposed design</b> .....	<b>33</b>
<b>3.6 Conclusion</b> .....	<b>35</b>

## Chapter 4: Realization

<b>4.1 Introduction</b> .....	<b>37</b>
<b>4.2 Implementation</b> .....	<b>37</b>
<b>4.2.1 Dbpedia</b> .....	<b>37</b>
<b>4.2.2 TorchKGE</b> .....	<b>38</b>
<b>4.2.3 Google Collab</b> .....	<b>39</b>
<b>4.2.4 Python</b> .....	<b>40</b>
<b>4.3 Conclusion</b> .....	<b>40</b>

<b>General conclusion</b> .....	<b>42</b>
---------------------------------	-----------



# *List of figures*

---

<b>Fig 1. 1: Evolution of Knowledge Representation Techniques.....</b>	<b>3</b>
<b>Fig 1. 2: Example of KG: things, not strings.....</b>	<b>4</b>
<b>Fig 1. 3: Example of KGC.....</b>	<b>12</b>
<b>Fig 2. 1: TransE model.....</b>	<b>20</b>
<b>Fig 2. 2: TransH model .....</b>	<b>21</b>
<b>Fig 2. 3: TransR model.....</b>	<b>22</b>
<b>Fig 2. 4: ConvE model .....</b>	<b>24</b>
<b>Fig 3. 1: Skip-gram model.....</b>	<b>30</b>
<b>Fig 3. 2: CBOW model .....</b>	<b>32</b>
<b>Fig 3. 3: CBOW vs SG.....</b>	<b>33</b>
<b>Fig 3. 4: Proposed design .....</b>	<b>40</b>
<b>Fig 4. 1: Our proposed solution.....</b>	<b>43</b>
<b>Fig 4. 2: Dbpedia dataset exemple.....</b>	<b>45</b>
<b>Fig 4. 3: TorcheKGE logo .....</b>	<b>46</b>
<b>Fig 4. 4: Google colab logo .....</b>	<b>47</b>
<b>Fig 4. 5: Python logo .....</b>	<b>47</b>

# *List of tables*

---

<b>Table 1. 1:Date and creation method of knowledge graphs.....</b>	<b>6</b>
<b>Table 1. 2:A comparison of knowledge graph components .....</b>	<b>7</b>
<b>Table 2. 1: TransE vs TransH vs TransR.....</b>	<b>22</b>
<b>Table 4.1 dataset used in the experiments.....</b>	<b>38</b>

# *List of abbreviations*

---

<b>KG</b>	:	knowledge graph
<b>KGC</b>	:	knowledge graph complement
<b>KGE</b>	:	knowledge graph embedding
<b>ILS</b>	:	Inform ledge System
<b>DBOW</b>	:	Distributed bag of words
<b>SG</b>	:	Skip Gram
<b>Word2vec</b>	:	Word to vector
<b>Doc2vec</b>	:	Document to vector
<b>NLP</b>	:	Naturel language processing
<b>RDF</b>	:	Resource Description Framework
<b>OWL</b>	:	Ontology Web Language

# *General introduction*

---

## **1. Introduction**

Techniques that define entities and relationships of a knowledge graph (KG) in a continuous low-dimensional space are called KG embed learning or knowledge representation learning. However, many cognitive graphs are far from complete and deficient in KG.

Recently, the search for textual information in KG embedding has attracted much interest due to the rich semantic information provided by texts.

KG in recent years have experienced rapid development. Some exemplary achievements have been created and published. KG offers a structural model for storing human knowledge. It is an organized representation of relational facts, consisting of entities, relationships, and descriptions. Entities represent specific objects and abstract concepts, relationships represent relationships between entities, and descriptions identify or describe entities. Knowledge, also called truth, is stored privately as a tripartite entity (main entity; relation; tail entity) within the Schema Resource Description Framework (RDF).

## **2. Problem Definition**

Approaches using only the observed triples can give the best results but fail in the case of sparse entities because the prediction models have been trained using only the existing triples.

## **3. Proposed Solution**

So, we propose in this work to suggest a new direction to solve this problem by using an external resource like text, because this later has rich contents. We propose to use the description of the entities to discover new relations between them using word embeddings.

The main objectives of the work are as follow:

- Extending the translation-based approaches like TransE with textual descriptions.
- Using WordToVec to calculates the scores between entities using

their descriptions in order to discover the degree of relations between entities.

- Extending the scoring function of TransE with WordToVec scores.
- Testing our proposal with real dataset.

#### **4. The structure of the thesis**

- 1) The first chapter gives definition of Knowledge Graph and introduces the context of KnowledgeGraph Completions.
- 2) The second chapter cites the Knowledge Graph Completions methods in two parts. The first one is for the general methods and the second one is for the works that use the textual contents.
- 3) The third chapter concentrates on our contribution.
- 4) the fourth chapter: realization of our proposal, the results and the discussion
- 5) The thesis is ended with conclusion and future directions.

# Chapter 1

---

## *Knowledge Graph Completions*

## 1.1. Introduction

Knowledge graph completion (KGC) is a hot topic in knowledge graph construction and related applications, which aims to complete the structure of knowledge graph by predicting the missing entities or relationships in knowledge graph and mining unknown facts. Starting from the definition and types of KGC, existing technologies for KGC are analyzed in categories. From the evolving point of view, the KGC technologies could be divided into traditional and representation learning based methods. The former mainly includes rule-based reasoning method, probability graph model, such as Markov logic network, and graph computation-based method. The latter further includes translation model based, semantic matching model based, representation learning based and other neural network model-based methods. To fully understand this technique, we provide in this chapter a background about knowledge graph completions tasks with concentrating in embedding models.

## 1.2. Knowledge Representation Techniques

The developments in information representation techniques are shown in figure 1. Knowledge representation and retrieval techniques mentioned so far deal with information as connected words at the time of input and processing. There is a need to develop new information representation technique that could incorporate innovative and intelligent knowledge retrieval properties into the system.

The usage of concepts has been restricted to representation of words. However, to represent the concept there is a need to connect with related sub-concepts e.g. Cow, as a word means nothing unless it is associated with its properties. Thus, the set of connected sub-concepts make a concept. Also, these systems fail to provide dynamic connectivity between existing nodes, wherein any new relationship needs to be specified using separate rules. Many of the social networks like Google, Facebook, and Twitter have included graph databases.

Graph databases as mentioned earlier provide explicit connectivity between nodes whereas human brain network does not provide a fixed and explicit connectivity between the neurons [1].

The researchers have believed that the information in the human brain, as well as information in knowledge and information systems, is stored as a network of interconnected nodes. However, the human brain network and human-made knowledge systems differ considerably in the way nodes are structured, connections between the nodes are made and the efficiency with which knowledge is retrieved. In the human brain, network links have varying properties that help in their fast or slow knowledge retrieval [2]. There is a need to develop a knowledge system that can



## Chapter 1: Knowledge Graph Completion

provide for an autonomous node with an ability to decide the subsequent connectivity.

In addition, connectivity between nodes is not just an assigned string of relationship but where the intelligence of the network lies.

Another promising approach for the development of intelligent knowledge system is provided by Informledge System (ILS). This knowledge system provides intelligent knowledge retrieval from the stored information by virtue of ILS autonomous nodes and the multilateral links [3].

It follows a distinct way of representation for its nodes with four quadrant structure to provide processing capabilities unlike the nodes provided by the other knowledge systems.

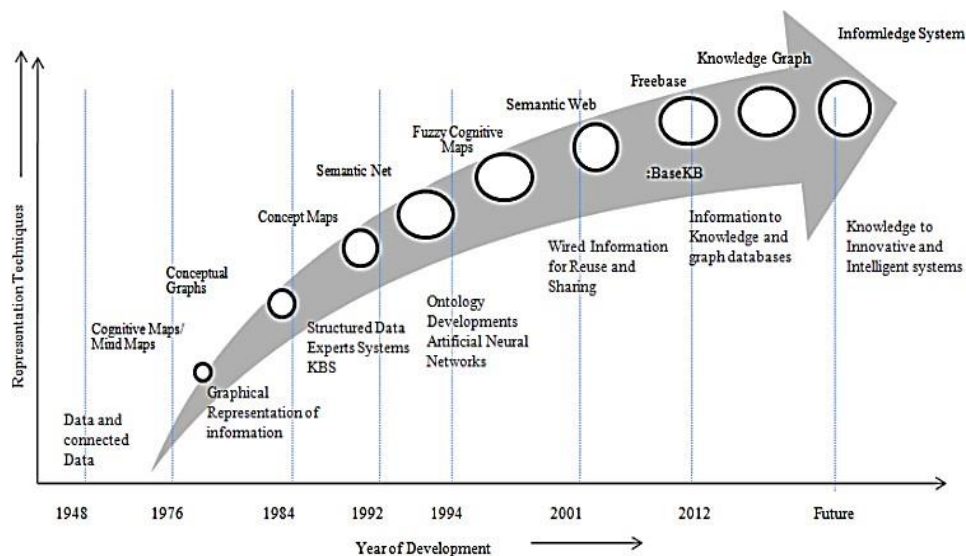


Fig 1. 1: Evolution of Knowledge Representation Techniques [4]

For summary, to represent a knowledge we usually we use the triple (head, relation, tail), where the head and tail are entities. For example, (Maqam Echahid, location, Algeria). We can use the most famous technique "one-hot vector" to represent this knowledge. But the entity and relation are too many and the dimensions are too big. Also, one-hot vector cannot capture the similarity if two entity or relation is close. By the inspiration of Word Embedding models in 2013 such as Word2Vec, the representation of the entity and the relation has become an distributed representation picks up the semantic and syntactic similarity of knowledges.

### 1.3. Graph

A graph is a structure used to represent things and their relations. It is made of two sets — the set of nodes (also called vertices) and the set of edges (also called arcs). Each edge itself connects a pair of nodes indicating that there is a relation between them. This relation can either be undirected, e.g., capturing symmetric relations between nodes, or directed, capturing asymmetric relations.

Graphs can be either homogeneous or heterogeneous. In a homogeneous graph, all the nodes represent instances of the same type and all the edges represent relations of the same type. In contrast, in a heterogeneous graph, the nodes and edges can be of different types.

