POLITECNICO
MILANO 1863

Politecnico di Milano
Facoltà di Ingegneria Industriale e dell'Informazione
MSc COMPUTER SCIENCE AND ENGINEERING

Knowledge Graph Embedding for Drug Repurposing

Master of Science thesis of:
Edoardo Ramalli
Matricola 922628

Advisor:
Prof. Marco D. Santambrogio

Co-advisor:
Doc. Ing. Alberto Parravicini

Academic Year 2019/2020

Acknowledgments

Io sto ancora imparando.

Preferisco poco parole, dirette, ma di maggior peso per ringraziare le persone che mi sono state accanto in questo percorso di studi e per la conclusione di questo lavoro.

Il primo ringraziamento va al mio relatore, il prof. Santambrogio che mi ha permesso di raggiungere questo risultato e che grazie al NecstLab ha permesso a me e a molti altri studenti, di mettersi in gioco, sviluppando soluzioni a problemi reali, creando un'ambiente giovane, dinamico ma sempre connesso con persone esperte ed aziende.

Un grande ringraziamento va ad Alberto, il mio advisor, tanto preparato quanto disponibile che mi ha dedicato piú attenzioni del dovuto stimolandomi a migliorare in ogni aspetto con i suoi preziosi consigli come un fratello maggiore farebbe. Un grazie anche a Guido e Mirko che mi hanno aiutato a sviluppare, con il loro supporto ed i loro consigli, questa tesi.

Ringrazio infine la professoressa Céline Hudelot che assieme al MISC Lab della CentraleSupélec mi ha fornito assistenza e la strumentazione necessaria per portare a termine questa ricerca.

Ringrazio e dedico questa tesi alla mia famiglia, ai miei amici e colleghi che mi hanno accompagnato durante tutto questo percorso.

Abstract

Nowadays is essential to able to respond to a new spreading disease in a brief time. For this reason, a conventional approach is not responsive enough.

Drug Repurposing is the investigation of existing drugs on the pharmaceutical market for new therapeutic purposes; drug repurposing reduces the time and cost of clinical trial steps, saving years, and billions of dollars in Research and Development (R&D) and can be also used to find a treatment for a rare disease that could have not a market for the high cost of R&D. In such a way drug repurposing represents a solution to a disease in a shorter time.

Identifying new diseases on which a drug can be effective is a complex problem: our approach leverages Knowledge Graph (KG), networks composed of many types of entities and relations, on which embedding and graph completion techniques can be applied to infer insights and analyses. Our KG is built from well-known databases such as DrugBank, UniProt, and CTD and contains over one million relationships between more than 70K biological and pharmaceutical entities like diseases, genes, proteins, and drugs. In this work, we research the applicability of knowledge graph completion techniques, such as link prediction (and triple classification) using a various number of different embedding models from different families: matrix factorization, geometric and Deep learning. Using these models is possible to infer new drug-disease relationships on our KG, and identify novel drug repurposing candidates.

Results are encouraging and show how state-of-the-art machine learning models, combined with the ever-growing amount of biological data freely available to the research community, could significantly improve the field of drug repurposing. In our evaluation is used H@10, that is the proportion of true triples respect to the top 10 predictions of the embedding model, as a measurement of accuracy. A score of H@10 around 0.5 means that the model is learning information from the dataset and it has significantly improved a random baseline based on guessing that achieves less than 0.002 on the same measure. In addition to this result, using more complex embedding models and more data improves the quality of results. Achieving a score of 0.5 is a good result due to the complexity of the problem.

It should be underlined that these techniques can produce these result, from the generation of the knowledge graph to the prediction, in just a few hours, and, for this reason, is capable of reducing the dimensional complexity of the problem in a very short time since the deep inspect of potential candidates for a disease can start from a small subset.

Finally, can be observed that starting from the generation of the network from biological expertise, and, using machine learning techniques, the model generates a piece of new biological knowledge, not only in the specific case of drug-disease links but provides a better understanding of the interactions and their importance between all the entities and relationships that are present in our knowledge graph.

This work is organized as follows:

- Chapter 1 provides an explanation of what is drug repurposing and knowledge graph embedding, showing an overview of current techniques on drug discovery. It also presents an overview of the main challenges and the solution proposed.
- Chapter 2 provides the necessary theoretical background on knowledge graph, knowledge graph embedding and completion and a presentation of the dataset that is used to perform drug repurposing.
- Chapter 3 offers an overview on existing state-of-the-art computational drug repurposing techniques and embedding models, and on the methodologies that have been proposed over the years.
- Chapter 4 discusses how has been implemented, which design choices have been made, and how it can be configured for different study cases.
- Chapter 5 reports results obtained running on different scenario, data-sets with a thorough comparison to other approaches available in the state-of-the-art.
- Chapter 6 discusses the results, and future works.

Sommario

Oggigiorno è fondamentale poter saper rispondere in breve tempo a una nuova malattia che si diffonde. Per questo motivo, un approccio convenzionale non è sufficientemente reattivo.

Drug Repurposing è lo studio di farmaci esistenti sul mercato farmaceutico per nuovi scopi terapeutici; il riutilizzo dei farmaci riduce i tempi e i costi delle fasi della sperimentazione clinica, risparmiando anni e miliardi di dollari in Research and Development (R&D) e può essere utilizzato anche per trovare una cura per una malattia rara che potrebbe non avere un mercato per l'alto costo di R&D. In tal modo la riconversione di farmaci rappresenta una soluzione a una malattia in breve tempo.

Identificare nuove malattie su cui un farmaco può essere efficace è un problema complesso: il nostro approccio fa leva su Knowledge Graph (KG), è un network composto da molti tipi di entità e relazioni, su cui è possibile applicare tecniche di embedding e completamento di grafi per dedurre nuovi collegamenti e per analisi. Il nostro KG è costruito partendo da informazioni contenute in database ben noti come DrugBank, UniProt e CTD e contiene oltre un milione di relazioni tra più di 70,000 entità biologiche e farmaceutiche come malattie, geni, proteine e farmaci. In questo lavoro, ricerchiamo l'applicabilità delle tecniche di completamento del knowledge graph, come la previsione di un nuovo collegamento, link prediction (e la triple classification), utilizzando diversi modelli di embedding provenienti da famiglie differenti: fattorizzazione della matrice, geometrica e deep learning. Utilizzando questi modelli è possibile dedurre nuove relazioni farmaco-malattia sul nostro KG ed identificare nuovi candidati per il drug repurposing.

I risultati sono incoraggianti e mostrano come i modelli di embedding siano all'avanguardia, che, combinati con la quantità sempre crescente di dati biomedici liberamente disponibili per la comunità di ricerca, potrebbero migliorare in modo significativo il campo del drug repurposing. Per poter valutare la precisione di queste predizioni, viene utilizzato H@10, ovvero la proporzione di triple vere rispetto alle prime 10 previsioni del modello di embedding. Un punteggio di H@10 intorno a 0,5 significa che il modello sta apprendendo informazioni dal set di dati e ha migliorato significativamente una baseline basata sulla predizione casuale di

un collegamento che raggiunge meno di 0,002 sulla stessa misura. Oltre a questo risultato, l'utilizzo di modelli di embedding più complessi ed il maggior nummero di dati migliora la qualità dei risultati. Raggiungendo un punteggio di 0,5 è un buon risultato considerando la complessità del problema.

Va sottolineato che questo approccio è in grado di produrre questi risultati, dalla generazione del knowledge graph alla predizione, in poche ore, e, per questo motivo, è in grado di ridurre la complessità dimensionale del problema in tempi brevissimi, riducendo il numero dei potenziali candidati per una malattia da cui può successivamente iniziare uno studio più approfondito.

Infine, si può osservare che, a partire dalla generazione della rete costituita di conoscenze biologiche e, utilizzando tecniche di machine learning, il modello genera una nuova conoscenza biologica, non solo nel caso specifico dei legami farmaco-malattia ma fornisce una migliore comprensione delle interazioni e della loro importanza tra tutte le entità e le relazioni che sono presenti nel nostro knowledge graph.

Questo lavoro è organizzato come segue:

- Chapter 1 fornisce una spiegazione di ciò che sono il drug repurposing ed il knowledge graph embedding, mostrando una panoramica delle attuali tecniche sullo sviluppo di nuovi farmaci. Presenta inoltre una panoramica delle principali sfide e della soluzione proposta.
- Chapter 2 fornisce il background teorico necessario per il knowledge graph, l'embedding e il completamento dello stesso e una presentazione del set di dati che vengono utilizzati per eseguire il repurposing dei farmaci.
- Chapter 3 offre una panoramica sulle tecniche computazionali per il drug repurposing, sui modelli di embedding, e sulle metodologie che sono state proposte nel corso degli anni.
- Chapter 4 discute come è stato implementato, quali scelte progettuali sono state fatte e come può essere configurato per diversi casi di studio.
- Chapter 5 riporta i risultati ottenuti nei diversi scenari e set di dati con un confronto approfondito con altri approcci disponibili presentati nello stato dell'arte.
- Chapter 6 discute i risultati e i lavori futuri.

Contents

List of Tables

List of Figures

Introduction and motivations 1

La bellezza non é una qualitá delle cose stesse: essa esiste soltanto nella mente che le contempla ed ogni mente percepisce una diversa bellezza.

This chapter explains the context of this research and presents the objective of our work. Starting from the problem description, the ordinary computational and pharmaceutical approaches for drug repurposing are presented. Afterwards, a powerful machine learning tool is introduced to be applied to a custom biomedical knowledge graph. In particular, section 1.1 outlines the problem in the current scenario of drug discovery, where the current paradigm is actually expensive and slow with respect to the need of the population and the responsiveness to a new spreading disease. Then, section 1.2 focuses on the current techniques of drug repurposing that try to overcome the issues of the high cost and slow development of a new drug using different approaches. Following on, section 1.3 suggests a way to represent the problem domain in a way that is more understandable to a computer so that it can infer new knowledge. This is possible using embedding techniques. Finally, section 1.4 outlines the proposed solution, its main guidelines, and its evaluation.

1.1 Drug Discovery

New drugs are continuously required by the healthcare systems to address unmet medical needs across diverse therapeutic areas, and pharmaceutical industries primarily strive to deliver new drugs to the market through the complex activities of drug discovery and development [42].

Drug Development is a high cost and very time-consuming process. To better understand this procedure Food and Drug Administration (FDA) has proposed a timeline regarding the step that is necessary to deliver a new drug.

- The first stage is the discovery and development of a new drug. Typically, researchers discover new drugs through new insights into a disease process that allow the design of a product to stop or reverse the effects of the disease. Otherwise, they can try to identify some tests of molecular compounds to find beneficial effects against a large number of diseases or using new technologies to manipulate genetic material or target specific sites within the body. The number of possible approaches to discover a potential candidate is massive and in the past and in recent years a lot of new one are born [10]. At this stage of the process, thousands of compounds may be potential candidates for the next stage of development to be medical treatment. After early testing, however, only a small number of compounds look promising and call for further study. Once researchers identify a promising compound for development, they conduct experiments to gather information on the dosage, the way in which the body can metabolize it, its effectiveness etc.. As expected this stage with all its exploration phase is very time consuming and in average it takes around 6.5 years [52].

- The next stage is the preclinical research: it aims to identify possible toxicity of the compounds. The number of tests is not so large but should be sufficient to provide enough information to the researcher whether they want to proceed to human testing. Usually, this stage takes some months [52].

- Then there is the clinical stage where the drug is tested on humans to know the effectiveness and possible side effects. Usually, this stage is split into sub-stage where a number of increasing volunteers participate in the test and correspondingly a number of decreasing drug goes to the next stage. The amount of time that is necessary to do all these clinical steps is the order of 5 years [52].

- Finally, all the collections of the clinical trials from the pharmaceutic company are submitted to FDA that decide if the study is sufficient and the drug is safe for commercialization and so can decide to approve it or not. FDA, in case of an approval, will continue to monitor the new drug to have a complete picture, and in some case can, for example, change the indication of the drug or the dosage.

In the current situation, all processes are time-consuming and expensive and the industry is under pressure owing to the extremely stringent regulatory requirements, environmental concerns, and reduced incomes due to patent expiration. These issues have had an adverse bearing on the R&D productivity in recent years, hence there is a need for innovative approaches as well as increased collaboration between industry, academia, and governmental research institutions, with a common objective of constantly delivering quality medicines [42].

1.2 Drug Repurposing

This section introduces Drug Repurposing, a technique central to efficiently develop new drugs. After a brief discussion about the benefits of drug repurposing in Section 1.2.1 is introduced the classic approach for drug repurposing. Then, section 1.2.2 introduces the main computational techniques that are expanded in chapter 3.

As said before, drug discovery is research, and being such, it has no guarantee of success. For this reason, it is a high risk, slow, and expensive process [52]. The hazard is due to the fact that the pharmaceutical sector is highly competitive and finding a drug for a disease is, therefore, a race against time for both health and economic needs [47].

Eastern Research Group (ERG) published a report that shows only 2% of new molecular succeed and the order of time needs to develop is 10-15 years [52]. Moreover, the yearly amount of investments reported by PhRMA in United States of America (USA) w.r.t the number of drugs approved by the FDA, shows that, while investments are increasing year

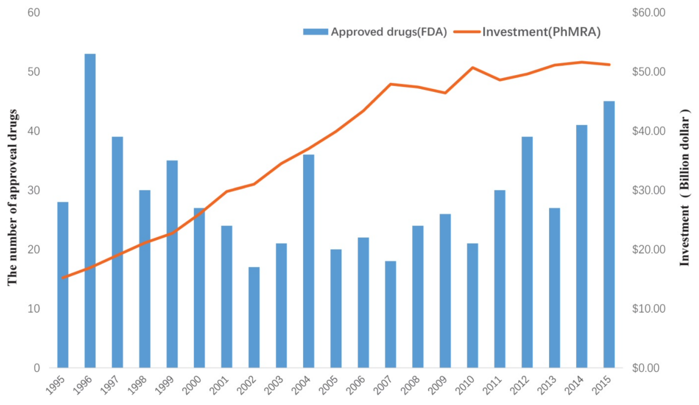


Figure 1.1: The amount of investment in drug development by Pharmaceutical Research and Manufacturers of America (PhRMA) member companies and the number of approved drugs by the FDA from 1995 to 2015. Figure from Xue et al. [52].

after year, the fewer and fewer drugs are being approved, as shown in fig. 1.1 [52].

Clearly, the global trend over the year is a constant increase regarding the number of investments and the decreasing number of approved drugs. That is translated into an increased cost of development of each new drug.

So, finding a treatment for a rare disease, that doesn't have a big share market, it is a too high-risk process since the number of the possible customers is limited and the volume of medical information on which the researcher can rely is narrow.

Drug repurposing is a valid solution: in section 1.1 are defined the five stages that are necessary to develop a new drug, instead of in drug repurposing there are only four that are faster fig. 1.2. Doing so, it is possible to save up to half of the time and billions of dollars in R&D reducing at the same time the risk to fail since the drug is already approved and declaring a new indication for a drug is easier [52].

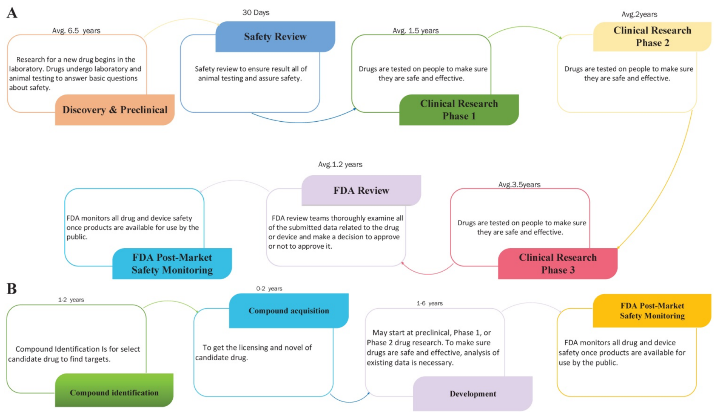

A
Avg. 6.5 years
Research for a new drug begins in the laboratory. Drugs undergo laboratory and animal testing to answer basic questions about safety.
Discovery & Preclinical
30 Days
Safety Review
Safety review to ensure result all of animal testing and assure safety.
Avg. 1.5 years
Drugs are tested on people to make sure they are safe and effective.
Clinical Research Phase 1
Avg.2years
Clinical Research Phase 2
Drugs are tested on people to make sure they are safe and effective.
Avg.3.5years
Drugs are tested on people to make sure they are safe and effective.
Clinical Research Phase 3
Avg.1.2 years
FDA Review
FDA review teams thoroughly examine all of the submitted data related to the drug or device and make a decision to approve or not to approve it.
FDA monitors all drug and device safety once products are available for use by the public.
FDA Post-Market Safety Monitoring
B
1-2 years
Compound Identification Is for select candidate drug to find targets.
Compound identification
0-2 years
Compound acquisition
To get the licensing and novel of candidate drug.
1-6 years
May start at preclinical, Phase 1, or Phase 2 drug research. To make sure drugs are safe and effective, analysis of existing data is necessary.
Development
FDA Post-Market Safety Monitoring
FDA monitors all drug and device safety once products are available for use by the public.

Figure 1.2: The contrast of traditional drug development and drug repositioning. A) Flowchart of the traditional drug development process. B) Flowchart of drug repositioning. Figure from Xue et al. [52]

1.2.1 Experimental Approach

In the past the idea of drug repurposing or drug repositioning has been largely opportunistic and serendipitous; a drug is a good candidate for a repurposed drug and be used for a different commercial exploitation when is discovered that it has some off-target effect that can be used or a new one. In any case, the most successful examples of drug repurposing so far have not involved a systematic approach but the opposite. Two examples are widely known: the repurposing of sildenafil citrate for erectile dysfunction relied on retrospective clinical experience, and repurposing of thalidomide for erythema nodosum leprosum and multiple myeloma was based on serendipity. Following these successes have brought encouragement in defining a more systematic approach to identify possible repurposable compounds leaving a marginal space to a serendipitous discovery. These approaches have resulted in the identification of a number of promising candidate drugs, some of which are in advanced stages of clinical trials, with the potential for use in the treatment of both common and rare diseases, giving a chance to develop drugs that otherwise are out of the market [39].

There are mainly two experimental approaches for the investigation of a possible candidate for a drug repurposing. Proteomic techniques such as affinity chromatography and mass spectrometry have been used as approaches to identify binding partners, so in this case, these methods are looking for a structural matching. In an era of chemical biology for target validation, analyses of the targets and off-targets of drugs and drug repurposing have become a normal procedure because allows you to collect more information also for future research [39]. Phenotypic screening can identify compounds that show disease-relevant effects in model systems without prior knowledge of the target. In the context of drug repurposing, if the compounds screened are approved or investigational drugs, this may indicate repurposing opportunities that can readily be pursued [39].

1.2.2 Computational Approach

A computational approach to the problem is cheaper and faster, and is some case also extendable to another target without spending so much effort. A possible categorization of the computational approaches can be done over the biological network that is used. This section follows the drug repurposing methods based on the types of approach proposed by Xue et al. [52].