Another class of graphs that is especially important for knowledge graphs are multigraphs. These are graphs that can have multiple (directed) edges between the same pair of nodes and can also contain loops. These multiple edges are typically of different types and as such most multigraphs are heterogeneous. Note that graphs that do not allow these multiple edges and self-loops are called simple graphs.

## 1.4. Knowledge graphs

knowledge Graph (KG) is a knowledge base that uses a graph-structured data model or topology to integrate data. KGs are often used to store interlinked descriptions of entities (objects, events, situations or abstract concepts) with free-form semantics.

From strings to things, knowledge graphs aim to structure what is known about the world. From powering up search to quick summaries of known entities, it makes information that much easier to discover and enables world-aware inferences

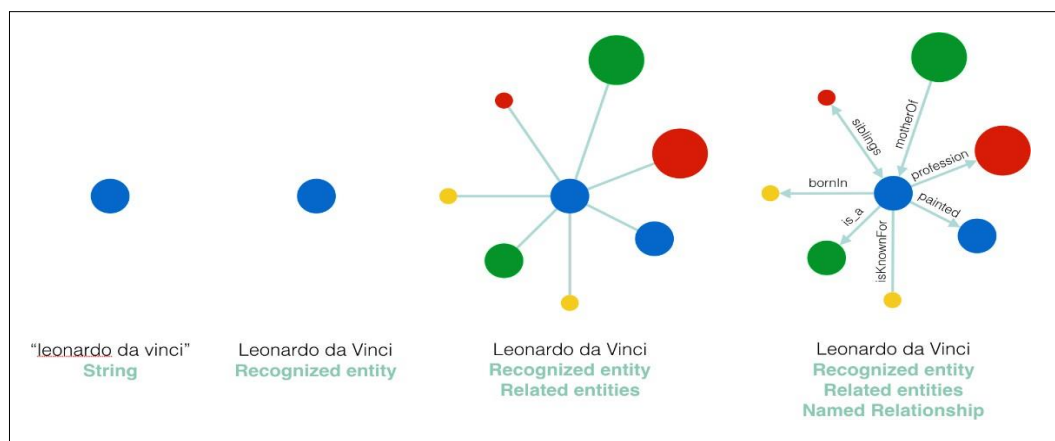


Fig 1. 2:Example of KG: things, not strings [5].

## 1.5. Different Knowledge Graphs

Cognitive KGs are numerous and differ from each other in several characteristics. Among the most popular non-specialized cognitive KGs available to the public, we mention the following:



**DBpedia** : is the most popular and prominent KG . The project was initiated by researchers from the Free University of Berlin and the University of Leipzig, in collaboration with OpenLink Software. Since the first public release in 2007, DBpedia is updated roughly once a year. DBpedia is created from automatically-extracted structured information contained in the Wikipedia, such as from infobox tables, categorization information, geo-coordinates, and external links. DBpedia contains many links to other datasets and is used extensively in the Semantic Web research community, but is also relevant in commercial settings: companies use it to organize their content [6] .



**Freebase**: Freebase: Freebase is a KG announced by Metaweb Technologies in 2007 and was acquired by Google . on July 16, 2010. In contrast to DBpedia, Freebase had provided an interface that allowed end-users to contribute to the KG by editing structured data. Besides user-contributed data, Freebase integrated data from Wikipedia uses a proprietary graph model for storing also complex statements [7] .



**OpenCyc** : The Cyc project started in 1984 as part of Microelectronics and Computer Technology Corporation. The aim of Cyc is to store (in a machine-processable way) millions of common sense facts such as “Every tree is a plant.” While the focus of Cyc in the first decades was on inferencing and reasoning, more recent work puts a focus on human-interaction such as building question answering systems based on Cyc. Since Cyc is proprietary, a smaller version of the KG called OpenCyc was released under the open source Apache license.



**Wikidata** : is a project of Wikimedia Deutschland which started on October 30, 2012. The aim of the project is to provide data which can be used by any Wikipedia project, including Wikipedia. Wikidata does not only store facts, but also the corresponding sources, so that the validity of facts can be checked. Labels, aliases, and descriptions of entities in Wikidata are provided in more than 350 languages. Wikidata is a community effort, i.e., users collaboratively add and edit information. Also, the schema is maintained and extended based on community agreements [8].



**YAGO**: Yet Another Great Ontology has been developed at the Max Planck Institute for Computer Science in Saarbrücken since 2007. YAGO comprises information extracted from the Wikipedia (e.g., categories), WordNet (e.g., synsets, hyponymy), and GeoNames [9] .

Some of the differences are due to the date and method of creation as shown in Table 1.

**Table 1. 1:Date and creation method of knowledge graphs.**

Date	Name	Creation Method
1984	Cyc	Handwritten by Experts
2007	Freebase	Crowd-Sourced
2007	DBpedia	Automated from Structured information in Wikipedia Project
2008/2017	YAGO	Automated from Structured & Semi-Structured Sources
2012	Wikidata	Crowd-Sourced

Another in-depth comparison was made between the previous Knowledge graphs, where each Knowledge Graph was created with different vocabulary rules applied. These rules lead to significant differences between the cognitive diagrams in the vocabulary of relationships, predicates, and categories, as shown in the following table:

Table 1. 2:A comparison of knowledge graph components

	<b>DBpedia</b>	<b>Freebase</b>	<b>OpenCyc</b>	<b>Wikidata</b>	<b>YAGO</b>
<b>Of Triplets</b>	411885690	3124791156	2412520	748530833	1001461792
<b>Of Classes</b>	736	53092	116822	302280	569751
<b>Of Relations</b>	58776	70902	18028	1874	106
<b>Unique Predicated</b>	60321	784977	165	4839	88736
<b>Of named-Entities</b>	4298433	49947799	41029	18697897	5130031
<b>Of instances</b>	20764283	115880761	242383	142213806	12291250
<b>Avg of named entiries per class</b>	5840.3	940.8	0.35	61.9	9
<b>Unique non literals in object position</b>	83284634	189466866	423432	101745685	17438196
<b>Unique literals in object position</b>	161398382	1782723759	1081818	308144682	682313508

Table 2 shows that Freebase has the most relations and predicates, but many of those are not useful. A third of its relations are declared to be inverses of other relations using the markup owl:InverseOf .

Inverse predicates can also occur. The inverse relations and predicates of Freebase can lead to misleading results when used to test relation and predicate prediction algorithms. Additionally, Freebase is becoming outdated. The knowledge graph was made read-only as of March 31, 2015.

Wikidata is also curated by a community but new predicates are only accepted by the committee, this limitation puts Wikidata at only 4839 unique predicates. The number of relations is also a low 1874. DBpedia, in contrast, has 58,776 relations created from Wikipedia.

YAGO is constructed by machine learning instead of crowd sourced. It has the fewest relations at 106. Given YAGO's ability to extend the triplet to store temporal and spatial information, it avoids dedicated relations such as "distanceToAlgeria". Interestingly, YAGO has the second largest number of predicates at 88,736.

There is also a difference in the creation of classes. YAGO automatically creates classes from structured and semi-structured sources. As a result, YAGO has 569,751

## Chapter 1: Knowledge Graph Completion

---

classes with an average of 9 named-entities per class. DBpedia, which manually creates classes, has only 736. There are many non-structural differences that also affect knowledge graph choice. An example is the number of knowledge graph entities related to a specific subject. How often the knowledge graph is updated is also a consideration.

### 1.6. Problems of Knowledge graphs

Most of KGs studied in the last section are graph-structured knowledge bases whose facts are represented in the form of relations (edges) between entities (nodes). This can be represented as a collection of triples ( headentity , relation, tailentity ) denoted as (h, r, t), for example (Algiers, CapitalOf, Algeria) is represented as two entities: (Algiers and Algeria along with a relation CapitalOf linking them. KGs are important sources in many applications such as question answering [10], dialogue generation [11] and recommender systems [12]. Containing billions of triples though, KGs still suffer from incompleteness, that is, missing a lot of valid triples [13].

### 1.7. Knowledge graphs completion

#### 1.7.1. Conception

Most knowledge graphs are created manually or semi-automatically, suffer from the problem of not discovering all the implicit entities and relationships, thus incompleteness becomes a universal problem in nearly all cognitive graphs. The goal of KGC is to solve incompleteness and scattering problems resulting from missing states or links in cognitive graphs. The knowledge graph completion technique complements the structure of the graph by predicting states of knowledge (entities, relationships, attributes, etc.), digging for lost entities or relationships, or discovering new facts. It is an essential tool for improving the quality of cognitive graphics.

For example, in the the knowledge graphs represented by Resource Description Framework (RDF), the triples like “entities-attributes-attribute values” or “head entity-relationship-tail entity” are used to describe nodes, edges and attributes in a graph network, in which the node corresponds to the entity in the real world, and edge represents all kinds of relations between entities.

In this way, the knowledge graph completion problem can be converted into estimating the missing parts of the triples by using the methods like semantic similarity. According to the missing parts in triples, knowledge graph completion can be divided into three kinds of specific tasks:

1. given the head entities and relationships in a triples, predict the corresponding tail entities,
  - such as (Algies, capitalOf, ?);
2. given the relationship and tail entities, predict the corresponding head entities,
  - such as (?, capitalOf, Algeria);
3. given the head and tail entities, and predict the relationship between them,
  - such as (Algeria, ?, country).

That is, from any two given elements in a triple and the third element can be deduced. For specific application, knowledge graph completion includes:

- ✓ link prediction,
- ✓ entity prediction,
- ✓ relation prediction,
- ✓ attribute prediction
- ✓ ...

### 1.7.2. Definition

The definition of KG given above allows us to specify an edge of KG with a triplet of elements  $(h, r, t) \in E \times R \times E$  where the head ( $h$ ) and the tail ( $t$ ) entities are elements of  $E$  and  $r$ , which is directed, is a type of relation of  $R$ . Notice that the order is important because not every relation is bidirectional. Formally, we define KGC as the task that tries to predict any missing element of the triplet  $(h, r, t)$ . In particular, we talk about:

- ✓ **entity prediction** when an element between  $h$  or  $t$  is missing ( $(?, r, t)$  or  $(h, r, ?)$ );
- ✓ **relation prediction** when  $r$  is missing  $(h, ?, t)$ ;
- ✓ **triplet classification** when an algorithm recognizes whether a given triplet  $(h, r, t)$  is correct or not.