1.2.2.1 Network-based cluster approaches

Network-based method are currently widely used due to their ability to integrate in a simple way multiple sources of data. In the specific case, the idea of using cluster is based on the observation that similar diseases have similar drugs because they share comparable biological characteristics. For these reason, creating cluster in the network based on its topology can be used to infer novel relation inside the network. To do that it is possible to compute a comprehensive similarity between drugs and diseases or extract drug-disease sub network or using a bi-random walk algorithm [52].

1.2.2.2 Network-based propagation approaches

In this method, the idea is to propagate prior information from the source node to all network nodes. There are several different ways to propagate the information but mainly there are two types. Local propagation methods only take into consideration a small amount of information on the network. On the other hand, global approaches try to propagate all the information to all the nodes that belong to the network or to its subnetwork. A global approach is more precise than the local one but is more computationally expensive. In any case, these kinds of approaches are easily extendable and have good results [52].

1.2.2.3 Text mining-based approaches

One of the main problems with the conventional approach to drug repurposing is the big amount of information that a scientist should be able to process. Due to the fact that the amount of literature produced in the medical and pharmaceutical field is remarkable since, for example, the study of the same phenomenon but on a different group of people, could bring new results. For this reason, it is often difficult to extricate oneself and to be able to process all previous knowledge and contribute effectively without having neglected the evidence already identified. Text mining is a solution to this problem that can be applied in the specific case of drug repurposing. Analyzing scientific texts, the idea is to extrapolate possible drug-disease connections that did not previously exist simply by connecting information from different sources. If the idea is clear and simple, its automated implementation is much more complex. The process mainly involves 4 steps: the Information Retrieval (IR) is in charge of the collection of relevant documents from the literature. Then these documents are filtered to extract useful concepts. Biological Name Entity Recognition (BNER) extract the entities in the text and then Biological Information Extraction (BIE) and Biological knowledge discovery (BKD) try to derive the concepts between entities. This process is necessary since it is difficult to automate the extraction of concepts from a human language. The results are good and with the advancement of Natural Language Processing (NLP) techniques, they are increasingly complete and precise [52].

1.2.2.4 Semantics-based approaches

The semantic approach is used in many areas, from image recognition to retrieval information. The method generally involves three steps: the first is the extraction of large quantities of information from databases, after which a semantic network is built where in the last phase it will be possible to apply techniques for extracting new information [52].

1.3 Knowledge Graph Completion

A Knowledge Graph (KG) is a way of representing a domain in a graph. This mode of representation is more schematic and is based on connecting entities by means of relationships. These simple concepts are widely used in computer science and for this reason, they are easy to interpret, manipulate and extend. Following this idea can easily build a graph made up of tens of thousands of entities for a total of millions of connections. Given its simplicity and applicability, they are also widely used in the industrial field as in the case of Microsoft and Google.

Once a KG is ready, it is feasible to apply on it a set of techniques called Knowledge Acquisition (KA). These methodologies are, for example, Knowledge Graph Completion (KGC) that wants to complete an existing knowledge graph, or regarding the extraction of relation and entities. Therefore, KG can be used as a sort of database to be interrogated to understand with which other entities a subject relates, or to complete the graph, in terms of entities or relation using inference techniques [21].

To be able to complete the KG, one of the most used techniques is that of Knowledge Graph Embedding (KGE). KGE is a vector representation of the entities and relations present within the graph but in a dimensional space of reduced size. There are several families of models that generally differ in their specialization in better representing a feature, a property of the graphic designer, however, often increasing the complexity of the representation. The representation of the graph corresponds to a matrix, called the embedding matrix. Exploiting this representation of the original KG, it is possible to deduce new connections inside. With the Link Prediction (LP) activity, in particular, a triple element of KG is damaged and is replaced with all or with a specific subset of elements. For each of these, the score is calculated and a ranking is formed, based on the highest score obtained. To assess the quality of embedding is measured the number of times a truly existing triple in the dataset is present in a number of main predictions. Another similar task that is often associated with LP is Triple Classification (TC). TC is to determine whether facts are correct in testing data, which is typically regarded as a binary classification problem. The decision rule is based on the scoring function with a specific threshold [21].

1.4 Proposed solution

Our solution aims to combine biological expertise with machine learning. Using information regarding genes, proteins, drugs, and diseases, it is possible to create a KG having approximately $180k$ triples between $70k$ different entities using 70 different relationships. Then, applying KGE algorithms, is generated an embedded representation of the graph in order to use inference techniques to identify possible new relationships in the graph that was not previously there. To measure the results obtained, following the previous literature, is used, as the measure of accuracy, the one proposed by Stanford [17] on an unknown part of the dataset. After that, is carried out a series of technical and semantic analyses to check which are the best parameters for the embedding model and applying downsampling or supersampling techniques to deduce biological conclusions such as the importance of gene's presence in the KG to infer new links between drug and disease. In this way, in addition to generating new links in the graph, it may be to better understand the importance of entities and relationships within the graph in relation to biomedical or topological motivations. So starting from a biomedical knowledge of the problem provided by the database, through the use of machine learning algorithms, it is probable to generate new evidence.

1.5 Challenges

The main challenges that have been faced during the writing of this thesis are of two types: the first of a semantic character, the second of a technical nature.

The semantic problems encountered during the course of this project concern the availability of data, the completeness of the graph, and its quality. In order to create a KG that was large enough to well represent the reality and able to learn from previous information, it was necessary to integrate information from different databases that are public or with an academic license. In our case, DrugBank[1], CTD[2], UniProt[3], and OMIM[4] have been used. The data-sets that are used in this thesis are frequently used even by international authorities and major pharmaceutical company since they are, often, manually cured by difference no-profit organizations that are involved in biomedical research. For this reason, is feasible to think that this KG is in line with the quality standards of the other graphs used in computational biology. In any case, the approach to the problem is completely generic, and this means that it is possible to replace, extend, or change the starting KG very simply.

As regards the technical problems, there were mainly two: the first one is a computational challenge and the second one is qualitative concerning the KGE. Working with an embedded representation of a KG of thousands of entities is very expensive, and consequently is necessary to develop some procedure to reduce the cost and deliver a solution in very short time respect to the previous procedure. Reducing the amount of time to generate an embedded solution on a big KG make it possible to do a larger number of test on the KGE representation to better understand the pros and cons of this technique. The second problem, concerning the KGE, was to face and verify what the previous literature proposed. Nowadays there are many embedding models available, each of which aims to significantly improve some specific features. However, rightly, these measures were taken using KG, complete, large, and precise, which often, however, do not represent the reality in which it is not always possible to verify the quality of a graph created from heterogeneous data. In any case, it should be emphasized, that the LP task applied to drugs and disease is a more complex task than the LP of other types of relationships for the simple fact that the number of interactions available is much lower than, for example, protein-protein interactions as presented in chapter 5.

Drug repurposing has already proved a notable success but this

[1]`https://www.drugbank.ca`
[2]`http://ctdbase.org`
[3]`https://www.uniprot.org`
[4]`https://omim.org`

method has itself some challenge: repurposing does not always succeed; most of the selected drug candidates for repurposing failed to encounter a problem in the later stage of the process. Therefore, although a computational approach is a low cost, the results of these cannot necessarily guarantee good results from a pharmaceutical point of view as in the usual process of drug discovery. However, repurposing a new drug does not have only challenges from a research point of view, but also from a bureaucratic one such as patent considerations, organizational obstacles and regulatory concern [39].

Background 2

La conoscenza é un'arma.
Munisciti bene prima di
andare in battaglia.

This chapter introduces fundamental concepts of the knowledge graph, knowledge graph embedding, and the notation that will be used in this work to denote it and the tools to apply link prediction. The definitions follow the one used by Rossi et al. [21].

It is also provided with an explanation of the operation of the embedding model which is used to generate the representation of Knowledge Graph (KG) and to be able to use the Link Prediction (LP).

2.1 Knowledge Graph

Definition 1. Knowledge Graph

Knowledge Graph (KG) is defined as $\mathcal{G} = \{E, R, F\}$, where E, R and F are sets of entities, relations and facts, respectively. A fact is denoted as a triple $(h, r, t) \in F$ where h stands for 'Head', r stands for 'Relation' and t for 'Tail'. Another notation that is often used in literature is $< head, relation, tail >$.

So a KG can be denoted as a multi-relational graph composed of entities and relations which are regarded as nodes and different types of edges, respectively. Over time the concept of KG has evolved according to the research directions and the applications and today there are mainly four different paths of research [21]:

- Temporal Knowledge Graphs, are KG which in addition to trying to represent the information also trying to trace their evolution on a temporal basis.

- Knowledge-aware Applications, on the other hand, are KG in which knowledge is injected to provide a series of downstream applications that can improve tasks such as recommend systems or Question and Answer (Q&A).

- Knowledge Representation Learning (KRL) deals with how to go to represent a KG in another space.

- Knowledge Acquisition (KA) concerns the extraction of information from the starting KG or its subsequent representation. In particular, KA deals with the completion of KG, extraction or identification of entities and relationships.

2.2 Knowledge Graph Embedding

Knowledge Representation Learning (KRL) is also known as Knowledge Graph Embedding (KGE) and the main differences between them are due to the representation space, the scoring function, encoding models and any other information that can be incorporated into the embedding.

The central point of embedding is to represent something in the space of a smaller dimension. The types of dimensional spaces that are commonly used are: pointwise, complex, Gaussian distribution and manifold.

Point-wise Euclidean space is widely used and bases its idea projecting relations and entities into a vector or matrix space. The main difference between the various types of embedding models is how space is built and consequently, what is the projection method of the triple Knowledge Graph (KG) elements. For example, at the expense of computational costs, one can decide to project relations and entities in different dimensional spaces, in order to better enhance the characteristics of the relations, as in the case of TransR. Or with TransA instead, one choose to approximate the entity closest to a point in space with an ellipse instead of a circle, just to solve the ambiguity introduced by it. With TransM, on the other hand, the embedding of the elements of the graph is calculated, taking care to maintain the one to many and many to many relationships also in the representation of the embedding by using a specific parameter. Ultimately the ways in which the embedding matrices are repeatedly improved do not differ much as explained in section 2.4.1. Each model introduces only additional parameters that deal with improving the representation of a given graph property.

Complex vector space instead, as the name suggests, use a complex space to represent entities and relationships. For this reason, each element of the graph has a complex and real representation which improves embedding in the case of symmetric and anti-symmetric relation, for example.

With Gaussian word embedding, the goal is to seek a multi-dimensional Gaussian distribution in such a way that the mean value of the distribution represents the entity or the relationship and the variance represents the uncertainty of this representation.

Finally, the approach of using manifolds and groups is very similar to Point-wise Euclidean space. A manifold is a topological space which could be defined as a set of points with neighbourhoods by the set theory, while the group is algebraic structures defined in abstract algebra. So in Point-wise Euclidean space, the embeddings are restricted in an overstrict geometric form even in some methods with subspace projection instead now they must not be in a well-defined geometric space but rather in a mathematical space. Manifold space has an advantage over point-wise Euclidean space by relaxing the point-wise embedding [21].

The second element that characterizes an embedding model is its score function. The score function is used to measure the plausibility of a triple and must be designed in such a way that a positive triple has a higher score than a negative triple.

There are two typical types of scoring functions to measure the likelihood of a triple: distance-based and similarity-based functions. Distance-based scoring function measures the plausibility of facts by calculating the distance between entities that depends on the space of representation, where addictive translation with relations as eq. (2.1):

$$h + r = t \tag{2.1}$$

eq. (2.1) is widely used for example. So the goal is to find an embedded representation of the head h and the relation r such that their vector sum equals to the embedding of the tail t. Instead, semantic similarity bases its scoring measurement on the likelihood of a triple with semantic matching, this implies finding a multiplicative matrix that depends on the relationship, which is able to transform the head into the tail in the representation space, i.e., eq. (2.2) [21].

$$h^T M_r = t^T \tag{2.2}$$

Encoding Models takes care of the interactions of entities and relations through different types of model architectures. The most commonly used are linear/bilinear models, factorization models, and neural networks. Linear models formulate relations as a linear or bilinear mapping by projecting head entities into a representation space close to tail entities. Factorization aims to decompose relational data into low-rank matrices for representation learning. Neural networks encode relational data with non-linear neural activation and more complex network structures [21].

Finally, the last element that characterizes an embedding model is the ability to encode additional features to the entity or relationship embedding. This additional information can be textual information, type constraints or attributes that enrich embedding. Additional information is used when the loss function is calculated. An example of additional textual information can be a label that categorizes the type of entity helping in the calculation of embedding. In general, is defined an additional loss function on information and improve embedding by minimizing a loss function which is given by the sum of the loss to represent entities and relation and the loss for additional information [40].

The most commonly used representation space is the Euclidean point-based space by embedding entities in vector space and modelling interactions via vectors, matrices or tensors. Ultimately, it is difficult to determine which combination of these features is best for the application domain. From time to time there is to evaluate the characteristics of the available KG, the type of relationships and properties that are need to represented and above all to establish a trade-off between the completeness of representation and its complexity in relation to the resources available [21].

2.3 Knowledge Acquisition

Knowledge acquisition tries to construct knowledge graphs from unstructured text such as in our case, complete an existing knowledge graph inferring new entities or relation for example, and recognize entities and relations based on some characteristics. The advantages of using a Knowledge Graph (KG) are obvious: it allows to organize the vision of the domain in a logical, synthetic and above all semantic way. In doing so, the information contained in each triple do not know in their own right but they are part of a wider context. Their applications can be very broad and can make many context-aware applications. However, in the creation phase of the KG it is not always possible to represent the domain in a complete way, due to lack of data for example. Knowledge Graph Completion (KGC) helps with this task. Typical subtasks include Link Prediction (LP), entity prediction and relation prediction. The triple classification, on the other hand, defines a classifier that results in providing an indicator if a triple is correct.

A formal definition of these tasks can be provided as follows definition 2.

Definition 2. Knowledge Graph Completion Given an incomplete knowledge graph $G = (E, R)$, KGC infers missing triples
$T = \{(h, r, t) \mid (h, r, t) \notin G\}$.

There are various ways to implement the Knowledge Graph Completion (KGC) but the most common is based on the use of the embedded representation of the Knowledge Graph (KG) and is called embedding-based methods. However, this type of method fails to recognize multi-step relationships where methods such as relation path inference and rule-based reasoning are more effective.

Rule-based Reasoning uses a set of rules to be applied to the KG to infer new elements in the graph. For example, if one considers a KG that represents the relationships between people and one can have, $< Bob, isSonOf, John >$, $< John, hasChild, 1 >$, then, if not present, can be formulated rules so that if a person has children, and a child "recognizes" the father then one can infer the triple in which the father recognizes the son by using a different type of relationship and creating a new triple $< John, isFatherOf, Bob >$.

Relation path reasoning wants to leverage path information over the graph structure. To do so random walk inference has been used, instead, in the case of Path Ranking Algorithm (PRA), that chooses relational path under a combination of path constraints, and conducts maximum-likelihood classification. To improve the quality of path search is possible to add other information such a textual content to increase the similarity [21].

2.3.1 Calibration

Once the embedding matrix is available it is possible to use these values, together with the score function, to derive a probability of the existence of a triple in the graph. However, this probability is not said to have a probabilistic sense, since the expressed probability value may not correspond to the frequency of the events that are actually observed. In order for the predicted probability to make sense, it is necessary to calibrate the results. Ideally, one would like a predicted result with 90% confidence to be true in 90% of cases. So using a dataset not used in the training phase, the probability of each prediction is calculated and compared if this result is found in reality: the calibration tries to match these values in such a way that the estimated probability value respects reality. For example, a result predicted at 90% but which is actually true few times will undergo a greater calibration and will have a final probability value lower than a predicted result at 80% but which occurs hundreds of times after it also undergoes a calibration. Calibration is crucial in high-stakes scenarios such as drug-target discovery from biological networks, where end-users need trustworthy and interpretable decisions [46].

Definition 3. Calibration Given a knowledge graph embedding model identified by its scoring function f_m, with $f_m(t) = \overline{p}$, where $\overline{p}$ is the estimated confidence level that a triple $t = (s, p, o)$ is true, we define f_m to be calibrated if $\overline{p}$ represents a true probability [46].

There are two popular calibration methods: Platt's scaling and isotonic regression. Platt's scaling amounts to training a logistic regression model on the classifier outputs with respect to the true class labels. The second popular method of calibrating is isotonic regression. The idea is to fit a piecewise-constant non-decreasing function instead of logistic regression.

Calibration is a topic in which there is great interest in recent years and among the most recent progress: Tabacof et al propose a calibration technique in the event that in the dataset used for calibration there is no triple for which a probability value is associated. The non-existence of this triple in the dataset does not necessarily imply that the probability associated with the predicted triple does not reflect the reality [46].

2.3.2 Evaluation metrics

As been said in the chapter 1, in the case of the knowledge graph, the drug repurposing task translates into the link prediction task to predict possible new links between drug and disease.

Definition 4. Link Prediction Given a training set G that includes only positive triples, the goal of Link Prediction (LP) is assigning a score $f(t) \in R$ proportional to the likelihood that each unlabeled triple t included in a held-out set S is true. Note S does not have ground truth positives or negatives [46].

The task of learning from a Knowledge Graph (KG) is connected to the rank task. The score function forms the embedding model provides a way to get a score from a general triple. This score, rank, is used to measure the performance of the embedding model. The metrics that are often used in these kinds of projects are Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hits@N.

Although the definition of metrics is very clear, special attention should be paid to how they are used. It is not a simple thing, working with entities that together with relationships represent complex concepts, to evaluate the work of embedding. For example, using metrics between entities of a different type of which have not been observed in the training phase would mean going to verify the existence of a link with a new entity and not a simple link between already existing entities.

Definition 5. Mean Rank Mean Rank (MR) is a measure to evaluate systems that return a ranked list of answers to queries. Equation 2.3 express the formula to compute a Mean Reciprocal Rank (MRR) over a set of queries Q. $rank_i$ is the rank of the i-th query [7].

$$MR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} rank_i \tag{2.3}$$

Lower is better. The range is from 1 to $|\mathcal{E}|$, where $|\mathcal{E}|$ is the number of entities in the Knowledge Graph (KG).

Definition 6. Mean Reciprocal Rank MRR is a measure to evaluate systems that return a ranked list of answers to queries. Equation 2.4 express the formula to compute a MRR over a set of queries Q. $rank_i$ is the rank of the i-th query [7].

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q} | \frac{1}{rank_i} \tag{2.4}$$

Higher is better. The range is from 0 to 1.

Definition 7. Hits@N Hits@N is the proportion of the original correct entity to the top N predictions. It is the count of how many positive triples are ranked in the top-n positions against a bunch of negative triples. So the idea is to evaluate the ranking of the positive triple compared to a set of negative triples and see how the positive triple compares to the others. In this way, one can have an idea of when the Knowledge Graph (KG) embedding is accurate and how accurate future forecasts are. The value N, therefore, indicates the observation range in which the ranking of the positive triple must fall to have a hit or vice versa a miss. The more other the value of N is, the more likely the positive triple is in the prediction N prime. Equation 2.5 express the definition of Hits@N in formula [7].

$$Hits@N = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \begin{cases} 1 & \text{if } rank_{(h,r,t)_i} \leq N \\ 0 & \text{Otherwise} \end{cases} \tag{2.5}$$

Where Q is a set of triples and $(h, r, t) \in Q$.

Higher is better. The range is from 0 to 1.

Definition 8. Corruption Corruption is carried out both during the model training phase and during the evaluation phase, in which the above mentioned metrics are calculated. Corruption of a triple that is present in the data-set i.e. a positive triple, involves the replacement of one or more elements at a time in order to create a new triple, i.e a negative triple. So can be chosen to corrupt or just the head or just the tail or both of the triple and replace them with random elements or with specific elements. In this way, negative sampling is done during the training phase. For each positive triple, it is decided how many negative triples must be generated in order to provide more information to the model on which to train. This method avoids the problem of vanishing gradient and thus obtain better performance [55]. Similarly, during the evaluation phase, to calculate the accuracy of embedding, given a triple positive, in the case of link prediction, the head or tail or both with other entities are corrupted and their ranking is calculated. By doing this, one is able to establish whether, the ranking deriving from the score function, is higher in the case of the positive triple, or rather of the true triple and lower in the others.

Definition 9. Type Constraint Type Constraint is used in the corruption stages. Once established which elements of the triple to corrupt, it is necessary to decide with which entities to replace the element in question. Replacing it with a random entity opens a line of research on how to go to extract another entity but it is always good to extract it from a subset of all possible entities since these constraints impose prior beliefs upon the structure of the embedding space, without negative impacts on efficiency or scalability [55, 11]. It is often chosen as a set to replace an entity with all other entities that appear in the data set with the same relationships.

Definition 10. Raw Raw is an evaluation setting that ranks the scores in ascending order, then can get the rank of the original correct triplet to compute the evaluation metrics. Then, the same procedure, is repeated removing the tail t instead of the head h of a triple [51]. So valid entities outscoring the target one are considered as mistakes. In equation 2.6 is presented the rank of target tail, t, given a triple $(h, r, t) \in Q$

$$r_t = |\{e \in \mathcal{E}\backslash\{t\} : f_m(\mathbf{h}, \mathbf{r}, \mathbf{e}) > f_m(\mathbf{h}, \mathbf{r}, \mathbf{t})\}| + 1 \tag{2.6}$$

The raw rank in head prediction can be computed analogously.

Definition 11. Filter

Definition While corrupted triplets that appear in the train/valid/test sets (except the original correct one) may underestimate the metrics, are filtered out those corrupted triplets before getting the rank of each testing triplet and this process is called Filter [51]. So valid entities outscoring the target one are not considered mistakes but they are skipped when computing the rank. Both the raw and the filter configuration can be combined with the use of type constraints. In equation 2.7 is presented the rank of target tail, t, given a triple $(h, r, t) \in Q$

$$r_t = |\{e \in \mathcal{E}\backslash\{t\} : f_m(\mathbf{h}, \mathbf{r}, \mathbf{e}) > f_m(\mathbf{h}, \mathbf{r}, \mathbf{t} \wedge < h, r, t > \notin Q)\}| + 1 \tag{2.7}$$

The raw rank in head prediction can be computed analogously.

Definition 12. Tie A tie is an event when the target entity obtains the same score as other ones. This event can be handled as follows [40]:

- **min**: the target is given the lowest rank among entities in tie
- **average**: the target is given the average rank among entities in tie
- **random**: the target is given a random rank among entities in tie
- **ordinal**: the entities in tie are given ranks based on the order in which they have been passed to the model
- **max**: the target is given the highest (worst) rank among the entities in tie

2.4 Embedding Model

Following the definition of Knowledge Graph Embedding (KGE) in section 2.2, now is possible to define a Knowledge graph embedding models. These are neural architectures that encode concepts from a Knowledge Graph (KG) G that is composed by a set of entities E and relations R, into low-dimensional, continuous vectors $\in R_k$ that is the embedding of dimension k. Embedded representation of KG elements are learned by training a neural architecture over G, that represents the input to the neural network. Although such architectures vary from model to model, the training phase always consists in minimizing a loss function L that includes a scoring function f_m, i.e. a model-specific function that assigns a score to a triple $t = (h, r, t)$. The score function takes as input the embedding of the subject, the head h, the relation or predicate r and the tail or object t. The final goal of the optimization procedure is learning optimal embeddings, such that the scoring function f_m assigns high scores to positive triples $t+$ and low scores to triples unlikely to be true $t-$ [46].

Since there are many embedding models, which as been have explained above differ in their characteristics, after a preliminary analysis that is going to be explained in chapter 5, have been chosen to use TransE [4] and ComplEx [49] as embedding models which represent an excellent trade-off between completeness, accuracy and computational cost in relation to this project which aims to explore this research technique in this new sector.

2.4.1 TransE

TransE is a canonical model which is easy to train, contains a reduced number of parameters and can scale up to very large databases. TransE is able to model relationships by interpreting them as translations operating on the low-dimensional embeddings of the entities. Despite its simplicity, this assumption proves to be powerful since extensive experiments show that TransE significantly outperforms the state-of-the-art [4].

The main idea behind the concept of translations in the embedding space, as show in fig. 2.1, is that if (h, r, t) holds, so it is a positive triple

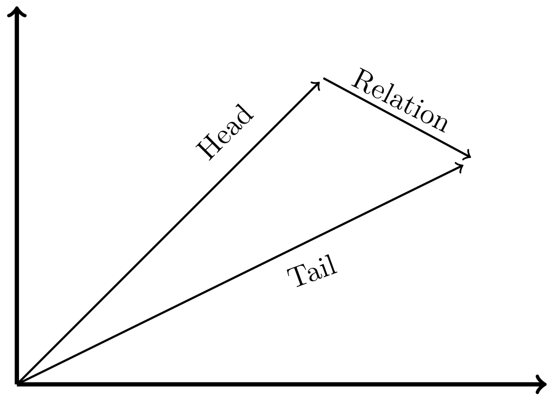


Figure 2.1: Translations in the embedding space of TransE. The vector representation of the head plus the vector representation of the tail should have as result the vector representation of the tail of the triple.

from the training set, then 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. Our approach relies on a reduced set of parameters as it learns only one low-dimensional vector for each entity and each relationship [4].

The main motivation behind this translation-based parameterization is that hierarchical relationships are extremely common in KG and translations are the natural transformations for representing them [4].

In section 2.4.1 is presented the pseudo-code of TransE regarding the input, the initialization and the training procedure [4]. The training phase stops when a certain accuracy, according to the a type of measurements is achieved on the validation set. This code wants to represent the backbone of the embedding model but is not complete: for example are missing techniques of negative sampling or specific corruptions using type constraints.

Given a training set S, i.e. a subset of the original KG of triplets (h, r, t), where each triple is composed of two entities $h, t \in E$ (the set of entities) and a relationship $r \in R$ (the set of relationships). The goal of the training phase is to learn the embedded representation of entities and relations. The embedded representation is a vector of dimension k, that is a hyperparameter that represents the dimension of the embed-

Algorithm 1 TransE

```
Require: Training set $S = (h, r, t)$
Require: Entities and relation sets E and R
Require: Margin $\gamma$
Require: Embedding dimension $k$
Ensure: $r \leftarrow uniform(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})\ \forall r \in R$
Ensure: $r \leftarrow \frac{r}{||r||}\ \forall r \in R$
Ensure: $e \leftarrow uniform(-\frac{6}{\sqrt{k}}, \frac{6}{\sqrt{k}})\ \forall e \in E$
 1: loop
 2:    $e \leftarrow \frac{e}{||e||}\ \forall e \in E$
 3:    $S_{batch} \leftarrow sample(S, b)$ {Sample a batch of size b}
 4:    $T_{batch} \leftarrow \emptyset$ {Initialize the set of pairs of triplets}
 5:    for $(h, r, t) \in S_{batch}$ do
 6:       $(h', r', t') \leftarrow sample(S'_{(h,r,t)})$ {Sample a corrupt triple}
 7:       $T_{batch} \leftarrow T_{batch} \cup \{((h, r, t), (h', r', t'))\}$
 8:    end for
 9:    Update Embedding w.r.t.
10:    $\sum_{((h,r,t),(h',r',t')) \in T_{batch}} \nabla[\gamma + d(h + r, t) - d(h' + r, t')]_+$
11: end loop
```

ding. The embeddings values are denoted with the same letters but in boldface characters. The basic idea behind TransE is that the functional relation induced by the r-labelled edges corresponds to a translation of the embeddings, i.e. we want that $\mathbf{h} + \mathbf{r} = \mathbf{t}$ when (h, r, t) is a positive triple (t should be the nearest neighbour of $\mathbf{h} + \mathbf{r}$), while $\mathbf{h} + \mathbf{r}$ should be far away from $\mathbf{t}$ if it is a negative triple. This corresponds, in the flat case, to a simple sum of vectors. The sum of the head vector plus the relation vector must result in the tail vector for each triple present in the training set. Using an energy-based framework, the energy of a triplet is equal to d$(\mathbf{h} + \mathbf{r}, \mathbf{t})$ for some dissimilarity measured, which is taken to be either the L1 or the L2 norm [4].

To learn such embedding, for each batch, for each triple is computed a loss function 2.8, where is minimize a margin-based ranking criterion over the training set [4].

$$\sum_{(h,r,t) \in S} \sum_{(h',r',t') \in S'} \nabla[\gamma + d(h + r, t) - d(h' + r, t')]_+ \qquad (2.8)$$

where $\gamma > 0$ is a margin and $[x]_+$ stands for the positive part of x.

Following the snipped of code in **??**: first of all embeddings for entities and relationships are first initialized following a random sampling that depends on the dimension of the embedding. Then for each main iteration of the TransE algorithm, the embedding vectors of the entities are first normalized. Later, a subset of triplets is sampled from the training set and will serve as the training triplets of the batch. Usually, the dimension of the batch is chosen according to the capabilities of the machine, a good practice is to use a value for the batch size that is able to fit all the data in the volatile memory.

Therefore, following the algorithm, each triple of the batch is corrupted in the following way: following the equation 2.9 it is possible to understand that the corrupt triple will be constituted by a new entity instead of the head or tail but not both, maintaining the same relationship. As it was designed, the loss function, which is an energy function, will favour lower values for positive triples than for corrupt ones [3].

$$S'_{(h,r,t)} = \{(h', r, t) \in E\} \cup \{(h, r, t') \in E\} \tag{2.9}$$

Note that for a given entity, its embedding vector is the same when the entity appears as the head or as the tail of a triplet [4].

So the parameters are then updated by taking a gradient step with constant learning rate. Usually, the optimization is carried out by stochastic gradient descent over the possible h, l and t, with the additional constraints that the L2 norm of the embeddings of the entities is 1 [4]. This constraint is important because it prevents the training process to trivially minimize the loss function by artificially increasing entity embeddings norms [3]. Instead, by constraining the size of the embedding, one can force the model to learn the embedding instead of memorizing the values by simply increasing the norm.

TransE has been tested by its authors on Wordnet [1] and Freebase [2]. It was subsequently used as a baseline for all models developed in the future and for this reason, it is widely tested. The results in terms of accuracy are very good in relation to the complexity in terms of the number of parameters to be trained $\theta(n_e k + n_r k)$.

[1] `https://wordnet.princeton.edu`

[2] `http://www.freebase.com`

2.4.2 ComplEx

ComplEx is a model embedding technique and it is presented as an arguably simple approach to matrix and tensor factorization for link prediction data that uses vectors with complex values and retains the mathematical definition of the dot product, as it only uses the Hermitian dot product, the complex counterpart of the standard dot product between real vectors. In this way, using a composition of complex embeddings one can handle a large variety of binary relations, like symmetric and antisymmetric relations. ComplEx is scalable to large datasets as it remains linear in both space and time respect the dimension of the embedding K, $O(K)$ [49].

As always the objective of link prediction is to find an automatic to discover inference triples using the existing one. However, many relations are non-deterministic and the way a human reason is so far away from an embedded representation, that means a simple connection of facts, or in our case of triple, could not be so obvious. To address this problem, over the years a lot of solutions have been proposed, one of them focuses on the tensor completion problem, where each slice is the adjacency matrix of one relation type in the knowledge graph. A partially observed matrix or tensor is decomposed into a product of embedding matrices with a much smaller rank, resulting in fixed-dimensional vector representations for each entity and relation in the database [49]. As can be noticed in the real world, there are a lot of possible types of relations like symmetry, antisymmetry or transitivity. Dot products of embeddings scale well and can naturally handle both symmetry and antisymmetry of relations and using an appropriate loss function even enables transitivity. Looking to the past literature, dealing with antisymmetric relations has so far almost always implied a huge number of parameters, making models prone to overfitting. ComplEx presents the dot product between embeddings as a very effective composition function. Instead of using embeddings containing real numbers, ComplEx demonstrate the capabilities of complex embeddings. Dealing with complex vectors, the dot product is called the Hermitian dot product, as it involves the conjugate-transpose of one of the two vectors. As matter of fact, the dot product is not symmetric any more since it involves a conjugate and a transposition, and in this way, the order of presenting entities in the triples matter, therefore the antisymmetric relations can receive different scores. In such a way complex vectors can effectively capture antisymmetric relations while retaining the efficiency benefits of the dot product, which is linearity in both space and time complexity [49].

The loss function used in the model is presented in eq. (2.10). In bold the embedding of the head, tail and relation. K is the dimension of the embedding.

$$\phi(h, r, t) = Re(\sum_{k=1}^{K} \mathbf{r} * \mathbf{h} * \mathbf{t}) \tag{2.10}$$

2.4.3 Hyperparameters

Since in order to learn the embedding of a KG a training procedure is necessary, it is good to indicate which are the main parameters that can be modified.

1. Epochs : Number of epochs is the number of times the whole training data is shown to the network while training.

2. K: is the embedding dimension. Each entity and relation will be represented with a vector of R^k

3. η: is the negative sampling rate, control how many negative samples are generated. for each positive sample.

4. Lr: The learning rate defines how quickly a network updates its parameters. For each batch of the training a score function is computed. Base on this score the goal is to minimize the loss. To do so a gradient is computed that indicates the direction of the minimum of the loss function. The learning rate tells how much big the step should be in the direction of the gradient

5. optimizer: is the optimizer of the model such that SGD [41], ADAM [26] or ADAGRAD [12]. Specify how should be performed the optimization procedure after the loss function is computed.

6. Batch Count: the training set will be divided is in such a way the number of batches correspond to the batch count

7. Model Family: is type of embedding model family to train the KG

8. Others: other parameters such as early stopping, adaptive learning rate, normalization

2.5 Knowledge Graph Creation

In order to be able to do drug repurposing, first of all, it is necessary to go to represent the domain. In our case, the application domain was built starting from databases concerning genes, proteins, drugs and disease. It was necessary to combine multiple databases from different sources to have a good number of triples to work with and above all because there is no cured biomedical database that contains all this different information and also to connect the different identified ones used in the various databases. In our specific case, DrugBank, CTD, OMIM and UniProt have been used. They are widely used research and industrial databases that have an internationally recognized and have a public access or academic license option.

The figure 2.2 shows the general structure of the domain that is used in this project, indicating the entities and the relationships that connect them. Our KG uses proteins, genes, diseases and drugs as entities. The main connections that are formed are: A gene encodes a protein. A protein interacts with another protein or itself. A protein is involved in the illness. A protein has different types of interactions with a drug. And a drug is used to treat a disease. The disease is part of a typology, therefore it has a parental relationship with another disease.

After extracting millions of information from databases, parsing and filtering the data is built a KG having about 180k triples among 60k entities using about 70 different relationships. The KG built in this way is not a cornerstone for its size but it means that it is possible to integrate information already present in a not too elaborate way, and therefore this approach is easily improved, replaceable and extensible.

The pie chart in the fig. 2.3 represents the percentage of entities of each type with respect to the total of entities present.

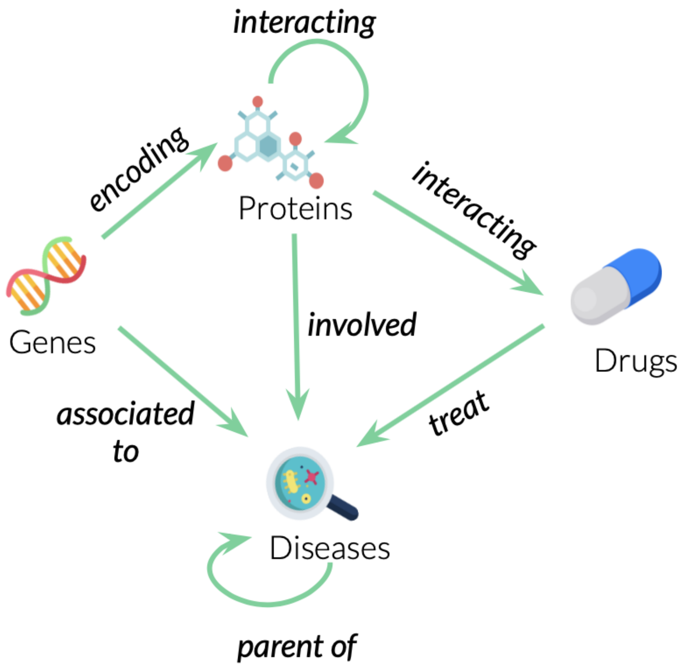


Figure 2.2: The domain under analysis for drug repurposing represented with a KG with the different types of entities and the main relations between them.

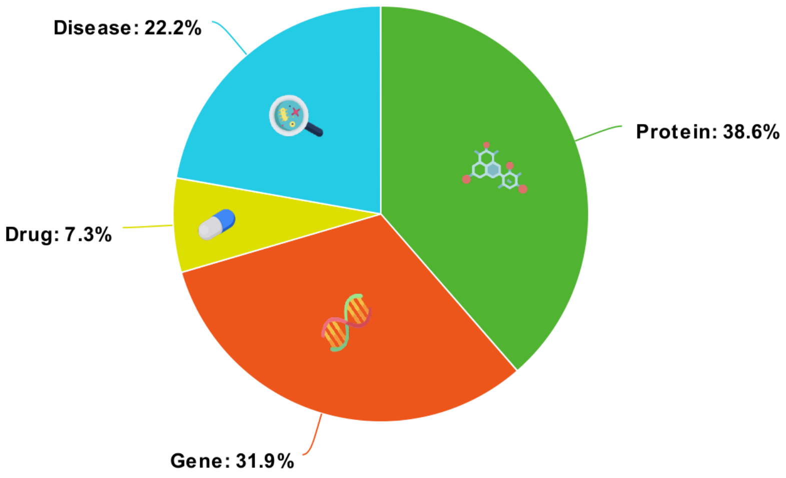


Figure 2.3: Pie chart representation of the % of all different types of entities that belongs to the knowledge graph used for drug repurposing.