Technically, KGC is very similar to link prediction in social network analysis: both of them try to complete an incomplete network. In addition, KGC is challenging for the following reasons:

- ✓ it is not trivial to create a KG;
- ✓ every entity could have a variable number of attributes (non-unique specification);
- ✓  $R$  could contain different types of relation (multi-layer network);

- ✓ a KG changes over time (evolution over time).

In order to tackle KGC, different approaches were developed in past years: for instance, entity resolution (ER), probabilistic soft logic (PSL) and knowledge graph embedding (KGE). The first one tries to group entities referring to the same underlying entity while the second approach uses probabilistic inference. Even though they are extremely interesting topics, we focus only on KGE in this article.

The idea behind the embedding is widely used in mathematics: if we need to study properties of elements of an unknown space, we find a function that maps each element of the unknown space into a known space by preserving some relationships.

Figure 3 presented an example of KGC, based on known relations (Green lines) between entities, a KGC algorithm should infer the unknown one (Red line).

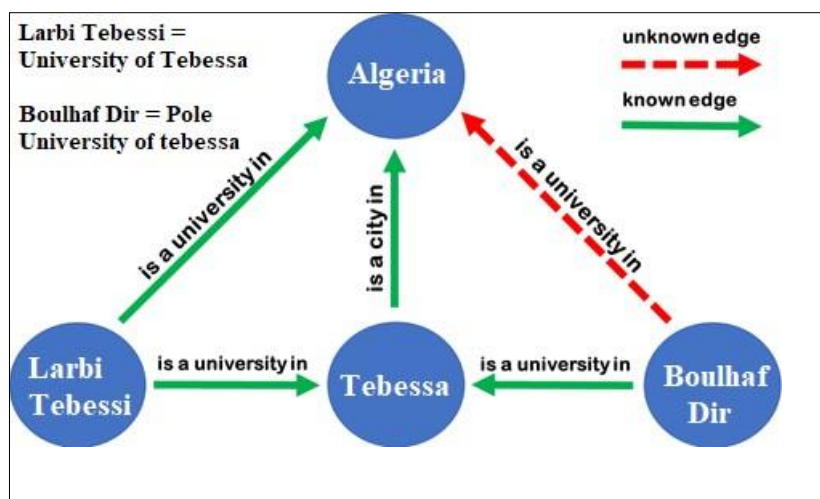


Fig 1. 3: Example of KGC [15].

### 1.7.3. Knowledge Embedding

A knowledge graph is embedded into a low-dimensional continuous vector space while certain properties of it are preserved [16]. Generally, each entity is represented as a point in that space while each relation is interpreted as an operation over entity embeddings. For instance, TransE

[17] interprets a relation as a translation from the head entity to the tail entity. The embedding representations are usually learnt by minimizing a global loss function involving all entities and relations so that each entity embedding encodes both local and global connectivity patterns of the original graph. Thus, we can reason new facts from learnt embeddings. Word Embedding. Generally, word embeddings are learned from a given text corpus without supervision by predicting the context of each word



## Chapter 1: Knowledge Graph Completion

---

or predicting the current word given its context [18]. Although relations between words are not explicitly modeled, continuous bag-of-words (CBOW) and Skip-gram [19] learn word embeddings capturing many syntactic and semantic relations between words where a relation is also represented as the translation between word embeddings.

### 1.7.4. Knowledge Graph Embedding

Knowledge graph embedding (KGE) is the task of completing the knowledge graphs by probabilistically inferring the missing arcs from the existing graph structure. KGE differs from ordinary relation inference as the information in a knowledge graph is multi-relational and more complex to model and computationally expensive.

### 1.7.5 Knowledge Graph completion Classification

The Knowledge Graph completion is divided according to the task scenarios in two parts:

#### 1.7.5.1 Closed environment KGC:

also called the static knowledge graph completion, if the entities and relationships involved in the completion process belong to the original knowledge graph.

At present, a large number of existing knowledge graph completion models are based on the closed environment hypothesis [20]. In such cases, all entities and relationships are supposed to be existing in the same knowledge graph, and graph completion can only be achieved by mining the potential connections between existing entities, instead of adding new entities and related relationships to the existing graph. The knowledge graph completion in closed environment relies heavily on the existing connected structure of the knowledge graph, which cannot achieve prediction for the weak connections and new entities and also cannot expand graph structure well. So, the knowledge graph completion in closed environment is mostly applicable to the domain knowledge graph with small scale and slow update. And the KGC under closed environment does not make full use of external data for missing completion, resulting in insufficient information and strong limitations in usage.

#### 1.7.5.2 Open environment KGC:

also called dynamic knowledge graph completion. the knowledge graph completion model in open environment provides a method to predict external entities and weakly connected entities. Most of the existing large-scale knowledge graphs are constantly updated and expanded through linking external entities to adapt to the explosive growth of information. The knowledge graph completion in an open environment is

## Chapter 1: Knowledge Graph Completion

---

relatively difficult to establish a connection between the local knowledge graph and the outside world, due to the wide range of alternative knowledge. But it has more advantages when expanding the scale of the knowledge graph [21]. Based on the knowledge graph completion in an open environment, the research in the field of dynamic updating of the knowledge graph has started.

### 1.8 Textual information

Various information in textual form can be used to include KG. Among these textual formulas, we can distinguish 3 types:

#### 1.8.1 Raw texts

Raw texts such as news and Wikipedia articles are characterized by the fact that they contain a lot of semantic information. However, these texts are weak in capturing direct links with KG, as entities appear at random places in the document with an unknown address.

**Note:** It is difficult to extract the required information from the primary texts, which makes the representation of entities and relationships with very poor accuracy.[22]

#### 1.8.2 Textual Mentions

Text mentions denote the odd sentence containing the entity pairs derived from ClueWeb. The sentence is processed by the dependency analyzer and represented as lexical dependency paths. Then the path is defined as the textual relationship between the pair of entities.

textual mentions are constructed to express the relationship of entities and introduce noisy information. Some authors [23] aims to provide precise textual mentions for the following relationship representation learning. An extractor is used to collect precise textual mentions for each fact (h, r, t). All the sentences with h and t are collected as mentioned in the candidate's text at the beginning. [22]

**Note:** The sentence is kept as the exact textual mentions only if it meets one of the conditions:

- have the word hyponym / synonyms of r,
- have similar words with relation names.

#### 1.8.3 Entity description

The entity description is considered to have a strong association with the KG. These entity descriptions are promising texts for the representation of entities. Classic textual

functionalities such as TF IDF could be used to extract keywords from the description of the entity.

**Note:** The flaw of the entity description is that not all entities have the associated description in KG. Other textual information such as the name of the entity is also taken into account although it contains little semantic information. [22]

## **1.9 Knowledge Graph Embeddings Applications**

Embeddings greatly simplify the use and completion of knowledge graphs. By condensing the information into a dense matrix, the information is easier to use and store. Using a knowledge graph embedding the probability of a relation can be easily calculated using the scoring function. Additionally, a comparison of different entity vectors gives the similarities of different entities. The similarity of relations can be found the same way. The labeled training data needed for tasks such as entity disambiguation can be reduced or eliminated because of these benefits. This section discusses the benefits of using a knowledge graph embedding instead of the knowledge graph itself.

### **1.9.1 Abbreviation Disambiguation**

Abbreviation disambiguation was previously done by training a neural network on costly handannotated data. However, this approach does not work for abbreviation not seen in the labeled dataset. Embeddings are more flexible. Both the embedding and its possible longform were embedded using the surrounding context. The embedded vectors were compared, and the abbreviation connected to its most similar longform.

### **1.9.2 Classifying Entities as Instances of a Class**

The relation is A is part of the knowledge graph embedding. Classification can be treated as a form a link prediction. The head is the entity to be classified and the relation is IsA. The probability that the entity is an instance of different classes can be calculated and the most likely class found.

### **1.9.3 Language Translation**

Some research uses knowledge graph embeddings for translation. This involves creating separate embedded knowledge graphs for each language. The relations and entities of the different embeddings are aligned. This alignment is done using crowdsourcing in knowledge graphs such as Wikidata and DBPedia.

### **1.9.4 Recommender Systems**

Sparse data is a known problem when working with recommender systems. The solution is to create a knowledge graph embedding containing the items. A book is embedded in a knowledge graph along with its summary and image. The structure, textual, and visual knowledge of the book can be combined to embed the book into a knowledge graph. The user can be recommended books similar to what the user liked in the past. Multiple users also can be embedded into the knowledge graph based on their history.

### **1.9.5 Question Answering**

In question answering a question is asked in natural language and answered using the information contained in a knowledge graph. Most questions can be answered by a machine if the question's corresponding head entity and relation can be identified. A word embedding model is used to embed the question's relation and the question's entity. These two embeddings are compared with the knowledge graph embedding to find the most likely match. This identifies the question's corresponding head entity and relation so the machine can answer the question.

### **1.10 Conclusion**

In this chapter we have provided a comprehensive overview of KGC which generally aims to predict missing entities and relationships in a Knowledge Graph. We first presented the history of the development of data representation, then we got acquainted with the concept of graphs, Knowledge graphs, knowledge graphs completion, the classification of knowledge graph completion, and finally we got to know most of the applications of this technology.

In the next chapter we present a study of the various text-related KGC applications.

# Chapter 2

---

*Knowledge Graph Completions methods*

## 2.1 Introduction

The concept of a knowledge graph was first proposed by Google in 2012, which was defined as a broad-based knowledge base made up of a large number of entities and relationship vessels between them. Knowledge graph, as a semantic network, has been used extensively in natural language processing. There are many models for completing the Knowledge Graph, including traditional and text-based methods. In this chapter, we present a study related to the main models that use text to complete a Knowledge Graph.

In this thesis we are interested in a set of methods based on knowledge representation learning, which in turn are divided into the translation, semantic matching and Network Representation Learning approaches. We give also the embedding approaches that extended with textual data.

## 2.2 Knowledge Graph Completion Method Based on Translation Model

The translation paradigm is the most representative classic method for learning to represent knowledge. The most famous example is the Word2Vec [27] algorithm, thus proposing the phenomenon of static vector translation. In other words, distributive word representation picks up some kind of semantic relationship itself. According to the phenomenon of transient stability, the more representative classic translation model has been proposed TransE [28], and has led a large number of researchers to study Trans series models, which include representative enhanced models TransH [29] and TransR [31].

The goal behind the translation model is to find valid triples as the process of translating entities and then reduce the loss function to see the representation of entities and relationships

### 2.2.1 TransE [28]

Given a training set  $S$  consisting of triples  $(h, r, t)$ , in the head and tail entity  $h, t \in E$ ,  $E$  is entity set, and  $r \in R$ ,  $R$  is relationship set. The main idea of TransE is that, if the triplet  $(h, r, t)$  is true, then the sum of the vector representations of head entity and relation is close to the vector representations of the tail entity; otherwise, it is far away, that is, when the triplet is formed,  $h+r \approx t$ , as shown in (Figure 1). From the above ideas, the score function  $f_r(h, t) = -\|k_h + r - t_k\|_1 / 2$  [28] of the TransE model can be obtained, which represents the Euclidean distance between the head entity

and the tail entity in low-dimensional continuous space

- **Advantages of TransE:**

- ✓ Efficient and concise model,
- ✓ Has good prediction effect

- **Disadvantages of TransE:**

- ✓ The flexibility of the TransE method is poor: it depends on the Euclidean distance as the measure of distance in the result function, and the same weight in the calculation is assigned to each feature vector, which affects the accuracy of the knowledge representation.
- ✓ A simple model with limitations in dealing with reflexive relationships: Such as face-to-many, many-to-one, and many-to-many relationships [31], [32], which cannot distinguish between entities having the same relationship [33].

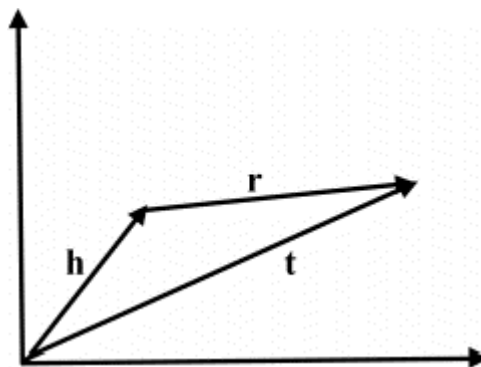


Figure 1: TransE model [28]

### 2.2.2 TransH[32]

To overcome the disadvantages of the previous model, a TransH model has been proposed that integrates knowledge into the hyperbolic level of a given relationship [32], as shown in (Figure 2). TransH learns an additional vector to mapping  $W_r$  for each relationship, which is used to map entities to the hyper level specified for Relationship.

That is, for triples  $(h, r, t)$ , the representation of the head and tail entities is first set to the hyperboloid.

If triple is true, then the relationship vector  $r$  can be used on the hyperplane to connect the head-to-tail vector mapped with this super-plane. At this time at the superscript  $fr$   $(h, t) = -kh\perp + r - t\perp k$  [32].

- **Advantages of TransH:**

- ✓ Mitigates the problem that TransE model cannot alter complex relationships well.

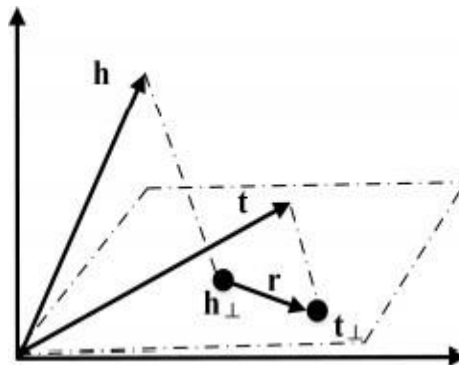


Figure 2: TransH model [29]

### 2.2.3 TransR[31]

The TransR model [31] represents entities and relationships in separate entity and relationship spaces according to specific relationships, as shown in Figure 3. That is, for a triple  $(h, r, t)$ , the representation of the head and tail entities is first mapped to the space corresponding to a specific relationship, and gets  $h_1 = Mrh$ ,  $t_1 = Mrt$  [34].

If the triple is established, the relationship vector is regarded as the transfer between entity vectors in the corresponding space. The score function is defined as  $f_r(h, t) = -\|h_1 - r - t_1\|_2^2$  [34].

- **Advantages of TransR**

- ✓ Has some improvements compared to the original translation model.

- **Disadvantages of TransR**

- ✓ Head and tail entities connected by the same relationship may differ greatly in type or attribute, which will have a certain impact on prediction accuracy.
  - ✓ The projection matrix in TransR is formed according to different relationships, ignoring the impact of different types of entities.
  - ✓ TransR, while introducing the projection matrix, increases the number of parameters and computation complexity.



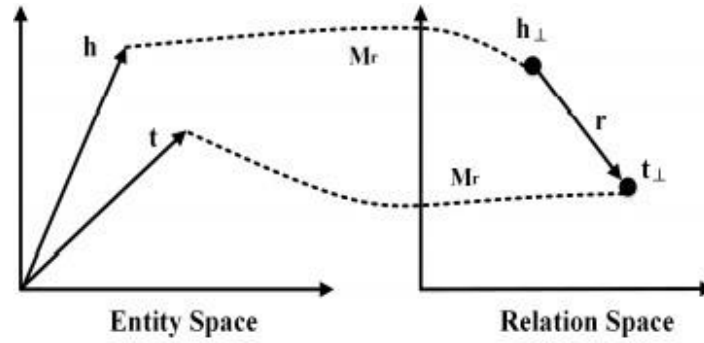


Figure 3: TransR model [31]

### 2.2.4 Comparison

The table below show a comparison between Knowledge Graph Completion Method Based on Translation Model:

Model	Score function	characteristics
TransE	$f_r(\mathbf{h}, \mathbf{t}) = -\ \mathbf{h} + \mathbf{r} - \mathbf{t}\ _{\frac{1}{2}}$	Include multiple relationships
TransH	$-\ (\mathbf{h} - \mathbf{w}_r^\top \mathbf{h} \mathbf{w}_r) + \mathbf{r} - (\mathbf{t} - \mathbf{w}_r^\top \mathbf{t} \mathbf{w}_r)\ _2^2$	Assigning entities to superplanes corresponding to specific relationships
TransR	$f_r(\mathbf{h}, \mathbf{t}) = -\ \mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\ _2^2$	Mapping different relationships to different semantic spaces

Table 1: TransE vs TransH vs TransR

## 2.3 Knowledge Graph Completion Method Based on Semantic Matching Model

The semantic matching model uses the result function based on semantic similarity to mine the possible semantic association between entities and relationships. By integrating the representation of entities and relationships in the vector space, it can obtain the possibility of new facts, so as to predict new knowledge and complete the cognitive graph [35]. Here are some semantic matching based representations model:

### 2.3.1 TransW[36]

The TransW [36] bases on word embeddings to compose knowledge graph embeddings and learns a function mapping from the word embedding space to the knowledge embedding space. Entities and relations are represented in the form of linear combinations of word embeddings in this model, which can detect unknown

facts.

- **Advantages of TransW**

- ✓ Entities and relations are represented in the form of linear combinations of wordembeddings.

### 2.3.2 TransC[37]

The TransC Model [37] combines structured information with entity concepts to improve KGE models, providing semantic similarity  $n$  to measure the characteristic of entity semantics using concept information. The relationships here consist of two sets of concepts:

- 1) the head concept set  $C^{head}_r$
- 2) tail concept set  $C^{Tail}_r$

The semantic similarity of the relationship, main entity, relationship and tail entity is used to measure the distinction between the semantic of the entity with the concept information.

- **Advantages of transC**

- ✓ TransC regards each entity concept as a concept vector and each entity as a set of concept vectors.

## 2.4 Knowledge Graph Completion Method Based on Network Representation Learning

Neural networks represent a fundamental solution in many areas as they are characterized by associative storage and high-speed optimization. Traditional distance-based and semantic matching models cannot meet KGE requirements. Therefore, a neural network model was introduced to have better and more effective entities and relationships. These forms also fall into two subcategories (with or without additional information).

### 2.4.1 ConvE[39]

A convolutional neural network model has been proposed to complete the junction prediction and to complete the knowledge edge graph. When completing a large knowledge edge graph some shallow models are often used for the correlation prediction task. But these types of models lack the ability to extract deeper basic

## Chapter 2: Knowledge Graph Completion methods

features, which results in poor prediction effects. To enhance the ability of models to extract features, the complexity and number of parameters of the models usually increases, the number of parameters is proportional to the number of entities and relationships. These methods cannot be used for a large scale cognitive graph, and the method of scaling the neural network layer is likely to cause a problem such as overfitting [38]. In order to solve the discrepancy between scale of data and overprocessing, the literature [39] has proposed ConvE, a two-dimensional convolutional neural network model with high parameter efficiency and scalability, for implementing representational learning of cognitive graph, and predicting new knowledge in knowledge graph (Figure 4).