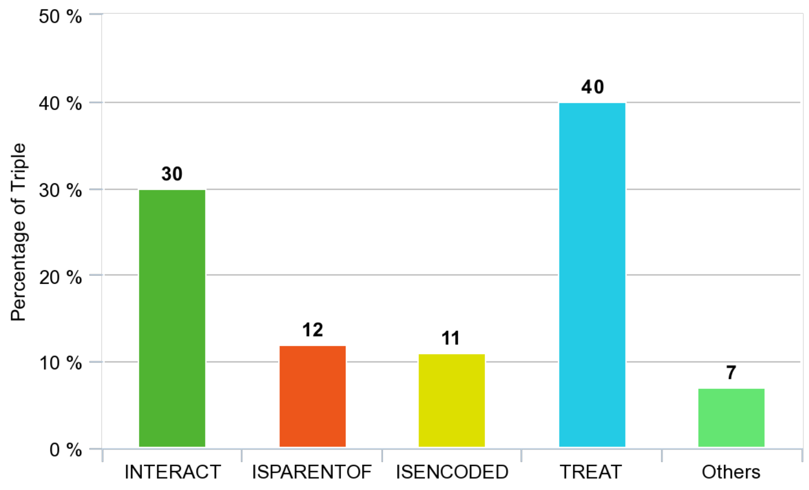


Figure 2.4: Bar chart that represent the main types of relation with the percentage w.r.t. the total amount of triple present on the KG for this research.

In the bar chart in fig. 2.4, on the other hand, the most frequent relationships are represented in percentage with respect to the totality of triples in the KG. Can be observed that the four most frequent relations in KG constitute about 90 % of the triples present in the graph.

Finally, we want to give, with the next sections, a quick presentation of the databases used to make up the KG.

2.5.1 DrugBank

DrugBank is a web-enabled database containing comprehensive molecular information about drugs, their mechanisms, their interactions and their targets. It is a unique bioinformatics and cheminformatics resource that combines detailed drug data with comprehensive drug target information. First described in 2006, DrugBank has continued to evolve over the past 12 years in response to marked improvements to web standards and changing needs for drug research and development [50].

The latest release of DrugBank at the current version 5.1.7 of 2020-07-02 contains 13,604 drug entries.

Each entry contains more than 200 data fields with half of the information being devoted to drug/chemical data and the other half devoted to drug target or protein data. DrugBank is widely used by the drug industry, medicinal chemists, pharmacists, physicians, students and the general public. Its extensive drug and drug-target data have enabled the discovery and repurposing of a number of existing drugs to treat rare and newly identified illnesses [50].

2.5.2 CTD

Comparative Toxicogenomics Database (CTD) is a database that is widely used from an companies and public medical authority, it is a publicly available database that aims to advance understanding about how environmental exposures affect human health. It provides manually curated information about chemical-gene/protein interactions, chemical-disease and gene-disease relationships so that's the reason why is a cornerstone in the field of biomedical databases. The initial release of CTD was on November 12, 2004. CTD has various entities in the database such as drugs, diseases, genes, protein, GO and pathways. Regarding the chemical, it integrates a drug subset of the Medical Subject Headings (MESH), the hierarchical vocabulary from the U.S. National Library of Medicine. CTD's MEDIC disease vocabulary is a modified subset of descriptors from the Diseases category of the U.S. National Library of Medicine (NLM) MESH, combined with genetic disorders from the Online Mendelian Inheritance in Man (OMIM) database. CTD biocurators mapped OMIM diseases to terms within the hierarchical MESH disease vocabulary to expand our disease representation just to overline the importance of manual curation of this database. The CTD gene vocabulary (symbols, names, and synonyms), instead is a derivation from the Gene database at the National Center for Biotechnology Information (NCBI). CTD contains curated and inferred gene-disease or chemical-disease associations. These types of associations are extracted from the published literature by CTD biocurators, or are derived from the OMIM database using the mim2gene file from the NCBI Gene database. Inferred associations, instead are established via CTD-curated chemical-gene or disease-gene interactions [8].

2.5.3 UniProt

The Universal Protein Resource or UniProt is a collection of sequences and annotations for over 120 million proteins across all branches of life. Detailed annotations extracted from the literature by expert curators have been collected for over half a million of these proteins. Our interest is in this subset of this database called "Swiss-Prot" that contains around 20k different humans proteins with all their interactions. These annotations are supplemented by annotations provided by rule-based automated systems, and those imported from other resources [6].

2.5.4 OMIM

OMIM is a comprehensive, authoritative compendium of human genes and genetic phenotypes. The consultation is free under academic license and is constantly updated. The entire collection contains information on all known mendelian disorders and over 15,000 genes. It focuses on the relationship between phenotype and genotype [15]. The main purpose of these database for this project is to complete a vocabulary for diseases to connect some fields of the other database used for the construction of the KG.

2.6 Biomedical Vocabulary

Often in the literature and in this thesis, terms related to the biomedical world that are not in the common vocabulary can be used. It is therefore good to try to give an intuitive definition of the meaning of each of them.

- Gene. A gene is the basic physical and functional unit of heredity. Genes are made up of DeoxyriboNucleic Acid (DNA). Some genes act as instructions to make molecules called proteins. However, many genes do not code for proteins. Every person has two copies of each gene, one inherited from each parent. Most genes are the same in all people, but a small number of genes (less than 1 percent of the total) are slightly different between people. Scientists keep track of genes by giving them unique names. Because gene names can be long, genes are also assigned symbols, which are short combinations of letters (and sometimes numbers) that represent an abbreviated version of the gene name [U.S. National library of medicine (NLM)].

- Protein. Proteins are large molecules composed of one or more chains of amino acids in a specific order determined by the base sequence of nucleotides in the DNA coding (gene) for the protein. The complex structures of different proteins give them unique properties. Proteins are very important for the structure, function, and regulation of the body's cells, tissues, and organs. Enzymes are proteins that speed up biochemical reactions in cells that otherwise take so much longer and life could not be possible.

- Pathway. Pathways are a collection of manually drawn pathway maps representing our knowledge on the molecular interaction, reaction and relation networks for human diseases, drug development, gene information, metabolism etc. [23].

- Gene Ontology. An ontology is a representation of something we know about. "Ontologies" consist of representations of things that are detectable or directly observable and the relationships between those things. The Gene Ontology project provides an ontology of defined terms representing gene product properties [5].

- Side effects. A side effect is a secondary, typically undesirable effect of a drug or medical treatment.

State of the Art 3

Ogni progresso é dovuto agli scontenti.
Le persone contente non desiderano alcun cambiamento.

In this chapter, the state-of-the-art work that is of interest for this thesis is presented. In the first place, is given an introduction about the current expertise in the subject of drug repurposing from a medical and pharmaceutical point of view, then is presented the main Knowledge Graph (KG) embedding techniques. Later is offered an overview of similar works that are already present in literature and finally is presented a similar biomedical knowledge graph that is built from several databases.

First of all, is provided with a brief overview of the pharmaceutical techniques used for drug repurposing highlighting the drawbacks and the importance of this study.

Once it is clear what the objective of this research is and what advantages it introduces, has been investigated in section 3.1 the state of the art regarding the main computational techniques applied to the field of drug repurposing.

Subsequently, is presented the most advanced techniques concerning the embedding of KGs in section 3.2, to then analyze the previous literature, in section 3.3, which used this machine learning technique to make an inference of biomedical links in KG.

In the end, in section 3.4, is introduced the main knowledge graphs in the biomedical field used for purposes similar to that of this thesis.

3.1 Drug Repurposing

Over time, scientific research techniques, still valid and used today, have left room for computational research techniques, allowing in a shorter time, at a lower cost, a higher number of experiments. Furthermore, these techniques can be easily and quickly applied to different targets, so they represent a general approach to solving the problem [47].

For this reason, computational approaches are largely used but they are data-driven. That means they need for a large number of informations that are not always available. A computational approach uses a systematic analysis of data of any type such as chemical structure, phenotypes, genotypes, gene expression, protein interaction and so on and so forth. All this kind of data can bring enough information that then leads to the formulation of repurposing hypotheses.

After presenting the main families of approaches to the drug repurposing problem in chapter 1, we now want to present the main computational techniques following the survey of [39].

3.1.1 Signature Matching

Signature matching, as the name suggests, try to compare unique characteristics of the drug or disease to see if there is some sort of relation between them [25]. The signature of a drug could be derived from three general types of data: transcriptomic such as RNA, metabolomic or proteomic information; adverse event profiles or chemical structures [39]. Matching transcriptomic signatures can be used to estimate the similarity between a drug and a disease or between two drugs. To measure the similarity between a drug and a disease first is needed to collect the signature of the drug: to do so, one can derive the signature by checking if the gene expression of a biological element such as a cell is changed after the treatment with the drug under investigation. The difference in gene expression between before and after the treatment is the signature of the drug. Once the signature is available, one can compare it with the expression profile of a disease that is obtained with a similar procedure but now is compared to the healthy stage with the ill stage [39]. The Signature Reversion Principle (SRP) supports this approach: if a drug can reverse the expression pattern of a given set of genes that are characteristic for a particular disease phenotype during the treatment, then that drug might be able to revert the disease phenotype itself [19]. Regarding the comparison of a drug with another drug to derive a similarity between them, means that looking for a way to identify shared mechanisms of action. To have a shared therapeutic application, and so shared potential off-target effects, the drugs have to share the transcriptomic signature [18]. Both approaches that use transcriptomic signatures rely on the gene expression to derive their signature, which implies the need to access a valid and huge amount of data to analyze and process.

The second type of signature matching used in drug repurposing is based on chemical structures that are used to understand the biological effects, hence if two drugs share a similar chemical structure, likely they share a biological activity. In order to do that, is necessary to identify a set of properties of the drug that are an expression of its structure and build a network based on shared chemical features [39].

Another strategy is to use side effects to identify the possible similar drug. This approach is base on the fact that two drugs that cause the same adverse effects may be acting on a shared target or protein or on the same pathway [39].

3.1.2 Genome-wide association studies

In the last years, the cost of the mapping of human DNA is going down. This is possible thanks to the lower price of technologies, called genotyping technology, but also is due to a large increase in the number of Genome-wide Association Studies (GWAS). A project leader in this field is the Human Genome Project that has brought to the whole field of research dwindling genotyping costs. The purpose of GWAS is to identify genetic variants associated with common diseases and thereby provide insights into the biology of diseases: the data obtained may also help identify novel targets, some of which could be shared between diseases treated by drugs and disease phenotypes studied by GWAS and thereby lead to repurposing of drugs. The drawbacks of this approach are the lack of information on the direction of effect of the gene variant and every time one needs to study a new effect of a gene is needed an entire functional study that implies time and money.

3.1.3 Computational molecular docking

The idea behind molecular docking is to leverage prior knowledge of a receptor target that is involved in a disease. It is a structure-based computational strategy to predict binding site complementary between a ligand, a drug, that tries to attach itself to a target, a receptor, in the docking site so it could stop or start a particular behaviour that is able to counterbalance the disease. So molecular docking, that is a structure-based computational strategy, interrogates multiple drugs against a particular target. Viceversa, parsing drug libraries we could explore target receptors to identify novel interactions, i.e inverse docking, that can be taken forward for repurposing. The main problems of such an approach are: structures for some protein targets of interest may not be available and there are no well-curated macromolecular target databases that provide accurate structural information [39].

3.1.4 Pathway or network mapping

The pathway-based or network-based approach is another way to find possible candidates for drug repurposing. There several different ways to find a possible drug target but doesn't mean that this information could be immediately used because we don't have the certainty that the drug will attach in a specific target for example. In such circumstances, a pathway-based strategy may provide information on genes that are either upstream or downstream for the gene-disease association. Hence, a network analysis that consists of the construction of a drug or disease networks based on gene expression patterns, disease pathology, protein interactions to help in the identification of repurposing candidates [39].

3.1.5 Retrospective clinical analysis

Once a novel drug is available on the market, its research study doesn't stop there. As we explained in chapter 1, the drug is constantly under observation even after the start of commercialization. During the stage is not rare that some other side-effects could appear, even if they don't hazard the health of the patient. For this reason, since now the drug is

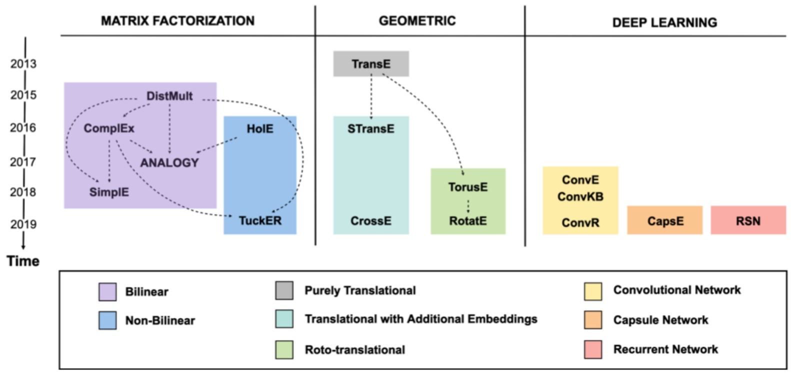


Figure 3.1: Taxonomy representation of embedding models. Dotted arrows indicate that the target method builds on the source method by either generalizing or specializing the definition of its scoring function. Figure from Rossi et al. [40].

consumed by a larger set, can be studied deeper and hence we can find another purpose to it [39].

3.2 Knowledge Graph Embedding

Rossi et al. [40] has proposed a novel taxonomy where it defines three main families of models with its subfamilies. In its survey, [40] decided to include the most valid representative models. The selection is based on the importance of the paper and the performance achieved by the model, preferring the model with a public implementation and ordering them by year of publication. In the end, 16 models are discussed, analyzed and tested for comparative analysis. In figure 3.1 is represented the taxonomy of the models for embedding as proposed by [40].

3.2.1 Tensor Decomposition Models

Embedding models that are based on tensor representation imagine the KG as a 3D adjacency matrix. This matrix, that should tell us all the connection in the domain, actually is incomplete. The embeddings of entities and relationship are just a decomposition of such matrix and to compute the score of a fact, a triple, we should combine the embedding of elements involved. The model learns the embedded representation optimizing the score function, and since, usually, the number of parameters involved is low, this makes them particularly light and easy to train.

3.2.1.1 Bilinear Models

These type of models use as scoring function the following equation where k is the dimension of the embedding, the symbol $\times$ is the matrix product.

$$\phi(h, r, t) = h \times r \times t, h \in \mathbb{R}^k, r \in \mathbb{R}^{k \times k}, h \in \mathbb{R}^k \tag{3.1}$$

The difference between these kind of models is on the way they learn the embedded representation using different constraints.

DistMult [53] The embedded representation of relation are diagonal matrices, in that way it simplify the number of parameters to be learn but force all the relation to be symmetric.

ComplEx [49] As discussed in 2.4.2, it uses the Hermitian instead of bilinear product, make possible the representation of asymmetric relation even if it uses the same structure as DistMult.

Analogy [28] It aims at modeling analogical reasoning. Impose that the matrices of relation are normal matrices and their composition has to be commutative.

SimplE [24] Uses the same idea as DistMult but it learn two embedding for each entities, one when they are used as head, the other when used as tail and also the relations have two diagonal embedding two represent the relation in the reagular and into the inverse use. In this way overcome the problem of symmetric relation.

3.2.1.2 Non-bilinear Models

They don't use the biliniar product to compute the score function.

HolE [38] The idea is to use the circular correlation instead of the product that is less expensive between head and tail, then it performs a matrix multiplication with the relation.

TruckER [1] Tucker decomposition can decompose a matrix in a set of vector with a shared core. This core is learned, and since it is shared is less expensive to learn. Trucker doesn't constraint the relation and entities to have the same size.

3.2.2 Geometric Models

Geometric models use the vector representation of entities and relation to project them into a geometric space using a transformation τ, in such a way, using a distance function δ, between the head and the relation with the tail is minimal. The equation for the score function is presented in 3.2

$$\phi(h, r, t) = \delta(\tau(h, r), t) \tag{3.2}$$

The models are different from each other by the definition of geometric space.

3.2.2.1 Pure Translational Models

They use a simple translation, i.e the embedding of the head is added to the embedding of the relation to gain the embedding of the tail. 2.4.1 is an example.

3.2.2.2 Translational models with Additional Embeddings

These kind of models associate to each element of the KG more than one embedding to learn specific relation or entities to be more specific. They use more paramaters rather than the other models.

STransE [37] To solve the issues of TransE to represent 1 to many, many to 1 and many to many relation, it uses a double embedding for the relation that helps to solve this problem and to be more relation specific.

CrossE [54] It use an additional relation specific embedding for each relation and then use the dot product, an element wise product, that allow to gain a triple-specific embeddings.

3.2.2.3 Roto-Translational Models

As the name suggest, instead of just perform a translation in the geometric space, they perform a combination of rotation and translation.

TorusE [13] TransE forces the entity embedding to lay into an hypersphere, here they use a torus to address this problem.

RotatE [45] The particularity of this approach is to use a relation that represents a rotation in the complex space. The relation embedding is a pure complex number that achieve a very good representation of symmetric and anti-symmetric relation, inversion and composition.

3.2.3 Deep Learning Models

Deep Learning Models use deep neural networks to learn the representation of the KG. In the literature, have been done an investigation about the types of layer, the activation function, the use of the bias and so on and so forth, but this study is out of the scope of this project, and for this reason we follow a simple presentation by [40]. In any case we should take in consideration that deep neural network are prone to recognize pattern but since KG embeddings are usually learned jointly with the weights and biases of the layers, makes the learning procedure potentially heavier, and more prone to overfitting.

3.2.3.1 Convolutional Neural Networks

These models use one or multiple convolutional layers that produce lower dimensional data from the input. The result is used to compute the score function.

ConvE [9] To compute the score of a triple first combine the head and relation embedding and reshape the result and it is used to feed a convolutional filter. The result is combined with the tail embedding to have the score of the fact.

ConvKB [36] The main difference from ConvE is that at the begging it combines head, realtion and tail into a matrix that is used as input to a convolution layer to derive the feature map. Finally, this is used with a dense layer with one neuron to guess the score function.

ConvR [22] Uses different dimension for the embedding of the entities and the relations. Due to this fact during the neural network is necessary a reshaping of the matrices

3.2.3.2 Others

Capsule Neural Networks Capsule networks are composed of groups of neurons, called capsules, that encode specific features of the input. Each capsule sends its output to higher order ones, with connections decided by a dynamic routing process. The probability of a capsule detecting the feature is given by the length of its output vector. An example is CapsE [35]

Recurrent Neural Networks They use recurrent layers to extract information from the training set as in the case of RSN [14].

3.2.4 Efficiency

For the sake of completeness, is reported the training times of the models mentioned above in relation to the results in terms of accuracy collected by [40]. This information is important since allows to understand which is the best embedding model to apply a biomedical knowledge graph.

In fig. 3.2 shows the training time of the model respect to the same amount of epoch for training.

Instead in table 3.1, is presented the ranking of accuracy of such model using as metric H@10.