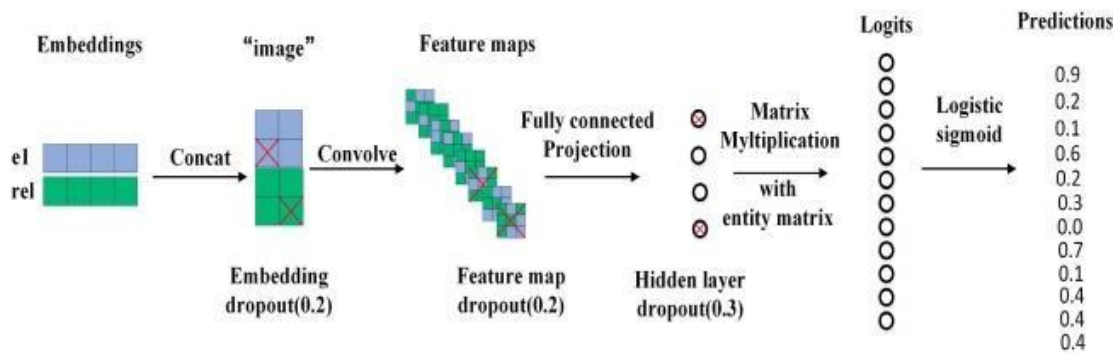


Figure 4: ConvE model [39]

ConvE first reconfigures the main entity embedding and merging relationship, sequencing it into an input matrix for the 2D warp layer, which then returns the feature map tensor. Then, the tensor is directed and projected into the space of the  $k$  dimension through a parameterized linear transformation by the matrix  $W$  and is finally matched with the inclusion of the tail entity through an internal product.

### ▪ Advantage of ConvE

- ✓ ConvE can only manipulate the regular Euclidean data such as images (2-dimensional grid), texts (1-dimensional sequence)

### 2.4.2 ProjE

ProjE [40] aims to complete lost information in KG by learning the co-inclusion of entities and edges and making modifications to the loss function, thus improving the KGE model through trivial changes in network architecture and eliminating complex feature engineering. By means of a acquired combinatorial factor, the embedding vectors of the main entities and relationships are combined into a target vector, which

## Chapter 2: Knowledge Graph Completion methods

also distributes the projection of the candidate entities for the order list, and the higher ranked candidates are the correct entities. ProjE is defined as  $e \oplus r = De_e + Dr_r + bc$ , where  $De$ ,  $Dr$  are diagonal matrices that act as a universal matrix Entity and relationship weights, respectively, and  $bc \in \mathbb{R}^k$  is the compound bias. Next, the inclusion drop function is defined as  $h(e, r) = g$ .

- **Advantage of ProjE**

- ✓ Compared to TransE, ProjE saves many transformation matrix calculations due to integrations

### 2.5 Embedding approaches with text data

The main difference between text-improved and the text-based KG embedding techniques is that the former does not cover the three key elements and focus on enhancing the structure-based KG embedding with the textual information. We roughly categorize the methods based on the usage of the texts.

The text-based KG embedding techniques integration have the distinction of text-improved in that they cover the three main elements and focus on enhancing the hierarchy-based KG inclusion with textual information.

In the following, we present roughly categorizing styles based on text usage. We categorize styles roughly based on text usage.

#### 2.5.1 Initialize the Entity Embedding

Entity embeddings are usually randomly configured in existing approaches. However, the entity name contains little semantic information.

##### 2.5.1.1 DISTMULT [41]

takes advantage of the previously trained embeddings released by Word2Vec. Word2Vec and GloVe were trained in [41] on a large group containing many entities from FreeBase. DISTMULT introduced another method that deals with the entity as the word / phrase in the body and learns the distributed embedding of the entity for the initialization directly.

- **Advantages**

- ✓ Initialization with text embedding is helpful for embedding a Knowledge Graph.

- **Disadvantage**

- ✓ Not all initializations with the word include improve performance in the downstream application.

## 2.5.2 Augment the Structure-Based Kg Embedding

To capture the implicit relationship between entities and attributes:

### 2.5.2.1 DEKE [43]

concatenate the pre-trained vectors of entity description released by the Doc2Vec models [42], i.e., DM and DBOW, as the description embedding.

The word produced and the inclusion of the description negatively affect the treatment of the interrelation problem. Therefore, a text-enhanced KG combination (TEKE) method was introduced [45]. In order to better deal with 1-to-N, N-to-1, and N-to-N problems, representations of entities are augmented with the included textual context. Additionally, TEKE defines the shared contexts of the even entity as the textual context to include each relationship between them [43].

### 2.5.2.2 TEKE [45]

converts a Wikipedia script into a co-presence network. Each entity is treated as a node and the words in the textual context are defined as the neighboring nodes  $n(e)$  from it. The co-occurrence frequency  $y$  is the edge of the connection between them in the context and the threshold is used to remove noise. Include textual context for an entity is defined as follows

Given two nodes  $h, t$ , the textual context of relation is defined as the intersection of  $n(h)$  and  $n(t)$ . Associated embedding is defined as the weighted average as well. The augmented representations of the existing embedding techniques [45].

- **Advantages**

- ✓ The extension on entity representation is available for the existing models.

## 2.5.3 Joint Embedding of The Texts and Facts

Joint embedding aims to project the textual information and structural knowledge into the same continuous vector space for improving the structure-based embedding.

### 2.5.3.1 Jointly [46]

## Chapter 2: Knowledge Graph Completion methods

The KG embedding and word embedding by aligning the facts and the words in raw texts. Knowledge model, text model and alignment model are the components of Jointly. The knowledge model follows TransE to score the facts and designs a conditional likelihood loss LK to learn the general KG embedding. The text model defines the scoring function to measure the plausibility of the two words  $w$  and  $v$  co-occurring in the context [46].

### 2.5.3.2 RLKB [47]

Is a method of jointly embedding the entities, relations and words in entity descriptions in the same vector space. Following Jointly, RLKB designs the LK based on TransE to measure the fitness of facts. Then entity description is made interactive with the entities. Given the set of keywords  $\{w_1, w_2, \dots, w_n\}$  extracted from the description of entity  $e$ , RLKB forces the entity embedding close to the embeddings of the keywords on Euclidean Distance [47].

### 2.5.3.3 JointE+SATT [48]

Introduce mutual attention mechanism between the knowledge model and text model to filter the noise in sentences and obtain more discriminative KG embeddings. Given the set of sentences  $\pi_{rs} = \{s_1, s_2, \dots, s_m\}$  containing the associated entities  $(h, t)$  and textual relation  $r_s$ , a position-based CNN is used to encode each sentence in text model. To represent  $r_s$ , latent relation  $r_{ht} = h-t$  is defined as the attention over the output embedding [48].

- **Advantages**
  - ✓ These methods represent and score the facts with the existing models.
  - ✓ model the textual information and make it interactive with the entities and relations.

## 2.6 Conclusion

In this chapter, we analyze and summarize the current prevailing methods of completing a cognitive graph, based on dividing them into methods of completing the traditional knowledge graph and methods of completing a cognitive graph based on representative learning from an evolutionary perspective. This chapter is concerned only with studying the relevant work of methods based on representational learning due to its potential for use on large-scale cognitive diagrams.

# Chapter 3

---

*Contribution*



### 3.1. Introduction

In this chapter, we provide a general detail about each of the word2vec and its use in NLP, its types, and the difference between them. Next, we detail TRansE technology with a view to arriving at a design proposal for our research approach based on improving TransE working technology using Word2Vec technique in order to enlacing knowledge graph completion using textual content.

### 3.2. Word2Vec

Word embedding is a numerical representation of words, it is the most famous word embedding algorithm, developed by a research team at Google under the direction of Thomas Mikulov in 2013. Its idea is based on two-layer neural networks and seeks to learn vector representations of words that compose a text, so that words that share similar contexts are represented by nearby numerical vectors.

Word2Vec contains two neural architectures:

-  The first is known as Continues Bag Of Word (CBOW)
-  The second is Skip-Gram (SG)

and each of them has a specific feature.

#### 3.2.1. Skip-Gram

In general Skip Gram neural network takes a word as input and tries to predict its context. Sowe try to predict the context words using the main word.

**Example:** From the sentence the black cat is sleeping, let's say we want to get the embedding for the word cat. First, we encode all words in the corpus to train by using one-hot encoding. We pick the word pairs of the word we want to find the embedding of: (cat, the), (cat, black), (cat, is), (cat, sleeping). Now from each of them, we use a Neural Network model with one hidden layer, as represented in the following image:



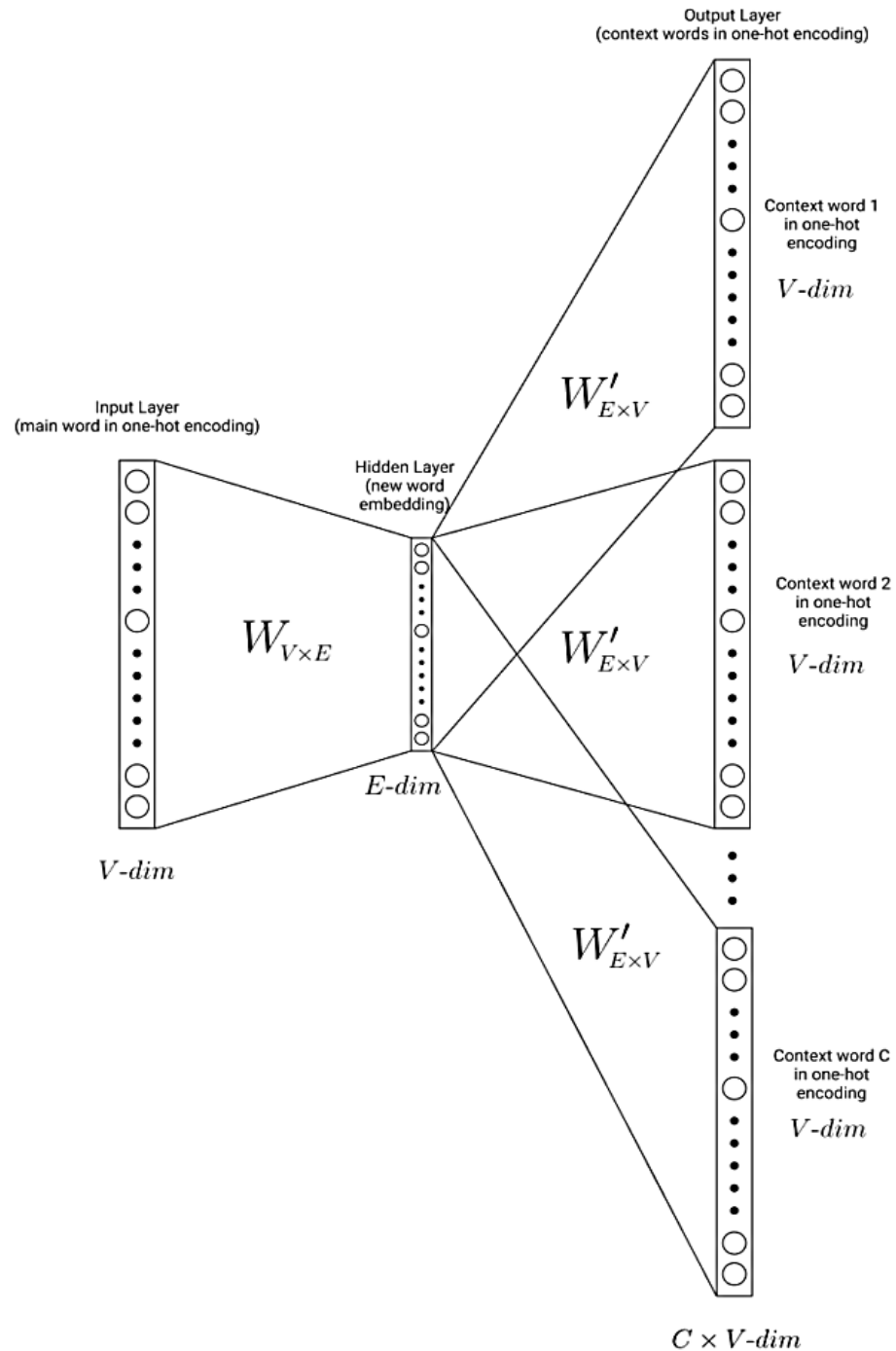


Fig 3. 1: Skip-gram model

As in the figure 2, the input layer with a size of  $1 \times V$ , where  $V$  is the number of words in the corpus vocabulary. The input is the main word in one-hot encoding, cat in our example. The weight matrix  $W$  transforms the input into the hidden layer.

This hidden layer has a size of  $1 \times E$ , where  $E$  is the desired size of the word embeddings. The higher this size is, the more information the embeddings will capture, but the harder it will be to learn it.

Finally, the weight matrix  $W'$  transforms the hidden layer into the output layer. As the outputs to be predicted are going to be the context words in one-hot encoding, the final layer will have a size of  $1 \times V$ . We'll run the model once per context word. The model will learn by trying to predict the context words.

Once the training is done in the whole vocabulary, we'll have a weight matrix  $W$  of size  $V \times E$  that connects the input to the hidden layer. With it, the embeddings can now be obtained. If it has been done correctly, the representation encapsulates semantics, as we mentioned before, and similar words are close to each other in the vectorial world.

### 3.2.2. CBOW

This neural network takes the context of the word, i.e. the surrounding terms in the sentence, as input, and tries to predict the word in question.

In Continuous Bag of Words, the algorithm is really similar to skip gram, but doing the opposite operation. From the context words, we want our model to predict the main word:

As in Skip-Gram, we have the input layer (which now consists of the context words in one-hot encoding – size  $1 \times V$ ). For every context word, we get the hidden layer resulting from the weight  $W$ . Then we average them into a single hidden layer, which is passed on to the output layer. The model learns to predict the main word, tweaking the weight matrices.

Again, once the training is done, we use the weight matrix  $W$  to generate the word embeddings from the one-hot encodings.

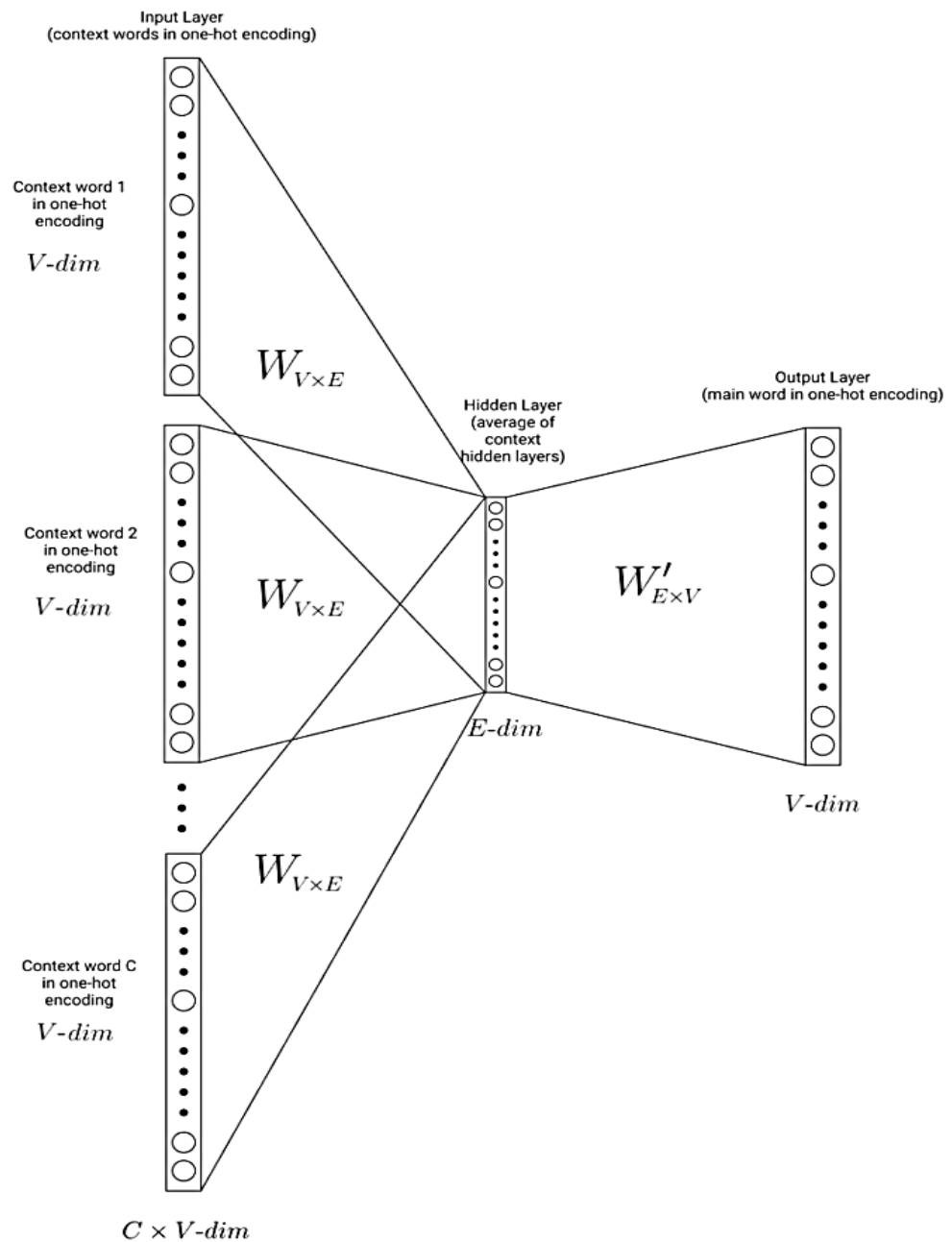


Fig 3. 2:CBOW model

In both cases, the network is trained by reviewing the provided text and adjusting the neuralweights to reduce the algorithm's prediction error.

Word2Vec has various settings, the most important of which are:

- ✓ The dimensions of the vector space to be constructed, i.e., the number of scalar descriptors used to describe words (generally between 100 and 1000).
- ✓ The size of the word context, that is, the number of terms surrounding the word in question (the authors suggest using contexts of size 10 with the Skip-Gram structure and 5 with the CBOW structure).

Since Word2Vec consists of only two layers, this algorithm is quick to train and run, which is a huge advantage over other word embedding methods.

### 3.2.3. CBOW vs Skip-Gram

According to the original paper, Mikolov et al., it is found that Skip-Gram works well with small datasets, and can better represent less frequent words.

However, CBOW is found to train faster than Skip-Gram, and can better represent more frequent words.

Of course, which model we choose largely depends on the problem we're trying to solve.

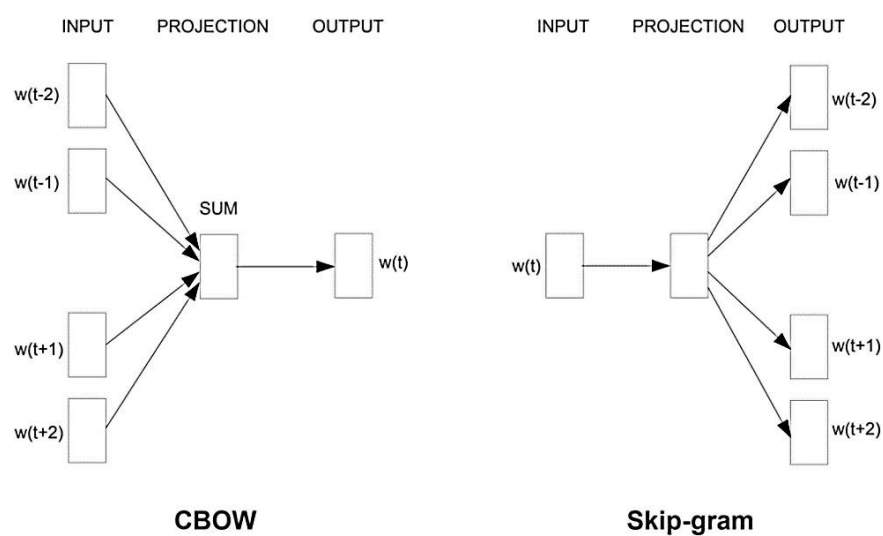


Fig 3. 3:CBOW vs SG

### 3.3. TransE

Inspired by word2vec, TransE is an energy-based model that produces knowledge base embeddings. It models relationships by interpreting them as translations operating on the low-dimensional embeddings of the entities. Relationships are represented as translations in the embedding space: if  $(h,r,t)$  holds, the embedding of the tail entity  $t$  should be close to the embedding of the head entity  $h$  plus some vector that depends on the relationship  $r$ .