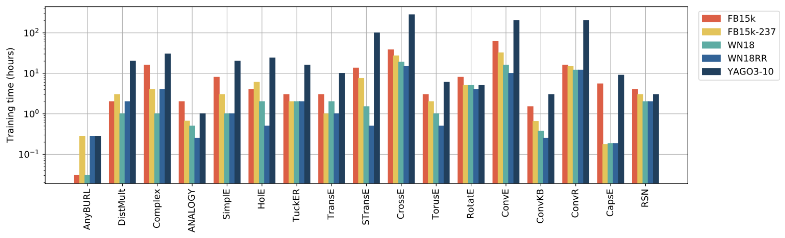


Figure 3.2: Training time of the model respect to FB15k, WN18, FB15k-237, WN18RR and YAGO3-10 databases. Figure from [40].

Model	FB15k	WN18	FB15k-237	WN18RR	YAGO3-10
DistMult	86.32	94.61	49.01	50.22	66.12
ComplEx	**90.53**	95.50	52.97	52.12	**70.35**
ANALOGY	83.74	94.42	35.38	38.00	45.65
SimplE	83.63	94.58	34.35	42.65	63.16
HolE	86.78	94.94	47.64	48.79	65.19
TuckER	88.88	95.80	**53.61**	51.40	68.09
TransE	84.73	94.87	49.65	49.52	67.39
STransE	79.60	93.45	49.56	42.21	7.35
CrossE	86.23	95.03	47.05	44.99	65.45
TorusE	83.98	95.44	44.71	53.35	47.44
RotatE	88.10	**96.02**	53.06	**57.35**	67.07
ConvE	84.94	95.68	47.62	50.75	65.75
ConvKB	40.83	94.89	41.46	52.50	60.47
ConvR	88.55	95.85	52.63	52.68	67.33
CapsE	21.78	95.08	35.60	55.98	0.00
RSN	87.01	95.10	44.44	48.34	66.43

Table 3.1: H@10 results for all Link Prediction (LP) models on each dataset. The best results of each metric for each dataset are marked in bold and underlined. Table from Rossi et al. [40]

The database that are used by [40] for this test are FB15k [1], WN18[2]

[1]`https://www.microsoft.com/en-us/download/details.aspx?id=52312`
[2]`shorturl.at/bfx78`

FB15k-237, WN18RR a subset of FB15k and WN18 and YAGO3-10 [3]. These databases are going to be presented in chapter 4.

Can be observed that ComplEx is a very good model in a various number of different data-sets. Instead TransE represents a good model compared to its computational cost. Regarding convolutional neural network models, even if they are so much expensive compared to the other models and sometimes they need as input a pre-trained network, are not able to achieve decent results.

3.3 Leveraging Knowledge Graph

In recent months there has been a growing interest in the field of knowledge graph applied to drug repurposing with the consequent increase in the production of academic articles. However, the field is still wide and the questions to be investigated are still many. In this section, to the best of our knowledge, we are going to present the main works on this topic ordered by publication date. Finally, we briefly summarize the current state-of-the-art of the topic.

3.3.1 Drug-Target

Mohamed et al. [32, 33] [4], between 2019 and 2020 presents an application of the Knowledge Graph Embedding (KGE) to complete KG by identifying new drug-target interactions and, therefore, the research aims to quickly identify new promising or unintended effects of drugs. In this way, identifying new drug-target relation is possible to operate on generic similarities of drug structure and protein sequence, therefore they can provide efficient predictions on new chemicals. They build a new biomedical knowledge database using the currently available dataset such as DrugBank [50], KEGG [5] [23] , Uniprot [6], InterPro [6] to create a novel KG composed by entities that are connected to both drugs and their potential targets. Regarding the completion task, in [32] they use ComplEx [49] model embedding for the task of KGE, but in [33] they

[3]`https://github.com/yago-naga/yago3`

[4]`http://hdl.handle.net/10379/16048`

[5]`https://www.kegg.jp`

[6]`https://www.ebi.ac.uk/interpro/`

develop a new knowledge graph embedding model, called TriModel, to learn vector representaions for all drugs and targets in the created knowledge graph that is combination of ComplEx [49] and DistMult [53] and is able to predict in a better way drug-target interactions. Combining these two approaches allow to achieve a scalable predictions of large volumes of drug-target interactions as it uses linear training time and constant prediction time. The results in the term of accuracy are very good but the project has limitations: the predictive capabilities are best suited to finding new associations between well-studied drugs and targets, but whether is needed to a prediction for a new drug, then the models that utilize drug structure and target sequence similarities will likely deliver better results [33].

3.3.2 Rare Disease

Natural Language Processing (NLP) is a field of computer science that concerns the interaction between machines and human languages such that a computer is able to process and analyze speeches and texts. In recent years, has been a rapid improvement in this field: such processes are now scalable and more precise. Applying NLP to PubMed abstracts, that is around 29 million of publications, one is able to create a graph of connection, relationship between the entities that NLP has recognized such as drugs, genes, proteins and diseases. The KG that is formed is called Global Network of Biomedical Relationships (GNBR).

The idea is to use GNBR to extract relevant information for rare diseases coming from disparate sources of knowledge to hypotheses some drug repurposing and mapping them directly back to the literature. GNBR contains edges or links between two entities from among the set of a gene, drug, disease and a support score (normalized between 0 and 1) representing the literature-derived confidence of the relationships between those two entities.

The main goal of this research was to find a repurposed drug for a rare disease. For this reason, they have been ranked based on the quantity and quality of information related to them deriving from the KG. In addition, these diseases are connected also to the current treatment. After that has been used a KGE that produces treatment hypotheses with strong evidence in the literature and evaluates our results using gold-standard drug indications. The same thing is been but for drug repurposing [43].

3.3.3 Knowledge-driven drug repurposing

Recent work by Zhu et al. introduce an approach to knowledge-driven drug repurposing based on a comprehensive drug knowledge graph. They build a drug-centric property graph model that is able to capture the general and essential information, revealing multiple aspects of the drug and its interaction with other biomedical entities. Due to the drug-centric property graph model, they were able to systematically extract and integrate data of six drug knowledge bases and construct the drug knowledge graph. In this research has been used path- and embedding-based data representation methods of transforming information in the drug knowledge graph into valuable inputs to allow machine learning models to predict drug repurposing candidates.

The path-based data representation method was able to provide local, yet very intensive information about interactions between drugs and diseases, while the embedding-based data representation method provided global and comprehensive information. Both methods applied to the drug knowledge graph produced high predictive results on diabetes mellitus treatments with certain machine learning models, while only using treatment information of other diseases.

The limitations of this study are related to not publish the KG due to license restrictions, they don't any negative samples in the test set and the evaluation is focused only on one disease [56].

3.4 Biomedical Knowledge Graph

The ogbl-biokg dataset is KG inside Open Graph Benchmark (OGB), which has been created using data from a large number of biomedical data repositories. It contains 5 types of entities: diseases, proteins, drugs, side effects, and protein functions. There are 51 types of directed relations connecting two types of entities, including 39 kinds of drug-drug interactions, 8 kinds of protein-protein interaction, as well as drug-protein, drug-side effect, drug-protein, function-function relations. All relations are modelled as directed edges, among which the relations connecting the same entity types (e.g., protein-protein, drug-drug, function-function) are always symmetric, i.e., the edges are bi-directional. This dataset is relevant to both biomedical and fundamental ML research. On the biomedical side, the dataset allows to get better insights into human biology and generate predictions that can guide downstream biomedical research. On the fundamental ML side, the dataset presents challenges in handling a noisy, incomplete KG with possible contradictory observations. This is because the ogbl-biokg dataset involves heterogeneous interactions that span from the molecular scale (e.g., protein-protein interactions within a cell) to whole populations (e.g., reports of unwanted side effects experienced by patients in a particular country). Further, triplets in the KG come from sources with a variety of confidence levels, including experimental readouts, human-curated annotations, and automatically extracted metadata.

After the machine learning task, the prediction task of new triplets given the training triplets is performed. The evaluation protocol use type constraints to rank against entities of the same type and just a random subset of 500 entities is used to rank it. [17]

In table 3.2 is shown the details about the type of entities present in the ogbl-biokg.

Entity	Quantity	%
Diseases	10,687	11.19
Proteins	17,499	18.66
Drugs	10,533	11.23
Side effects	9,969	10.63
Protein functions	45,085	48.07
Total	93,773	100

Table 3.2: Summary of the characteristics of entities' OGB knowledge graph. The number of entities for each type with their percentage respect to the total.

Implementation 4

Se non posso fare grandi cose
posso fare piccole cose
in un modo fantastico

This chapter presents the architecture of Knowledge Graph Embedding (KGE) applied to Drug Repurposing, together with design choices and the implementation of each component.

First, it is explained how to create a Knowledge Graph (KG) starting from heterogeneous dataset (section 4.1) and how to set up the environment to infer information from the KG(section 4.2.2). Then, it is presented the baseline to compare with our results (section 4.3), and how to split the dataset for training and validation (section 4.4). To validate the result, and compare it with other baselines, it is shown how to measure the accuracy (section 4.6) and the role of constraint (section 4.5) in this procedure. The last section, section 4.7, explains it is shown possible to deal with a huge graph to perform a consuming operation.

4.1 Knowledge Graph Creation

Section 2.5 presents the databases that are used to build the KG that has been proposed in this project. In this section, is presented the extraction methods used in each database and the triples produced by it.

To summarize for what are used each data-set:

- DrugBank
  - Drug Vocabulary
  - Drug-Protein interaction

  - Drug-Disease interaction

- UniProt
  - Protein-Gene interaction
  - Protein-Protein interaction
  - Protein-Disease interaction
  - Protein-Drug interaction

- CTD
  - Chemical Vocabulary
  - Gene Vocabulary
  - Disease Vocabulary
  - Chemical-Disease interaction
  - Gene-Disease interaction

- OMIM
  - Drug Vocabulary

4.1.1 DrugBank

As has been explained in section 2.5.1, DrugBank is one of the largest and most comprehensive collections of pharmacological data. Each entry in the database contains more than 200 data fields with half of the information being devoted to drug/chemical data and the other half devoted to drug target or protein data [50]. All the information is collected in an XML file, the overall scheme of which is expensive to report entirely 4.1 [1], for this reason only the portion of the scheme that is used during the extraction is reported.

Listing 4.1: Schema structure of XML DrugBank Database

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema>
    <xs:element name="drugbank" type="drugbank-type"></
        xs:element>
```

[1] https://www.drugbank.ca/docs/drugbank.xsd

```
    <xs:complexType name="drugbank-type">
        <xs:sequence>
            <xs:element name="drug" type="drug-type" maxOccurs="unbounded"/>
        </xs:sequence>
        </xs:attribute>
    </xs:complexType>

    <xs:complexType name="drug-type">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="1" name="drugbank-id"
                type="drugbank-drug-salt-id-type"> </xs:element>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="indication" type="xs:string"/>
            <xs:element name="synonyms" type="synonym-list-type"/>
            <xs:element name="products" type="product-list-type"/>
            <xs:element name="targets" type="target-list-type"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="synonym-list-type">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="0" name="synonym" type="synonym-type"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="synonym-type">
        <xs:simpleContent>
            <xs:extension base="xs:string"></xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="product-list-type">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="0" name="product" type="product-type"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="product-type">
        <xs:sequence>
            <xs:element name="name" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>

    <xs:group name="interactant-group">
```

```
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs=
                "0" name="polypeptide"
                type="polypeptide-type"/>
        </xs:sequence>
    </xs:group>

    <xs:complexType name="target-list-type">
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs=
                "0" name="target" type="target-type"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="target-type">
        <xs:sequence>
            <xs:group ref="interactant-group"/>
        </xs:sequence>
        <xs:attribute name="position" type="xs:integer"
            use="optional"/>
    </xs:complexType>