### 3.4. Problem study

Completing the knowledge graph is a task that has the main objective of completing the graphs with the missing knowledge. Such as DBpedia, Freebase, Wordnet, and others. Each knowledge graph is a triple set like "Algeria Algiers" where the first is the subject entity, the second is the relationship and the last is the object entity. The main tasks are correlation prediction and relationship categorization where the first predicts the relationship between two particular entities and later categorizes certain triads with true or false. Approaches using only the observed triples can give the best results but fail in the case of invisible entities because the prediction models have been trained using only the existing triples.

### 3.5. Methodology

#### 3.5.1. Proposed solution

In this thesis, we propose a new direction to solve the problem of incompleteness of knowledge graph tasks when using text as an external resource because the text contains rich contents.

#### 3.5.2. Proposed design

Embeddings can be used in many applications such as question answering systems, recommendations systems, sentiment analysis, text classification and it also makes it easier for search. Our design is illustrated by figure 3.1.

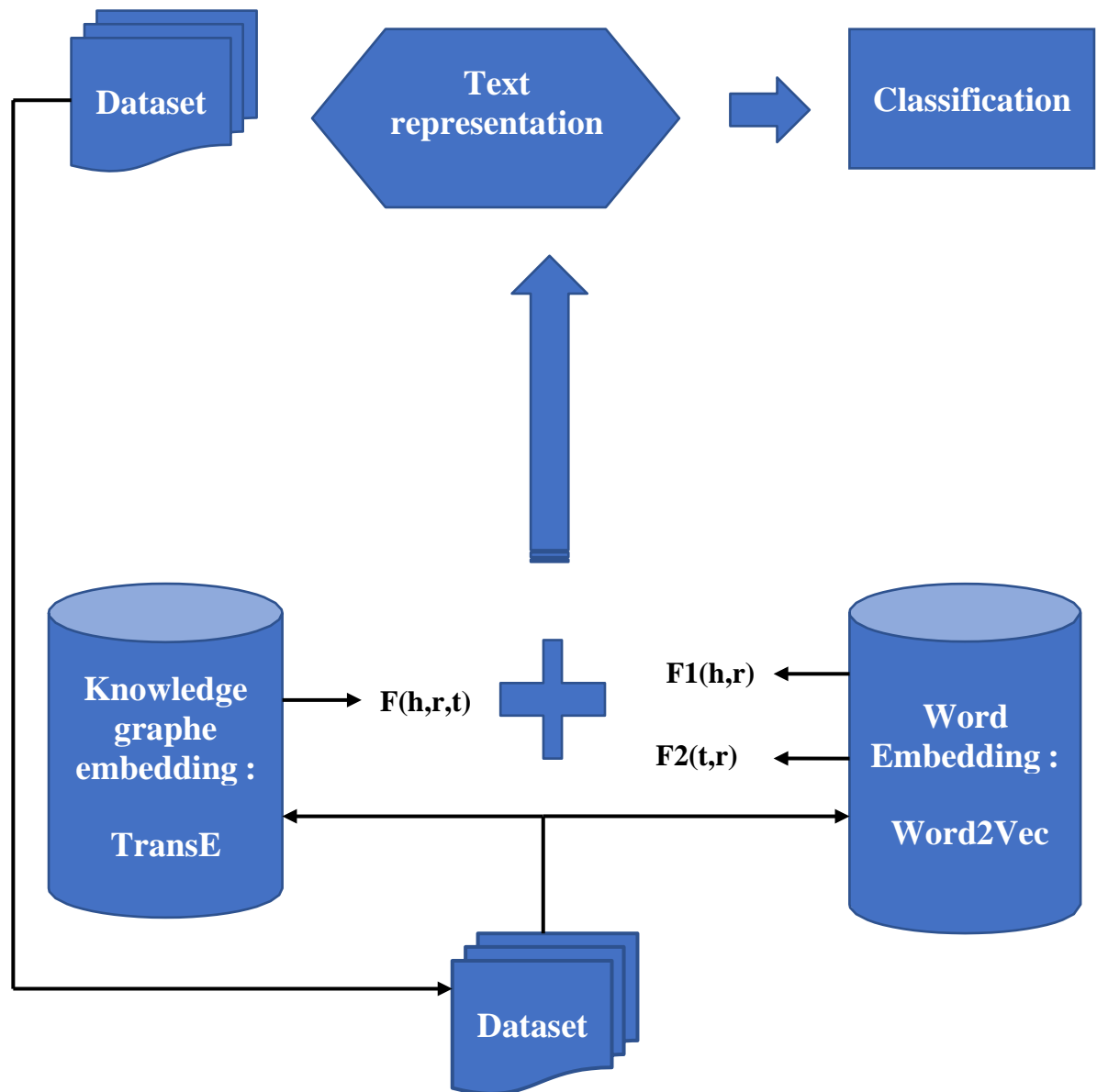


Fig 3. 4: Proposed design

The simple use of a word vector representation without realistic auxiliary information tends to have the effect of a generic model. Single use of a representation of knowledge will make the semantic information of the word itself missing. Therefore, in order to overcome the shortcomings, we propose a knowledge representation learning model based on the integration of word2vec and transE :

1. First, we assume that each entity "h" and "t" has a text description.

2. Second, word2vec reads weddings in every description.
3. Third, TransE predict the score of triplets
4. Finally, the end result is:

**TransE result + word2vec result.**

**Example:** If we have h, r, and t, we will consider two descriptions of h and t.

- In describing h: If Word2vec finds a score between h and t using r, the result will be denoted by f1.
- In describing t: if word2vec finds a degree between h and t using r, the degree is indicated by f2.

**The end result is = f + f1 + f2**

Where f is the result of orgasm.

### 3.6. Conclusion

In this chapter we have exposed our contribution, as well as the proposed architecture. We will present, in the rest of this thesis, our realization on specifying the selected database as well as the implementation of the model.

# Chapter 4

---

## *Realization*



## 4.1. Introduction

In this chapter, we present the stages of building the proposed approach and detail each of the programming language and all the tools involved in building a knowledge graph completion model using DBPEDIA's dataset content, and finally we present and discuss the obtained results.

The steps of realization are organized as next:

1. Using existing knowledge graph embedding models: TransE.
2. Using existed datasets DBpedia.
3. Concentrate on the datasets that are augmented with textual mentions about entities and relations.  
Evaluate the work.

## 4.2. Implementation

### 4.2.1. Dbpedia

DBpedia (from "DB" for "database") is a project aiming to extract structured content from the information created in the Wikipedia project. This structured information is made available on

the World Wide Web. DBpedia allows users to semantically query relationships and properties of Wikipedia resources, including links to other related datasets.

The 2016-04 release of the DBpedia data set describes 6.0 million entities, out of which 5.2 million are classified in a consistent ontology, including 1.5M persons, 810k places, 135k music albums, 106k films, 20k video games, 275k organizations, 301k species and 5k diseases. DBpedia uses the Resource Description Framework (RDF) to represent extracted information and consists of 9.5 billion RDF triples, of which 1.3 billion were extracted from the English edition of Wikipedia and 5.0 billion from other language editions.

From this data set, information spread across multiple pages can be extracted. For

example, book authorship can be put together from pages about the work.

One of the challenges in extracting information from Wikipedia is that the same concepts can be expressed using different parameters in infobox and other templates. Because of this, queries about where people were born would have to search for both of these properties in order to get more complete results. As a result, the DBpedia Mapping Language has been developed to help in mapping these properties to an ontology while reducing the number of synonyms. Due to the large diversity of info boxes and properties in use on Wikipedia, the process of developing and improving these mappings has been opened to public contributions.

Version 2014 was released in September 2014. A main change since previous versions was the way abstract texts were extracted. Specifically, running a local mirror of Wikipedia and retrieving rendered abstracts from it made extracted texts considerably cleaner. Also, a new data set extracted from Wikimedia Commons was introduced.

```
PREFIX dbprop: <http://dbpedia.org/ontology/>
PREFIX db: <http://dbpedia.org/resource/>
SELECT ?who, ?WORK, ?genre WHERE {
  db:Tokyo_Mew_Mew dbprop:author ?who .
  ?WORK dbprop:author ?who .
  OPTIONAL { ?WORK dbprop:genre ?genre } .
}
```

Fig 4. 2: Dbpedia dataset example

Table 4.1 dataset used in the experiments

Dataset	Entities	Relations	Train	Validation	Test
DBPedia 50K	49900	654	32388	399	10969

### 4.2.2. TorchKGE

TorchKGE is a Python module for knowledge graph (KG) embedding relying solely on PyTorch. This package provides researchers and engineers with a clean and efficient API to design and test new models. It features a KG data structure, simple model interfaces and modules for negative sampling and model evaluation. Its main strength is a very fast evaluation module for the link prediction task, a central

application of KG embedding. Various KG embedding models are also already implemented. Special attention has been paid to code efficiency and simplicity, documentation and API consistency. It is distributed using PyPI under BSD license.



Fig 4. 3: TorcheKGE logo

### 4.2.3. Google Collab

Often shortened to "Collab", is a cloud service, offered by Google (free), based on Jupyter Notebook and intended for training and research in machine learning, allows you to write and run Python code in your browser.

This platform allows you to train machine learning models directly in the cloud. It offers the following advantages:

- ✓ No configuration required
- ✓ Free access to GPUs
- ✓ Easy sharing



Fig 4. 4: Google colab logo

### 4.2.4. Python

Is an interpreted, multi-paradigm, cross-platform programming language. It promotes structured, functional and object-oriented imperative programming. It has strong dynamic typing, automatic memory management by garbage collection and an exception handling system.

It is placed under a close free license and works on most computer platforms. It is designed to optimize the productivity of programmers by offering high-level tools and an easy-to-use syntax.



Fig 4. 5:Python logo

### 4.3. Conclusion

In this chapter, we have presented our contribution, we have detailed our proposed design where we have specified the dataset set and the model of word2vec and transE. we also detail all tools used in the implementation of our proposal.

*General conclusion*

---

## **1. Conclusion**

The aim of this study was to try to propose a new approach to improve textual information embedding techniques based on word embedding technique.

We chose this approach to solve the problem of the inability of the unit knowledge graph to predict correlation and classify relationships, and the most important thing is to use an external resource such as text, because text is the largest source of diverse and rich contents.