</xs:schema>
```

From DrugBank database are extracted three files deriving from three different kinds of information: the first one is the vocabulary, then is extracted the drug-protein interaction and finally the information if a drug is used to treat a disease.

4.1.1.1 Vocabulary

Using the information about the `drug-id` is possible to built a vocabulary of drugs. It associates the `drug-id` with the commercial name and all its synonymous. This dictionary is useful for the next steps because helps us to identify uniquely a drug with its different commercial names that are used in different countries. This kind of information is extracted from the tag `<synonym>` under `<synonyms>` and `<name>` under `<product>`.

4.1.1.2 Drug-Protein

There are a different kind of interaction between a drug and a protein. This kind of relations is reported in the tag `<target>`. One of the components of this tag is the type of action or interaction with the target in the object. The object, often, could be a protein, that under the tag

`<polypeptide>` is named with the `Uniprot ID`, otherwise could be a more complex concept like DNA, RNA etc. The last option is that the target is a drug itself, but here is reported with the commercial name, for this reason, the dictionary of drugs is helpful. Even if a target is present but no action is registered, then this information is not complete and for this reason, it is discarded.

4.1.1.3 Drug-Disease

The tag `<indication>` is very powerful information for our work since connects drugs to diseases. The problem with this information that is textual. It is a phrase that describes the therapeutic use for the drug. To extract the name of a disease from a text in an automatic way is used a Named-Entity Recognition (NER). NER is a task that wants to locate and classify named entities mentioned in unstructured text into pre-defined categories [34]. To solve this task one can use SpaCy [2], a library for advanced Natural Language Processing (NLP) in Python and Cython. SpaCy offers pre-trained models such us "en_ner_bc5cdr_md" that is trained to recognize disease and chemicals. For each entity that SpaCy is able to recognize in the "indication" field is translated into its Medical Subject Headings (MESH) or Online Mendelian Inheritance in Man (OMIM) ID. The result is a triple with the relation "TREAT" and the `DrugBank ID` of the drug with the ID of the disease.

4.1.2 UniProt

section 2.5.3 describes the idea of UniProt [6] and tells us that the investment of resources behind this project makes understand the importance of the role of proteins in life and consequently the non-marginal role they have in our KG. Swiss-Prot is an open database that allows extracting a lot of information in a personalized way. Among the many fields available, those of interest for the KG are: Protein, gene, involvement in disease, interacts with and pharmaceutical use. This information creates four different types of triples: Protein-Protein, Protein-Disease, Protein-Gene and Protein-Drug.

[2]`https://spacy.io`

Protein ID	Gene ID	Involvement in disease	Interacts with	Pharmaceutical use
Q99598	TSNAX	...	Q12904	...
Q9H489	TSPY26P	...	O95257	...
P49815	TSC2	Sclerosis	P62136	...

Table 4.1: Structure of UniProt database

4.1.2.1 Protein-Protein

A protein can interact with other proteins. This type of interactions is deeply studied. UniProt through the "Interacts with" field reports a list of proteins, using a `UniProt ID`. In the list, one can also find the words "itself". In this case, it indicates that the protein interacts with itself. For each element of the list, a triple is created with the protein in question with the wording "INTERACTWITH".

4.1.2.2 Protein-Disease

In the "Involvement in disease" field of UniProt, it is possible to find information that links proteins to the disease. If the content of that field is not a disease identifier, it may contain a PubMed code that identifies the scientific publication. Using this information one can build a triple that links a protein with a disease using the relation "ISINVOLVED".

4.1.2.3 Protein-Gene

Each protein is encoded by a gene, hence the `Gene ID` field. It reports the code and more synonyms for the gene, but since the protein is encoded by only one gene, it is sufficient to keep only the first name on the list and discard the remaining synonyms. For each gene-protein association, a triple is created with the relationship "ISENCODED".

4.1.2.4 Protein-Drug

To know which is the relation between a Protein and a drug the "pharmaceutical use" field helps. As in the case of DrugBank, it is textual information and for this reason is necessary to apply a ner to extract the drug name. Once one knows the commercial name of drug associates to a protein, the dictionary built from DrugBank is used to translate the name into a `DrugBank ID`. If the operation of finding the commercial name in the dictionary succeed, a triple with the relation "USEDIN" is created.

4.1.3 CTD

Comparative Toxicogenomics Database (CTD) wants to illustrate how chemicals affect human health as explain in section 2.5.2. CTD offers several databases but to build the knowledge graph are used only five of them: Chemical vocabulary, gene vocabulary, disease vocabulary, Chemical-Disease and Gene-Disease.

4.1.3.1 Chemical vocabulary

Chemical vocabulary is useful to increase the knowledge about drugs. It used in the following database to translate the ID used for drugs by CTD, that is a MESH code, to `DrugBank ID`, table 4.2. The translation from `MESH ID` to `DrugBankID` is made using the commercial name of the drug.

ChemicalID	MESH ID
methoxyaffinisine	C467068
succinoylryanodol	C481699
hydroxypalmatine	C549168

Table 4.2: Example of the structure of chemical vocabulary of CTD

4.1.3.2 Gene Vocabulary

Using this CTD file, the conversion between a literal name of a gene to a unique code is easier as in table 4.3. This dictionary is also useful for the completion and connection of another databases rather than CTD.

GeneSymbol	GeneID
AAW62_GT12	24251922
AA854_GT01	24287966
A9L33_GT15	27923927

Table 4.3: Example of the structure of gene vocabulary of CTD

4.1.3.3 Disease Vocabulary

Disease vocabulary is very helpful in two tasks table 4.4. The first one, using the fields `DiseaseName` and `DiseaseID`, the total disease dictionary can be improved and subsequently the extraction of information from other fields can be more consistent. The second task is the relationship between disease and another one. The CTD uses to identify disease a MESH or OMIM identifier. The structure of MESH representation is hierarchical, for this reason is possible to infer a parent relation. Given a disease is possible to create for each parent a triple using the relation "ISPARENTOF".

DiseaseID	DiseaseName	AltDiseaseIDs	ParentIDs
C535484	Deletion Syndrome	D002872	D025063
C567575	Hermaphroditism	D050090	D058490
D058186	Kidney Injury	C567006	D050398

Table 4.4: Example of the structure of disease vocabulary of CTD

4.1.3.4 Chemical-Disease

CTD offers a chemical-disease database that connects drugs to disease along with an inference score, like presented in table 4.5. The score represents the likelihood of that link exists. If the score is not present, means that the relation is manually curated and for this reason is certain.

Definition A percentile (or a centile) is a measure used in statistics indicating the value below which a given percentage of observations in a group of observations falls.

From this dataset is possible to create triple between chemical and disease using the relation "TREAT". One keeps only the information above the 90th percentile. The inference score is calculated as the log-transformed product of two common-neighbour statistics used to assess the functional relationships between proteins in a protein-protein interaction network. The first statistic takes into account the connectivity of the chemical and disease along with the number of genes used to make the inference. The second statistic takes into account the connectivity of each of the genes used to make the inference [8].

ChemicalID	DiseaseID	InferenceScore
C534883	D077192	4.67
C534422	D003924	9.94
C476756	D001249	13.51

Table 4.5: Example of the structure of chemical-disease file of CTD

4.1.3.5 Gene-Disease

As for chemical and disease, Gene-Disease is another database from CTD that connects gene with disease specifying a score for the existence of that link as in table 4.6. Hence can be created a triple among the relation "ISASSOCIATED" if the score is above the 90th percentile. The inference score is calculated as the logarithmic product of two common-neighbour statistics used to assess the functional associations between proteins in a protein-protein interaction network. The first statistic takes into account the connectivity of the gene and disease along with the number of chemicals used to make the inference. The second statistic takes into account the connectivity of each of the chemicals used to make the inference [8].

MIM Number	Preferred Title	symbol	Alternative Title	symbol
100070	Aneurysm	AAA1	Abdomin. Aneurysm	AAA

Table 4.7: Example of the structure of mimTitles file in OMIM database

GeneID	DiseaseID	InferenceScore
100174880	D000740	4.24
100528489	D007333	3.90
548848	D004412	4.75

Table 4.6: Example of the structure of gene-disease file of CTD

4.1.4 OMIM

In section 2.5.4 is presented OMIM that helps in the creation of a dictionary of disease connecting the OMIM identification number with all possible name of a disease. In table 4.7 is reported the structure of mimTitles file in OMIM database. For each OMIM code is shown all possible names, alternative names and symbols of a disease.

4.1.5 Biomedical Knowledge Graph

Collecting and processing all this information from different databases results in the creation of a KG containing 62,800 different entities that are connected between 69 relations among 183,029 triple. Table 4.8 shows the quantity for different kind of entities, instead, table 4.9 shows the eight most frequent relations in the dataset with their percentage respect to the total amount of triples.

Entity	Quantity	%
Drug	4,596	7.32
Disease	13,945	22.21
Gene	20,017	31.87
Protein	24,242	38.6
Total	62,800	100

Table 4.8: Summary of the characteristics of entities' knowledge graph

Relation	Quantity	%
TREAT	72,976	39.87
INTERACTWITH	55,521	30.33
ISPARENTOF	21,913	11.97
ISENCODED	20,219	11.05
ISINVOLVED	4,399	2.4
INHIBITOR	2,774	1.52
ANTAGONIST	1,748	0.96
AGONIST	1,180	0.64

Table 4.9: Summary of the characteristics most frequent relations' knowledge graph out 69 relations

4.2 Knowledge Graph Embedding

A fundamental part of this work was the choice and identification of a framework that would provide the necessary implementations to be able to create an embedding starting from one kg. On the internet, there are implementations, often provided by the authors themselves, of some embedding models but OpenKE [16] and Ampligraph [7] represent libraries capable of providing a wider selection of embedding models to choose from. Key features for our selection were: availability, a wide choice of model family, extensibility, ease of use and documentation. In addition to this, the training time is a fundamental requirement, and for this reason the Machine Learning (ML) tools such as Keras, TensorFlow

(TF) or Pytorch, could offer a faster Graphics Processing Unit (GPU) implementation in addition to the one in Central Processing Unit (CPU).

4.2.1 OpenKE

OpenKE is an open-source framework for knowledge embedding organized by the Natural Language Processing Lab at Tsinghua University (THUNLP). OpenKE is presented as an efficient implementation based on PyTorch for Knowledge Representation Learning (KRL). To do so, they use a layer in C++ to implement some underlying operations such as data preprocessing and negative sampling. For each specific model, it is implemented by PyTorch with Python interfaces so that there is a convenient platform to run models on GPUs. OpenKE is composed mainly of two repositories: The first one is based on PyTorch, the most developed and TF. In reality, the TF-based implementation is not optimized in C++ and contains fewer templates than the versions in Pytorch. For this reason, has been chosen to opt for this second version. The Pytorch version of the framework offered as embedding models:

1. RESCAL
2. DistMult [53], ComplEx [49], Analogy [28]
3. TransE [4], TransH [**t**], TransR [27], TransD [20]
4. SimplE [24]
5. RotatE [45]

However, since this framework is not flexible for user extensions, after a preliminary analysis of the various models, this framework has been abandoned. The main problems encountered were due to the implementation part of C++. This did not allow easy extensions or modifications of some parts also because of the library that is used to interface C++ to Python.

4.2.2 Ampligraph

AmpliGraph is a suite of neural machine learning models for relational Learning. AmpliGraph's machine learning models generate knowledge graph embeddings that are a vector representation of concepts in a metric space, for this reason, is feasible to discover new knowledge from an existing knowledge graph, complete large knowledge graphs with missing statements, or generate stand-alone knowledge graph embeddings. It then combines embeddings with model-specific scoring functions to predict unseen and novel links. Through its Application Program Interface (API) is easy to use and easily extensible and it is GPU-ready since is developed over TF, moreover, it has very good documentation with reference to papers to justify implementation.

AmpliGraph is composed of four modules:

1. Datasets: it helps to download a state-of-art dataset, load a custom one and offer several options to elaborate the dataset such as the splitting.

2. Models: it is a collection of knowledge graph embedding models. AmpliGraph contains TransE [4], DistMult [53], ComplEx [49], HolE [38], ConvE [9], ConvKB [36]

3. Evaluation: to measure the quality of embedding implementing different metrics.

4. Discovery: API for knowledge discovery, such as facts, cluster entities or predict near duplicates.

For each model, AmpliGraph includes the following components:

Scoring Function is given by the specific model.

Loss function has to be minimized during the training to learn the embedding. The loss functions are passed to models as hyperparameters, and they can be thus used during model selection. Ampligraph implemented Pairwise max-margin loss, absolute margin max-margin loss, self adversarial sampling loss, negative log-likelihood loss, multiclass null loss and Binary Cross-Entropy Loss.

Optimization The next component is the optimization step. The goal of the optimization procedure is learning optimal embeddings, such that the scoring function is able to assign high scores to positive statements and low scores to statements unlikely to be true.

Ampligraph provides, through TF implementation of Adam, AdaGrad, momentum and SGD.

Negatives generation The corruption strategy to adopt during the training, definition 8.

In addition to these features, the model supports L1, L2 and L3 regularization and various strategy to initialize the embedding parameters.

4.2.2.1 Fork

Although Ampligraph provided many more tools than OpenKE, the fact that it is easily extensible has allowed us to implement additional features that help us to better monitor the entire training phase. For this reason, has been necessary to start a fork from this project and implement the validation loss, early stopping, callbacks and visualization in TensorBoard. The implementation of these components was inspired by the Keras framework. They help to track more metrics during training and more arbitrary events, callbacks, which activated according to a certain criterion can, for example, save, validate or change some parameters of the model during training. Their implementation burdened the workload but allowed to divide an entire training into sub-parts by monitoring them more accurately.

4.3 Random Baseline

In order to understand if through embedding the model was really learning a meaningful representation of the KG, has been decided to implement a baseline to compare the results. The purpose of the baseline is to verify which results in terms of accuracy can be obtained in a validation set in which has been corrupted one of the two entities and measure the H@N score.

Ampligraph [7] provides a baseline implementation which is however unsuitable for the task of Drug Repurposing. Ampligraph randomly assigns a score from a uniform distribution. Once the score was available, the ranking calculation part had to be implemented. This approach was not chosen for the simple fact that the scores do not follow a uniform distribution, but in reality, they are a consequence of the energy function they represent. This energy function or score function will be similar for similar triples, and therefore the distribution is not uniform. Since the representations of the various ones in the embedding tend to form clusters and the embedding values sometimes assume a mono or bi-modal distribution, it would have been more appropriate to choose a different distribution. Another alternative would have been to initialize the embedding values, and then use these to calculate the score function.

Since choosing the right distribution is a rather difficult task, has been decided to approach the problem differently. H@N represents the proportion of triples corrected in the first N predictions. Following this idea, has been given N attempts in which the corruption of the triple can be the positive triple under observation from the validation set. In section 4.3 is shown the pseudo-code for our baseline.

One must provide the algorithm with a validation set, a relationship, which in our case will be "TREAT" since it represents the link between drug and disease, the H@N value that we want to calculate and the possibility of using the Type Constraints. If one wants to use the Type Constraints, is necessary to create a set of entities that are present in the validation set and are involved through the relationship taken as input. Otherwise, this set will consist of all possible entities. Again, the problem is relaxed, since are considered all the entities involved in the whole dataset and not just in the validation. However, this does not represent a problem since it results in an overestimation: by restricting the number of entities involved, the probability that the exact entity will be extracted is higher, increasing the value of H@N, thus showing a better baseline compared to a real one. After that, either only the head or only the tail of the triple gets corrupted randomly. This entity is replaced by another randomly chosen from the previously created set. The substitution occurs N times, in accordance with the index of H@N, if in the N substitution one gets the positive triple back then is registered a hit otherwise the prediction has failed and can not increase the index.

4.4 DataSet Split

Once a dataset has been created, it must be prepared for parameter training. Therefore, has been divided the dataset into several parts in such a way as to use each one for different purposes without overfitting the dataset. The largest part of the split concerns the training set, about 60% of the triples end up here, instead 20% in the validation set and the remaining 20% in the test set. These last two splits are used to fine-tune the parameters and to check the quality of the training on data not yet seen. To split the dataset one can use a function defined by Ampligraph. This function 4.2 carves out a test set that contains only entities and relations which also occur in the training set. In this way has been ensured that in the validation set and in the test set there are only entities that are already been seen in the training set, otherwise, is possible to evaluate the accuracy of the model on completely new entities, without their representation embedding matrix in the training set. Since our dataset contains many entities but is not particularly large, we have chosen to use the "allow_duplication" flag which allow to replicate triples both in the training set and in the validation set in order to complete the split respecting the constraint expressed by the "test_size" parameter. This variable indicates how many triples there must be in the validation set. The variable "X" instead is the input data set to be divided and "seed" is the seed for the random numbers which will then be used for the random split.

Listing 4.2: Signature of split function

```python
from ampligraph.evaluation import
    train_test_split_no_unseen

train_test_split_no_unseen(X, test_size=100, seed=0,
    allow_duplication=False)
```

4.5 Type Constraint

Type constraints allow us to evaluate the quality of embedding in a more pertinent way. During corruption, instead of replacing the entities of a triple with a random one already seen during the training set, one can replace it with an entity that has a link with the relationship involved. Following the listing 4.3: given in input a list of triples, "dataset", a list of constraints, "list_constraints" and the "corruption_side" a list of entities will be returned that will respect the corruption side expressed by the variable and the relationship of the triple must be present among one of those indicated in the list. The corruption side is important in the case of non-symmetrical relationships: let's consider a KG in which is represented a parental relationships. They are not symmetrical with respect to the relationship, and therefore corruption must also be done with criteria in order not to incur completely meaningless scores by replacing entities that are present in that type of relationship but that maintain the correct role in the triple.

Listing 4.3: Implementation of type constraint extraction

```python
def extract_type_constraint(dataset, corrupt_side,
    list_constraints=["TREAT"]):

    type_constrain = set([])

    for triple in dataset:
        if triple[1] in list_constraints:
            if corrupt_side == "s+o" or corrupt_side ==
                "s,o":
                type_constrain.add(triple[0])
                type_constrain.add(triple[2])
            elif corrupt_side == "s":
                type_constrain.add(triple[0])
            elif corrupt_side == "o":
                type_constrain.add(triple[2])

    return list(type_constrain)
```

4.6 Accuracy & Validation

Accuracy is the measure of how well our model can predict correctly. In our specific case, it is a question of having removed from the original dataset some drug-disease triples, the validation set, which therefore was not observed during the training and verifying whether the model is able to correctly reconstruct the triple. This procedure is called validation or evaluation. Validation is important in order to understand how training is going on a different dataset than the training one and to measure accuracy. This procedure is usually carried out cyclically, every tot epoch of training: in our case, it occurs approximately every 100 epoch. The frequency of this operation is dictated by the complexity and time required to carry it out. In our case, measuring accuracy metrics is not an indifferent expense in terms of time, if taken individually, in the order of tens of minutes.

The evaluation procedure follows the idea of [**evaluation**]. The idea of the paper is that of:

1. Artificially generate negative triples by corrupting first the subject and then the object.
2. Remove the positive triples.
3. Rank each test triple against all remaining triples.

In other words, the procedure consists of: Considering a triple, first corrupt the head and then the tail separately, remove the positive triple from the set of triples (Filtered solution), replace the corrupted entity with all the other entities. The entity that constitutes the positive triple, if the model always predicts perfectly, should always obtain the highest ranking compared to the other entities, which translates into a higher score of the score function. In the definition of AmpliGraph evaluation function 4.4 it is possible to see which parameters are required for the input. With "X" id indicated the list of triples on which to measure the accuracy. "model" is the trained model with the parameters that will be used to calculate the accuracy. "corrupt_side" indicates which side to bribe during the evaluation. "entities_subset" is the set of entities with which it is possible to replace the corrupt entity in the triple. "filter_unseen" is a flag that allows to check if the entities involved in the test set have been observed during the training, otherwise, it skips the triple.

Our accuracy evaluation implementation is slightly different from that proposed by [17]. Since is better to have a more precise, realistic accuracy of the model as far as drug-disease predictions are concerned and their relationship that connects them, "TREAT". Therefore the validation set will consist of triples that only admit this relation. We use the function defined by Ampligraph in a different way to calculate it: First we evaluate first only on the subject or head and then on the object or tail. Then we use the concept of Type Constraint either only of the subject or only of the object. From this list of entities deriving from the Type Constraints, following the indications of [17], randomly are choosen 500 entities. To overcome the randomness in the extraction of these entities, an average is made over several iterations of the same procedure as can be observed in listing 4.5.

Listing 4.4: Signature of evaluation function

```python
def evaluate_performance(
    X,
    model,
    filter_triples=None,
    verbose=False,
    filter_unseen=True,
    entities_subset=None,
    corrupt_side='s,o',
    use_default_protocol=False)
```

Listing 4.5: Definition of accuracy procedure

```python
hit_N = []
number_of_trials = 10

# Subject
for _ in range(number_of_trials):
    ranks = evaluate_performance(
        dataset,
        model=model,
        filter_triples=positive_filter,
        corrupt_side='s',
        entities_subset=random.sample(
            corruption_entities_subject, dim),
        filter_unseen=True,
        verbose=False)

    hit_N.append(hits_at_n_score(ranks, n=N))

# Object
for _ in range(number_of_trials):
    ranks = evaluate_performance(
        dataset,
        model=model,
        filter_triples=positive_filter,
        corrupt_side='o',
        entities_subset=random.sample(
            corruption_entities_object, dim),
        filter_unseen=True,
        verbose=False)

    hit_N.append(hits_at_n_score(ranks, n=N))

hits_50 = statistics.mean(rank_50)
```

4.7 Technical Workaround

One of the main challenges that are emerged during this project was computational resources. In the simple case of an embedding with a few thousand entities, it is necessary to work with an embedding matrix that has as many columns as there are entities and as many rows as the size of the embedding. It is not difficult to arrive at dimensions that are not easily managed in memory. The situation becomes complicated when it is necessary to carry out the evaluation. In this case, in addition to the embedding matrix, it is also necessary to have in memory the entire validation set with the rankings of each triple. In this way, the available memory runs out quickly, also due to a problem related to TF [30] which does not manage well in these situations the release of resources no longer needed. The idea to force TF to release the memory no longer needed is to divide the overall training into more chunks after at the end of each the model is saved and a new one is reinitialized starting from the previous weights but on a different process.

Algorithm 2 BaseLine

```
Require: Validation set $V = (h, r, t)$
Require: Relation R
Require: $TypeConstraint$
Require: H@N
 1: Create Type Constraint C set
 2: $C \subset E, E = \{h, t | h, t \in V\}$
 3: for $t = (h, r, t) \in V$ do
 4:    if $r == R \wedge TypeConstraint$ then
 5:       $C \leftarrow C \cup h \cup t$
 6:    end if
 7: end for
 8: if $\neg TypeConstraint$ then
 9:    $C \leftarrow E$
10: end if
11: $hit \leftarrow 0$
12: for $i <= N$ do
13:
14:    for $t = (h, r, t) \in V$ do
15:
16:       $P_{side} = rand(h, t)$
17:       if $P_{side} == h$ then
18:          $t_{corrupt} = (h_{new}, r, t), h_{new} \in C$
19:       else
20:          $t_{corrupt} = (h, r, t_{new}), t_{new} \in C$
21:       end if
22:       Compute H@N
23:       if $t == t_{corrupt}$ then
24:          $hit \leftarrow hit + 1$
25:       else
26:          break
27:       end if
28:    end for
29: end for
30: $H@N \leftarrow \frac{hit}{|V|}$
```

Experimental results 5

Qualcuno di noi
fa le cose bene,
qualcun'altro no,
ma tutti assieme
verremo giudicati
da una cosa soltanto:
il risultato.

This chapter presents the results achieved during this thesis: First, in section 5.1, is described as the characteristics of other data-sets are used in our evaluation. In section 4.3 is shown the result of the baseline. Section 5.3 describes how the metrics are used to evaluate the performance of the system. Section 5.4 explains which family of models are chosen for the future steps of evaluation. In section 5.5 is presented the hyper-parameters selection of the embedding model. Techniques of down-sampling are presented in section 5.6. In section 5.7.1 is analyzed instead ogbl-biokg. Final comments about the overall results are given in section 5.8.

5.1 Data-Sets

In this work, to evaluate the performance of the Knowledge Graph Embedding (KGE) over a biomedical Knowledge Graph (KG) to detect a possible new link between drug and disease, first is necessary to reduce the complexity of the problem cutting down same problems that are related to the model selection. In order to do so, are used a variety of non-medical data-sets, with different characteristics in terms of content, links and therefore different in terms of entities and relations. Those data-sets are commonly used, publicly available, in literature in the task of Link Prediction (LP) using KGE and have been used to compare to the state-of-the-art.

In table 5.1 are reported the details about the data-sets that are used in this project.

Name	Entities	Relations	Triples
FB15K	14,541	1,345	592,213
FB15K-237	14,541	237	310,116
WN18	40,943	18	151,442
WN18RR	40,943	11	93,003
YAGO3-10	123,182	37	1,089,040

Table 5.1: Description of the data-sets, including name, number of entities, relation and triples.

5.1.1 FB15K

The FB15K dataset was introduced in TransE work in 2013 [4]. It is a subset of Freebase which contains about 14,951 entities with 1,345 different relations. Freebase was a large collaborative knowledge base, now dismissed, consisting of data composed mainly by its community members and for this reason was constantly growing. It was an online collection of structured and maintained data harvested from many sources, including individual, user-submitted wiki contributions. Freebase aimed to create a global resource that allowed people (and machines) to access common information more effectively [2]. The project was bought by Google which made it a part on which to build the Google knowledge graph. Freebase contains around 1.2 billion triplets and more than 80 million entities. From this huge data-set, Freebase, is created FB15K, which contains the entities that are present in Wikilinks database and have at least 100 mentions in Freebase. From this subset has been removed also relationship where the head and the tail of it can be switched. This resulted in 592,213 triplets with 14,951 entities, from here the name '15k' and 1,345 relationships [4].

5.1.2 WN18

The WN18 knowledge graph is designed to produce an intuitively usable dictionary and thesaurus, and sup- port automatic text analysis. Its entities (termed synsets) correspond to word senses and relationships define lexical relations between them. This data-set is built from WordNet and presented with TransE embedding model [4], it includes the full 18 relations scraps from WordNet for roughly 41,000 synsets. WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet is also freely and publicly available for download. WordNet's structure makes it a useful tool for computational linguistics and natural language processing [31].

5.1.3 FB15K-237

FB15K-237 [1] is a subset of FB15K where are filtered out highly redundant relations. The FB15K dataset was found to suffer from major test leakage through inverse relations and a large number of test triples can be obtained simply by inverting triples in the training set [48].

5.1.4 WN18RR

Similar to FB15K, WN18 dataset was found to suffer from test leakage, for this reason, was introduced WN18RR. As a way to overcome this problem, WN18 is filtered, such that only 11 relations are left, no pair of which is reciprocal [9].

5.1.5 YAGO3-10

YAGO[2] is a knowledge base that contains both entities and relations between these entities. YAGO contains more than 50 million entities and 2 billion facts or triples. YAGO arranges its entities into classes that

[1] `https://www.microsoft.com/en-us/download/details.aspx?id=52312`
[2] `https://yago-knowledge.org`

are organized in a taxonomy. In such a way, a type of relationship can be used only between a specific couple of entities that belong to a couple of classes. The definition of these relations, together with the taxonomy is called the ontology [44]. YAGO combines two great resources:

- Wikidata is the largest general-purpose knowledge base on the Semantic Web. It is a great repository of entities, but it has a difficult taxonomy and no human-readable entity identifiers.
- Schema.org is a standard ontology of classes and relations, which is maintained by Google and others, but it does not have any entities for example.

YAGO combines these two resources, thus getting the best from both worlds: a huge repository of facts, together with an ontology that is simple and used as a standard by a large community. In addition, all identifiers in YAGO are human-readable, all entities belong to at least one class, and only classes and properties with enough instances are kept. To this, YAGO adds a system of logical constraints. These do not just keep the data clean, but also allow for reasoning on the data [29].

As of 2019, YAGO version 3 has knowledge of more than 10 million entities and contains more than 120 million facts about these entities [29].

5.2 Baseline

To understand if the embedding model was learning when applied to a biomedical KG, a baseline was created to compare the results: in section 4.3 has been presented the implementation idea behind for the baseline and now are shown the results. These results are an average over multiple execution and represent the score of $H@N$,
$N \in \{1, 2, 3, 5, 10, 20, 50, 100, 150, 200, 500, 1k, 2k, 3k\}$. The baseline measures only the accuracy of random guessing using constraint over the drug-disease relationship. The value of N stops at 3k because, in our split of the data-set between train, validation and test are present 4123 drugs and 3050 diseases. Using values of N greater than the number of different entities of drug and disease bring an insignificant accuracy of

1.0 since the random sample of entities picks up all the entities excluding the guessing factor. Table 5.2 shows the results for each value of N. Increasing the value of N, increases the accuracy because the baseline test has more chance to pick up the right entity from the set specified by the type constraint.

N	H@N Type Constraint	H@N No Type Constraint
1	0.0001	0.0
2	0.00093	0.0
3	0.00104	0.0
5	0.00166	0.0
10	0.0029	0.0001
20	0.00663	0.00021
50	0.01564	0.00083
100	0.02952	0.00155
150	0.03998	0.00249
200	0.055	0.00311
500	0.14345	0.00808
1,000	0.28721	0.01709
2,000	0.56624	0.02993
3,000	0.84578	0.0464

Table 5.2: H@N results of baseline over different values of N using type constraint and only the subject or the object of the relation are corrupted at the same time with reasonable entities (First column). H@N results of baseline over different values of N without using type constraint and regardless the side of corruption (Second column).

In table 5.2 is also reported the analysis of the baseline without using type constraint over the relation drug-disease, and regardless of the side of corruption, subject or object. The accuracy is lower than in the previous case due to the higher number of possible entities from which the random sample can pick up. In this case, there are 62,671 entities.

5.3 Accuracy cycle

Stanford proposed a simple procedure to measure the accuracy for LP, in KG as explained in section 4.6 [17]. For each triple in the test set, is corrupted only the subject or the object, but not both at the same type. The corrupted entity is substituted with another entity from a subset of according type constraint. The dimension of this subset is 500. Between this subset is computed the ranking of the entities and check if the correct one is in the first N position to get the H@N score. Since the number of entities in case of drug and disease is much more than 500, and to be more precise in the delivery of a score is better to do an average of K execution of the evaluation procedure over a different random subset of the type constraint. A higher k-value results in a corresponding preciser accuracy value as the evaluation is performed k times on different subsets, reducing the randomness and luck factor. However, a higher value takes many more runs and therefore a long time. To decide which value of k is a faithful representation, several tests were carried out on the KG presented in this work, using TransE as an embedding model and keeping the same embedded representation over all the tests after 500 epoch and comparing the results using H@10.

In table 5.3 is reported the mean values of H@10, MRR and MR over a different number of iteration K. Computing the mean value, median and the variance of these results, K=10 is chosen as a reliable parameter and a good tread off between precision and cost since the standard deviation, median and the mean value of the measurements are close to the result of K=10. In average, on a Graphics Processing Unit (GPU) GeForce GTX 960, it takes 40 seconds for each iteration.

5.4 Pre-model selection

In the chapter 3 various embedding families have been presented that can be applied to KG. Finding the most suitable embedding model is not easy: each model has its own characteristics that translate into better representing certain properties but understanding if certain characteristics are a salient aspect of the domain in question is a difficult task. For this reason, choosing a model that is very precise on a specific feature can be reductive in a general case. In addition to this problem, the embedding of one kg strongly depends on the hyper-parameters and for this reason, it is necessary to reduce the complexity of the problem.

With this idea, we use databases that have been used in the litera-

K	H@10	MRR	MR
1	0.48739	**0.24667**	32.97328
2	0.48413	0.24402	35.15942
3	0.47238	0.23889	34.94864
5	0.47706	0.24174	35.35604
10	0.48191	0.24374	35.23929
15	0.47914	**0.24667**	**32.97328**
20	**0.48739**	0.24513	35.06560
25	0.48149	0.24507	34.62349
50	0.47956	0.24370	35.25633
75	0.47874	0.24283	35.22018
100	0.48090	0.24439	34.79608
125	0.47862	0.24280	35.12275
150	0.47839	0.24321	35.20228
Average	0.48054	0.24375	34.76435
Stdev	0.00411	0.00205	35.12275
Median	0.47956	0.24374	35.22018

Table 5.3: Mean values of H@10, MRR and MR over different number of iterations K. The last three rows report the average, median and standard deviation of H@10, MRR and MR. In bold the best results.

ture, presented in section 5.1, to get an initial idea of the performance of the embedding models and then apply a selection of these to the biomedical KG.

5.4.1 Ampligraph

Ampligraph is a versatile framework written in python that provides several embedding models and data-bases ready to use. By applying these models to FB15K we obtain the results proposed in table 5.5, FB15K-237 in table 5.4, WN18 in table 5.6, WN18RR in table 5.7 and YAGO3-10 in table 5.8.

In the section 4.1.5, the biomedical KG created from different data-sets available on the internet for this project of drug repurposing was presented. By characteristics, this KG, in terms of entity, relations and number of triples, is similar to WN18RR and FB18K237. In both cases, looking to the performance table, the best performing embedding models are TransE and ComplEx, respect to H@10 score.

As for the convolution-based embedding models, such as ConvE and ConvKB, generally, in practice, they are the worst-performing ones, although they require a higher amount of computational resources than the other models.

As for HolE and DistMult, they manage to get results in line with ComplEx and TransE without major differences, but since both ComplEx and TransE are simpler and less expensive, they were chosen at this preliminary stage. In addition, these two models are two representations of different embedding model families and for this reason [40], they are a better choice than the other two.

Model	MR	MRR	H@1	H@3	H@10
TransE	208	0.31	0.22	**0.35**	**0.50**
DistMult	**199**	0.31	0.22	**0.35**	0.49
ComplEx	184	**0.32**	**0.23**	**0.35**	**0.50**
HolE	184	0.31	0.22	0.34	0.49
ConvKB	327	0.23	0.15	0.25	0.40
ConvE	1060	0.26	0.19	0.28	0.38

Table 5.4: MR, MRR, H@1, H@3 and H@10 score for FB15K237 using different emebedding model from Ampligraph framework. In bold the best results.

Model	MR	MRR	H@1	H@3	H@10
TransE	**44**	0.63	0.50	0.73	0.85
DistMult	179	0.78	0.74	**0.82**	**0.86**
ComplEx	184	**0.80**	0.76	0.82	0.86
HolE	216	**0.80**	0.76	0.83	**0.87**
ConvKB	331	0.65	0.55	0.71	0.82
ConvE	385	0.50	0.42	0.52	0.66

Table 5.5: MR, MRR, H@1, H@3 and H@10 score for FB15K using different emebedding model from Ampligraph framework. In bold the best results.

Model	MR	MRR	H@1	H@3	H@10
TransE	**260**	0.66	0.44	0.88	**0.95**
DistMult	675	0.82	0.73	0.92	**0.95**
ComplEx	726	**0.94**	**0.94**	**0.95**	**0.95**
HolE	665	**0.94**	0.93	0.94	**0.95**
ConvKB	331	0.80	0.69	0.90	0.94
ConvE	492	0.93	0.91	0.94	**0.95**

Table 5.6: MR, MRR, H@1, H@3 and H@10 score for WN18 using different emebedding model from Ampligraph framework. In bold the best results.

Model	MR	MRR	H@1	H@3	H@10
TransE	**2692**	0.22	0.03	0.37	0.54
DistMult	5531	0.47	0.43	0.48	0.53
ComplEx	4177	**0.51**	**0.46**	**0.53**	**0.58**
HolE	7028	0.47	0.44	0.48	0.53
ConvKB	3652	0.39	0.33	0.42	0.48
ConvE	5346	0.45	0.42	0.47	0.52

Table 5.7: MR, MRR, H@1, H@3 and H@10 score for WN18RR using different emebedding model from Ampligraph framework. In bold the best results.

5.4.2 Biomedical Knowledge Graph

Once the preliminary results of the models were obtained on other data-sets, their performance was measured on the biomedical data-set which will then be used for drug repurposing.

From the analysis made previously, the models that use convolution were excluded, for the remaining ones, TransE, DistMult, ComplEx and HolE, are used the hyper-parameters specified by Ampligraph for WN18NN, which is the most similar in terms of structure, to the biomedical graph proposed in this thesis. The parameters indicated by Ampligraph are optimal ones found after a tuning phase.

In table 5.9 are presented the hyper-parameters in details. The table specifies the dimension of embedding, the number of epoch for the

Model	MR	MRR	H@1	H@3	H@10
TransE	1264	**0.51**	0.41	**0.57**	**0.67**
DistMult	**1107**	0.50	0.41	0.55	0.66
ComplEx	1227	0.49	0.40	0.54	0.66
HolE	6776	0.50	**0.42**	0.56	0.65
ConvKB	2820	0.30	0.21	0.34	0.50
ConvE	6063	0.40	0.33	0.42	0.53

Table 5.8: MR, MRR, H@1, H@3 and H@10 score for YAGO3-10 using different emebedding model from Ampligraph framework. In bold the best results.

training phase, the number of the negative samples generated for each positive sample (eta), the type of loss, the type of optimizer for the score function, the learning rate (lr), the number of batches in which the train data-set should be divided and the type of regularization.

HP	TransE	DistMult	ComplEx	HolE
Dim. Emb.	350	350	200	200
Epoch	4,000	4,000	4,000	4,000
Eta	30	30	20	20
Loss	multiclass	multiclass	multiclass	multiclass
Optimizer	adam	adam	adam	adam
Lr	1e-4	1e-4	5e-4	5e-4
Batch #	150	100	10	50
Regularizer	LP	LP	LP	LP

Table 5.9: Hyperparameters that are used by Ampligraph for WN18RR for TransE, DistMult, ComplEx and Hole. The table specifies the dimension of embedding, the number of epoch and nagative sampling (eta), the type of loss and optimizer, the learning rate (lr), the batch count and the type of regularization.

In fig. 5.1 are shown the accuracy of the models in terms of H@10 score (y-axis) respect to the number of epochs (x-axis). TransE, in dark orange, has the absolute better result, then ComplEx, in blue is the second one. DistMult is the third-best score. HolE, in light orange, doesn't learn anything, for this reason, is possible that the model has a bug in the implementation of Ampligraph. The models run for a different number of epochs because the velocity of overfitting is different for model by model as shown in fig. 5.3.