We selected the DBPEDIA dataset and used the open source TransE code and changed the latter in line with the proposed model in promoting improved text embedding techniques.

We first wrote the first chapter that includes background on the tasks of completing the knowledge graph with a focus on embedding models.

Then we wrote the second chapter, which contains a citation of the main models that use the text to complete the knowledge graph.

This is followed by the third chapter, which is the most important chapter, which includes the proposed model.

Finally, in the fourth and final chapter, we built the model to enhance the performance of embedding with text information.

## **2. Perspectives**

Our work does not stop here, as we seek in the future to:

1. Test the proposed model on another data set.
2. Try other algorithms like TransH and TransG and compare the results.
3. Use the proposed model in a real-world application such as text classification.

# References

---

- [1] : LINDLEY, David. Brains and bytes. Communications of the ACM, 2010, vol. 53, no 9, p.13-15.
- [2] : Sandberg, A. and Bostrom, N. (2008): *Whole Brain Emulation: A Roadmap*, Technical Report #2008-3, Future of Humanity Institute, Oxford University
- [3] : Nair, T. R. G. and M. Malhotra. "Informedledge System - A Modified Knowledge Network with Autonomous Nodes using Multi-lateral Links." KEOD (2010).
- [4] : MALHOTRA, Meenakshi and NAIR, TR Gopalakrishnan. Evolution of knowledge representation and retrieval techniques. International Journal of Intelligent Systems and Applications, 2015, vol. 7, no 7, p. 18.
- [5] : [https://medium.com/@sderymail/challenges-of-knowledge-graph-part-1-d9ffe9e35214\(19/04/2021\)](https://medium.com/@sderymail/challenges-of-knowledge-graph-part-1-d9ffe9e35214(19/04/2021))
- [6] : AUER, Sören, BIZER, Christian and KOBILAROV, Georgi. Dbpedia: A nucleus for a web of open data. In: The semantic web. Springer, Berlin, Heidelberg, 2007. p. 722-735.
- [7] : BOLLACKER, Kurt, EVANS, Colin and PARITOSH, Praveen. Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of data. 2008. p. 1247-1250.
- [8] : VRANDEČIĆ, Denny and KRÖTZSCH, Markus. Wikidata: a free collaborative knowledgebase. Communications of the ACM, 2014, vol. 57, no 10, p. 78-85.
- [9] : SUCHANEK, Fabian M., KASNECI, Gjergji, and WEIKUM, Gerhard. Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web.2007. p. 697-706.
- [10]: BORDES, Antoine, CHOPRA, Sumit, and WESTON, Jason. Question answering with subgraph embeddings. arXiv preprint arXiv:1406.3676, 2014.
- [11]: HE, He, BALAKRISHNAN, Anusha and ERIC, Mihail. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. arXiv preprint arXiv:1704.07130, 2017.

- [12]: ZHANG, Fuzheng, YUAN, Nicholas Jing and LIAN, Defu. Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016. p. 353-362.
- [13]: SOCHER, Richard, CHEN, Danqi, MANNING, Christopher D., et al. Reasoning with neural tensor networks for knowledge base completion. In: Advances in neural information processing systems. 2013. p. 926-934.
- [14]: PAULHEIM, Heiko. Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic web, 2017, vol. 8, no 3, p. 489-508.
- [15] : <https://towardsdatascience.com/embedding-models-for-knowledge-graph-completion-a66d4c01d588> (19/04/2021).
- [16]: BORDES, Antoine, WESTON, Jason, COLLOBERT, Ronan, et al. Learning structured embeddings of knowledge bases. In : Proceedings of the AAAI Conference on Artificial Intelligence. 2011.
- [17]: BORDES, Antoine, USUNIER, Nicolas, GARCIA-DURAN, Alberto, et al. Translating embeddings for modeling multi-relational data. In : Neural Information Processing Systems (NIPS). 2013. p. 1-9.
- [18]: BENGIO, Yoshua, DUCHARME, Réjean, VINCENT, Pascal, et al. A neural probabilistic language model. The journal of machine learning research, 2003, vol. 3, p. 1137-1155.
- [19]: MIKOLOV, Tomas, CHEN, Kai, CORRADO, Greg, et al. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.
- [20]: REITER, Raymond. On closed world data bases. In: Readings in artificial intelligence. Morgan Kaufmann, 1981. p. 119-140.
- [21]: WANG, Shuo, DU, Zhijuan, et MENG, Xiaofeng. Research progress of large-scale knowledge graph completion technology. Scientia Sinica Informationis, 2020, vol. 50, no 4, p.551-575.
- [22]: Lu, Fengyuan, Peijin Cong, and Xinli Huang. "Utilizing Textual Information in Knowledge Graph Embedding: A Survey of Methods and Applications." IEEE Access 8(2020): 92072-92088.
- [23]: An B, Chen B, Han X, Sun L. Accurate text-enhanced knowledge graph representation learning. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) 2018 Jun (pp. 745-755).



- [24]: H.-Y. Zhang, L.-W. Wang, and Y.-X. Chen, “Research progress of probabilistic graphical models: A survey,” *J. Softw.*, vol. 24, no. 11, pp. 2476–2497, Jan. 2014.
- [25]: A. Bordes, X. Glorot, J. Weston, and Y. Bengio, “A semantic matching energy function for learning with multi-relational data,” *Mach. Learn.*, vol. 94, no. 2, pp. 233–259, Feb. 2014.
- [26]: M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, “Asymmetric transitivity preserving graph embedding,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1105–1114.
- [27]: T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” 2013, arXiv:1301.3781. [Online]. Available: <http://arxiv.org/abs/1301.3781>
- [28]: A. Bordes, N. Usunier, A. G. Duran, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proc. 27th Annu. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, vol. 4, Dec. 2013, pp. 2799–2807.
- [29]: Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph embedding by translating on hyperplanes,” in *Proc. AAAI*, Jun. 2014, pp. 1112–1119.
- [30]: G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, “Knowledge graph embedding via dynamic mapping matrix,” in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Co*
- [31]: Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embedding for knowledge graph completion,” in *Proc. 29th AAAI Conf. Artif. Intell.* AAAI, Austin, TX, USA, Jan. 2015, pp. 2181–2187.
- [32]: S. Ma, J. Ding, W. Jia, K. Wang, and M. Guo, “TransT: Type-based multiple embedding representations for knowledge graph completion,” *Mach. Learn. Knowl. Discovery Databases*, vol. 10534, pp. 717–733, Jan. 2017.
- [33] : Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, “RotatE: Knowledge graph embedding by relational rotation in complex space,” 2019, arXiv:1902.10197. [Online]. Available: <http://arxiv.org/abs/1902.10197>
- [34]: G. Qi, H. Gao, and T. Wu, “The reaserch advances of knowledge graph,” *Technol. Intell. Eng.*, vol. 3, no. 1, pp. 4–25, 2017
- [35]: B. Yang, W. Yih, X. He, J. Gao, and L. Deng, “Embedding entities and relations for learning and inference in knowledge bases,” in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 2–13.

[36]: M. Fan, Q. Zhou, E. Chang, and F. Zheng, “Transition-based knowledge graph embedding with relational mapping properties,” in Proc. 28th Pacific Asia Conf. Lang., Inf., Comput., Phuket, Thailand, pp. 328–337, Dec. 2014.

[37]: Wang, Y.; Liu, Y.; Zhang, H.; Xie, H. Leveraging Lexical Semantic Information for Learning Concept-Based Multiple Embedding Representations for Knowledge Graph Completion. In Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data; Springer: Berlin/Heidelberg, Germany, 2019; pp. 382–397.

[38]: W. Y. Wang, K. Mazaitis, and W. W. Cohen, “Programming with personalized pagerank: A locally groundable first-order probabilistic logic,” in Proc. 22nd ACM Int. Conf. Conf. Inf. Knowl. Manage. CIKM, 2013, pp. 2129–2138.

[39]: T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, “Convolutional 2D knowledge graph embeddings,” in Proc. 32st AAAI Conf. Artif. Intell., New Orleans, Louisiana, USA, Feb. 2018, pp. 1–9.

[40]: B. Shi and T. Weninger, “ProjE: Embedding projection for knowledge graph completion,” in Proc. 31st AAAI Conf. Artif. Intell., San Francisco, CA, USA, Feb. 2017, pp. 112–122.

[41]: T. Long, R. Lowe, J. C. K. Cheung, and D. Precup, “Leveraging lexical resources for learning entity embeddings in multi-relational data,” in Proc. 54th Annu. Meeting Assoc. Comput. Linguistics, 2016, pp. 112–117

[42]: N. Veira, B. Keng, K. Padmanabhan, and A. Veneris, “Unsupervised embedding enhancements of knowledge graphs using textual associations,” in Proc. 28th Int. Joint Conf. Artif. Intell., Aug. 2019, pp. 5218–5225

[43]: X. Sun, Y. Man, Y. Zhao, J. He, and N. Liu, “Incorporating description embeddings into medical knowledge graphs representation learning,” in Proc. Int. Conf. Hum. Centered Comput., 2018, pp. 188–194.

[44]: Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in Proc. Int. Conf. Mach. Learn., 2014, pp. 1188–1196.

[45]: Z. Wang, J. Li, Z. Liu, and J. Tang “Text-enhanced representation learning for knowledge graph,” in Proc. 25th Int. Joint Conf. Artif. Intell., 2016, pp. 1293–1299.

[46]: Z. Wang, J. Zhang, J. Feng, and Z. Chen, “Knowledge graph and text jointly embedding,” in Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), 2014, pp. 1591–1601.

[47]: M. Fan, Q. Zhou, T. F. Zheng, and R. Grishman, “Distributed representation learning for knowledge graphs with entity descriptions,” *Pattern Recognit. Lett.*, vol. 93, pp. 31–37, Jul. 2017.

[48]: X. Han, Z. Liu, and M. Sun, “Neural knowledge acquisition via mutual attention between knowledge graph and text,” in Proc. 32th AAAI Conf. Artif. Intell., 2018, pp. 4832–4839.