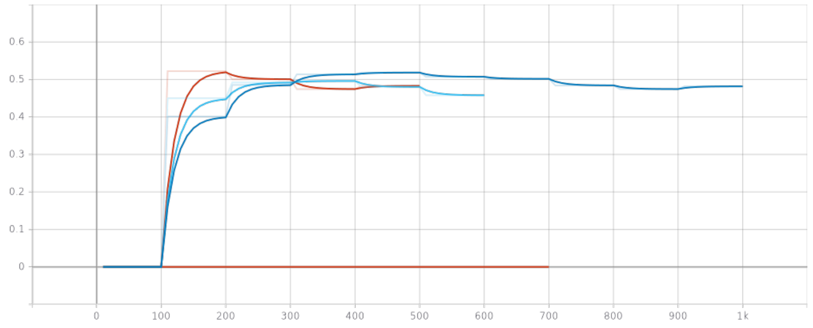


Figure 5.1: Accuracy H@10 on vertical axis, number of epochs on horizontal axis on WN18RR. TransE: Dark Orange. ComplEx: Blue. DistMult: Heavenly. HolE: Light Orange.

In fig. 5.2 is presented the train loss over the number of epoch of the models.

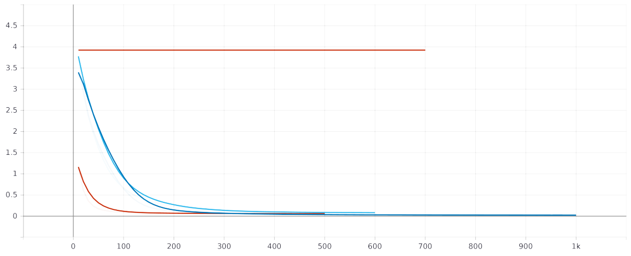


Figure 5.2: Train Loss on vertical axis, number of epochs on horizontal axis on WN18RR. TransE: Dark Orange. ComplEx: Blue. DistMult: Heavenly. HolE: Light Orange.

In fig. 5.3 is presented the validation loss over the number of the epoch of the models. The loss is computed in the same way as to train loss but on a different data-set. This data-set is not used to train the model. TransE and ComplEx are the first that start to overfitting, instead DistMult requires more epochs.

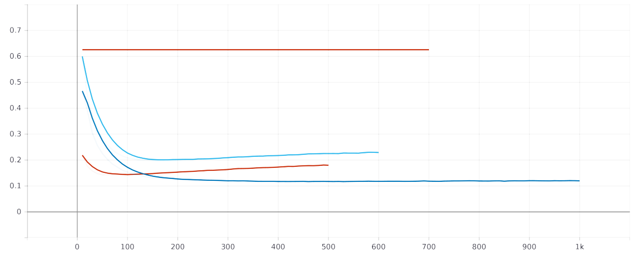


Figure 5.3: Validation Loss on vertical axis, number of epochs on horizontal axis on WN18RR. TransE: Dark Orange. ComplEx: Blue. DistMult: Heavenly. HolE: Light Orange.

From the collected data we can understand that the best models for the biomedical KG are ComplEx and TransE also in this case. They belong to two different families of models and they achieve better result respect to the other in a shorter time, i.e. less number of epochs, and so are less expensive to train.

5.5 Hyper-parameters

A substantial part of the knowledge graph embedding is choosing the parameters for training. In this section, we discuss the main parameters for the model such as the dimension of the embedding, the negative sampling, the type of optimizer, and its learning rate. Both TransE and ComplEx are tested but here for simplicity are reported just the numerical results of TransE. ComplEx's results don't differ so much from TransE and have the same trends.

5.5.1 Embedding Dimension

The embedding of a graph strongly depends on the size of the embedding. A larger embedding implies more ability to learn but too large a size is expensive and does not allow you to learn the embedding but to memorize it. To test this, TransE is trained with the default parameters presented in section 5.4.2, but with a different dimension of embedding: 10 (In orange), 50 (In grey), 100 (In green), 150 (In fuchsia), 200 (In red purple) and 500 (In heavenly). The number of epochs is fixed to 500.

In fig. 5.4 are presented the results. H@10 is used to measure the accuracy and to understand which is the best dimension of embedding for the biomedical knowledge graph built for this thesis. The figure shows that 150 (In fuchsia) as dimension is the best result and a bigger size doesn't bring any benefits but only a heavier computation. Instead, if a small dimension is used, like 10 (In orange), the model loses performances.

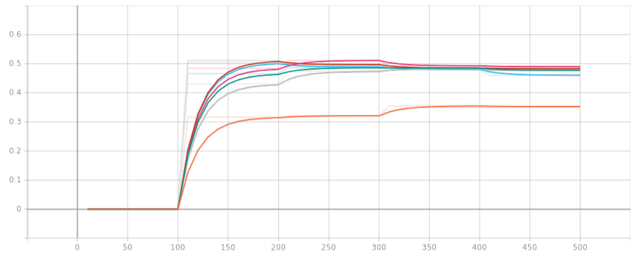


Figure 5.4: H@10 accuracy of TransE with different dimension on biomedical knowledge graph for drug repurposing: 10 (In orange), 50 (In grey), 100 (In green), 150 (In fuchsia), 200 (In red purple) and 500 (In heavenly)

In fig. 5.5 is shown the train loss over different dimension of embedding. A smaller size implies a smoother convergence.

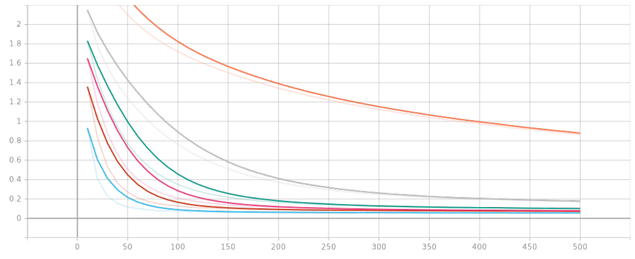


Figure 5.5: Train loss of TransE with different dimension on biomedical knowledge graph for drug repurposing: 10 (In orange), 50 (In grey), 100 (In green), 150 (In fuchsia), 200 (In red purple) and 500 (In heavenly)

In fig. 5.6 is shown the validation loss over different dimension of embedding. A bigger size implies faster overfitting reducing the number of epochs that are necessary to achieve the best absolute result.

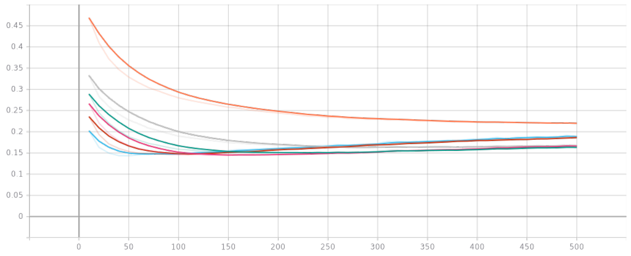


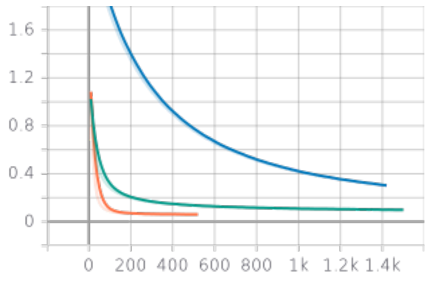
Figure 5.6: Validation loss of TransE with different dimension on biomedical knowledge graph for drug repurposing: 10 (In orange), 50 (In grey), 100 (In green), 150 (In fuchsia), 200 (In red purple) and 500 (In heavenly)

After this analysis, the dimension of 128 is chosen as a default dimension for the embedding. It guarantees good performance, fast convergence and it is less computationally expensive.

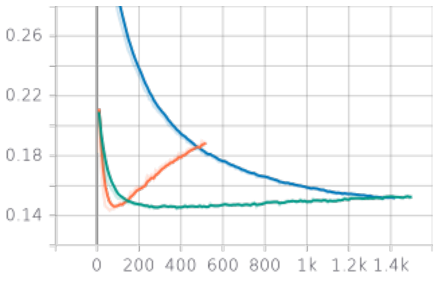
5.5.2 Optimizer

The optimizer in a machine learning algorithm is an important choice to make. It can affect the final results in terms of accuracy or the speed of convergence and therefore it is necessary to choose this parameter as correctly as possible. Ampligraph, through its implementation in Keras, provides three types of optimizers as we talked about in chapter 4: Adam, AdaGrad and SGD. In this section, we take all three methodologies and examine to establish which is the best optimization algorithm using TransE as embedding model and 128 as embedding size, applied to biomedical KG. Similar results and conclusions were obtained for ComplEx. In reality, the search for this parameter was more complex as the variables we took into consideration at the same time were two: the optimization algorithm and the learning rate. Now are compared the best results for each optimizer and then are analyzed, in the next section, the importance of the learning rate for the best optimizer found.

In fig. 5.8 are reported the H@10 scores for different optimizer: In blu SGD, in orange ADAM and in green Adagrad. The best result is achieved by ADAM around the 200th epoch. The second-best result is Adagrad followed by SGD. The best learning rate associated with ADAM is 1e-04, instead of for Adagrad is 1e-01 and for SGD is 1e-03.



(a)



(b)

Figure 5.7: (a) Training loss and (b) validation loss for TransE with dim. emb. of 128 for different optimizer on biomedical knowledge graph for drug repurposing: in blu SGD, in orange ADAM and in green AdaGrad. On x-axis the number of epoch.

In fig. 5.7a can be observed that ADAM is capable of minimizing in less epoch the training loss. SGD, instead, is the slower and its curve is smoother.

Finally, in fig. 5.7b is shown the validation loss of the model on a different data-set. Can be observed that ADAM is the first to start overfitting the data-set around the 150th epoch, instead, between the 400-600 epoch, AdaGrad starts to overfitting. Regarding SGD, it is performed an in-depth analysis: we specifically extended the test, adding thousands of epoch to see where the model overfits. SGD starts overfitting around the 3k epoch without improving significantly the H@10 score.

With this analysis can be concluded that the best optimizer is ADAM with learning rate 1e-04. The optimizer gives the model the property to learn in very few epochs without the need of long training phase.

5.5.3 Learning Rate

In this subsection are studied the effects and the importance of a correct learning rate associated with an optimizer. In this case, is used as optimizer ADAM e are sampled some learning rate: 1e-01 in fuchsia, 1e-02 in heavenly, 1e-03 in purple red and 1e-04 in orange. TransE is used as embedding model on the biomedical KG. The dimension of the embedding is 128. Similar results are obtained also for ComplEx.

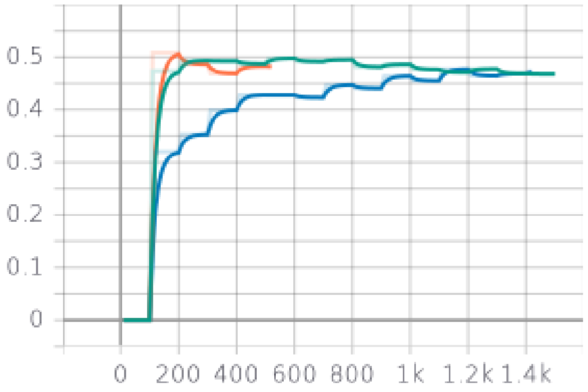


Figure 5.8: H@10 accuracy for TransE with dim. emb. of 128 for different optimizer on biomedical knowledge graph for drug repurposing: in blu SGD, in orange ADAM and in green AdaGrad. On x-axis the number of epoch.

In fig. 5.9 is presented the H@10 accuracy of the model using different learning rate for ADAM optimizer. A more aggressive learning rate brings the worst results, instead a more calm learning rate is rewarded with a better score. The waves in the plot for the fuchsia learning rate are not related to the model but are due to the restoration of the training in more phases to overcome the limits of the hardware. The number of epochs is equal for all learning rates because is related to the validation loss and its overfitting.

A higher learning rate is not capable of minimizing the training loss of the model, a lower learning rate, instead, can minimize the loss in fewer passages fig. 5.10b.

In fig. 5.10a is presented the validation loss for this test. The validation loss for 1e-01 and 1e-02 is random with up and down and without any decreasing trend. Instead, with a lower learning rate, the loss goes down and the start to overfitting. In case of the learning rate equals to 1e-03, the model start to overfitting immediately after the first epochs.

With this inspection are shown the pitfalls of using a bad learning rate and its high impact on the accuracy of the model. From now on, 1e-04 is used as learning rate with ADAM optimizer.

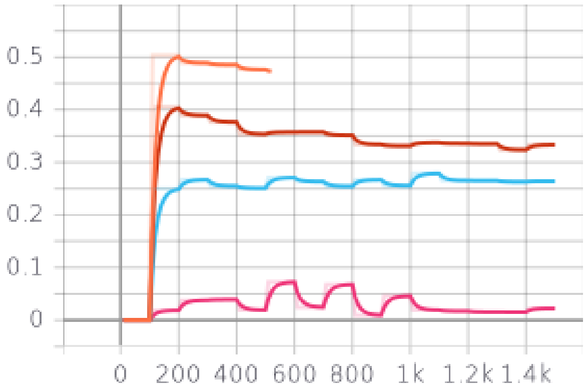
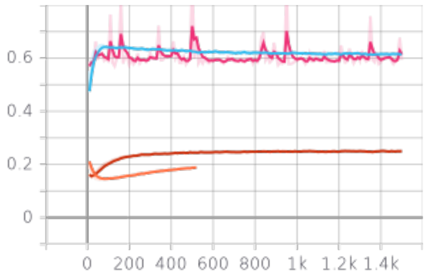
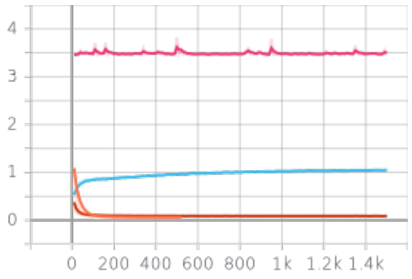


Figure 5.9: Accuracy score of H@10m regarding TransE with dim. 128 and optimizer ADAM with different learning rate on biomedical knowledge graph for drug repurposing: 1e-01 in fuchsia, 1e-02 in heavenly, 1e-03 in purple red and 1e-04 in orange. On x-axis the number of epoch.



(a)



(b)

Figure 5.10: (a) Training loss and (b) validation loss for TransE with dim. 128 and optimizer ADAM with different learning rate on biomedical knowledge graph for drug repurposing: 1e-01 in fuchsia, 1e-02 in heavenly, 1e-03 in purple red and 1e-04 in orange. On x-axis the number of epoch.

5.5.4 Negative Sampling

The negative sampling parameter that is denoted in short, eta, represent the number of negative triples that should be generated for each positive triple of the training set. In this subsection is studied the effect of different values for this parameter in case of a TransE model applied to a biomedical data-set using ADAM with learning rate 1e-04 as optimizer

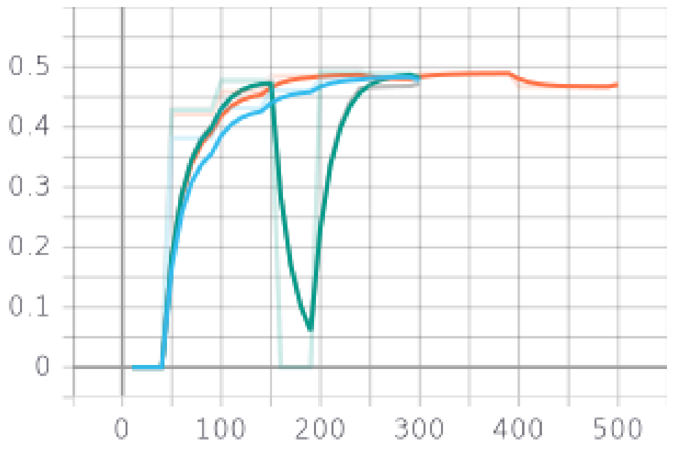


Figure 5.11: Accuracy H@10 of TransE with Dim 128, Optimizer ADAM and learning rate 1e-04 with different values for the negative sampling (eta) on biomedical knowledge graph for drug repurposing: 10 in Heavenly, 30 in orange, 60 in Green and 100 in Grey. The steep v-shape in the green plot is due to a start-stop of the training phase. The number of epoch for the orange plot is bigger to highlight the overfitting.

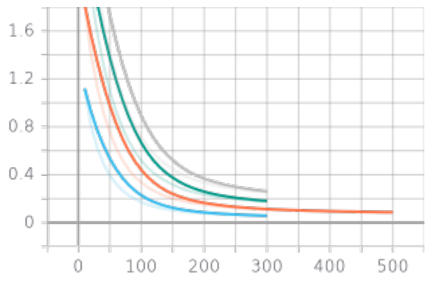
and 128 as the dimension of embedding.

In fig. 5.11 is presented the H@10 accuracy score: even using different eta values with an order of difference is not easy to observe a difference in the results. More or less the results are overlapped and a bigger value does not guarantee a better result.

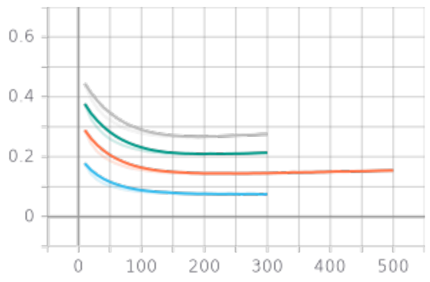
Regarding the training loss, there is the slight difference using different values for the negative sampling. A bigger value implies a slower a less steep convergence as in fig. 5.12a.

In case of the validation loss, fig. 5.12b, the results are pretty similar to the training loss. There is no steep behaviour and a bigger value start overfitting before the others. The order in that sense is respected according to the value of the negative sampling.

At the end of this analysis can be deduced that negative sampling is an important parameter but does not affect the accuracy as the other parameters analyzed before. Since there is no benefit using a bigger value of eta that implies a more expensive computation, from now on is used 30 as value for the negative sampling.



(a)



(b)

Figure 5.12: (a) Training loss and (b) validation loss for TransE with Dim 128, Optimizer ADAM and learning rate 1e-04 with different values for the negative sampling (eta) on biomedical knowledge graph for drug repurposing: 10 in Heavenly, 30 in orange, 60 in Green and 100 in Grey 100.

5.6 Size Reducing

To evaluate how much the quality of the KG created for this project impacts the accuracy in the predictions, a series of analyzes were prepared with the aim of analyzing the reduction or removal of some parts of the graph. In particular, three cases were analyzed: the first involves the training of only triple drug-disease, the second concerns the exclusion of genes from the KG and the last case analyzes the behaviour of predictions in case of the removal of drug-drug relationships.

The training parameters that were used are those that were found in the previous section to be optimal. Since the graph is different, it is not guaranteed that the parameters are optimal also in this case. However, as we have seen, the parameters that have been chosen guarantee, in most cases, an analogous behaviour at other values, reducing the computational cost. For example, the size of the embedding, unless an extreme reduction, does not significantly impact the results in terms of accuracy but only from a computational point of view. So overestimating some of these parameters can be considered a risk-free procedure from an accuracy point of view but not from a training cost point of view.

Ultimately, the same configuration was used for these analyzes. TransE with size 128, a learning rate of 1e-04 and ADAM as an optimizer, 30 as

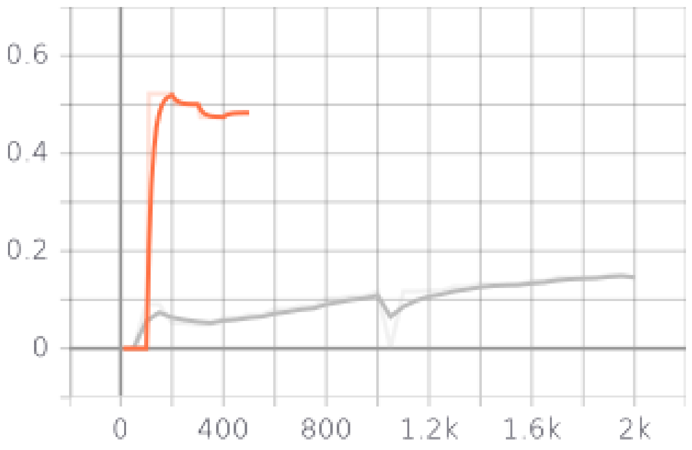


Figure 5.13: Accuracy of TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In Grey the training set is only composed by drug-disease triple, in orange the entire training-set.

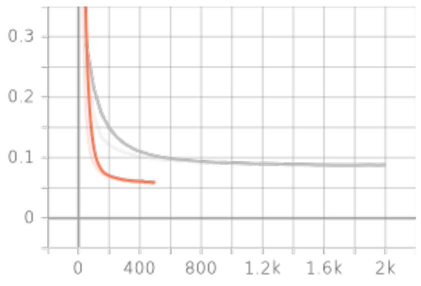
a negative sampling value, was used as the embedding model. Similar results obtained with ComplEx.

5.6.1 Drug-Disease

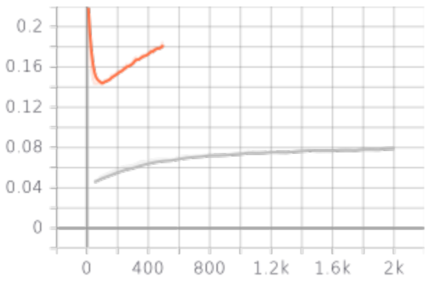
In the drug-disease reduction are considered part of the data-set, and consequently of the training and validation set, just the triples from the biomedical knowledge graph that involve drug and disease.

The goal of this experiment is to find out which is the importance of the other entities in the KG and see whether is possible to infer a new link base on just the facts that a drug can treat more disease.

In fig. 5.13 is presented the H@10 score on the validation set, of this experiment in grey, against the base KG in red. There is a significant difference in using more information to build the embedding of the domain. On the x-axis, there is a number of epochs. The difference in terms of epochs is due to the fact that the loss both on the validation set, fig. 5.14b, and on the training set, fig. 5.14a shows that the sub-model has a very slow convergence. Increasing the number of epochs in the case of the drug-disease data-set brings an improvement but it is very slow compared to the initial jump in the accuracy of the base model.



(a)



(b)

Figure 5.14: (a) Training loss and (b) validation loss for TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In Grey the training set is only composed by drug-disease triple, in orange the entire training-set.

5.6.2 No Gene

In this case, are excluded from the training set the gene entities and the triples where they are involved. In table 4.8 can be observed that genes represent around the 30% of entities involved in the KG and with these entities, are composed around the 10& of triples, table 4.9. The object of this reduction is to know whether an important part of the data-set really matters for our purpose of drug repurposing.

In fig. 5.15 is presented the H@10 score of the embedding model over the validation set against the number of training epoch. In the plot are compared the accuracy of the base data-set, in orange and the reduce KG, without gene entities in red. The difference in term of accuracy is very thin, and for this reason, means that the information provided by the genes is not so much useful. The benefit coming from this type of entity is not worth an heavier computational cost. This trend is confirmed by the validation and training loss that are very similar fig. 5.16a fig. 5.16b.

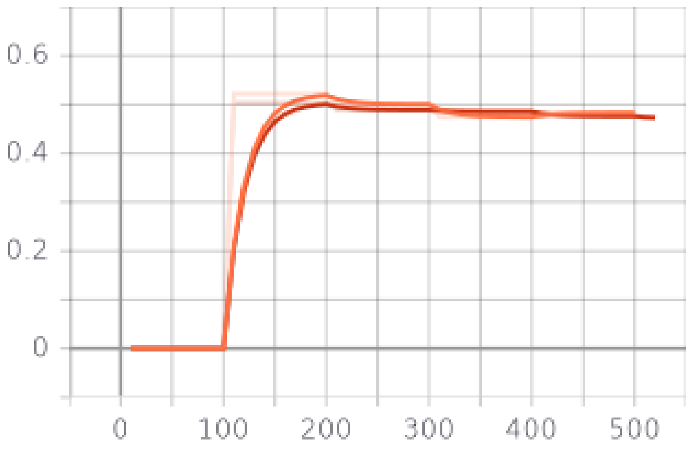


Figure 5.15: H@10 accuracy of TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In red, from the training set are excluded the gene entities and their relation, in orange the entire training-set.

5.6.3 No Parent

Another test that is conducted is to remove from the data-set the relation that connects a disease to another one. This kind of relation expresses a hierarchy between diseases: a disease can be classified based on its specification but they belong to a more generic family group. This information can be helpful because is very likely that a drug that treats a disease could bring some benefits to a similar drug that belongs to the same family.

This idea is confirmed by the results of the H@10 score on the validation set fig. 5.17. If the disease-disease relation is removed from the training set, is registered a decrease in the accuracy of 0.05 - 0.1 points. So it represents a 10% percentage of information loss. Since the computational cost of using this information is not so heavy, around 10% of the total triples is worth using it compared to the benefits.

In fig. 5.18a and fig. 5.18b are plotted the training and validation loss of this experiment. Removing the disease-disease relations implies a slower convergence and for this reason, has been extended the number of epochs in this case.

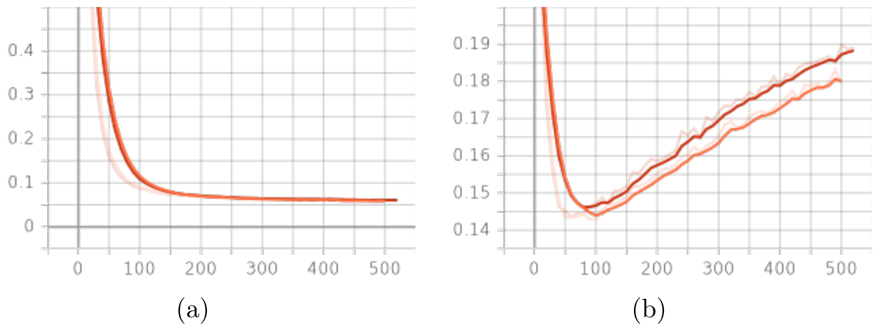


Figure 5.16: (a) Training loss and (b) validation loss for TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In red, from the training set are excluded the gene entities and their relation, in orange the entire training-set.

5.6.4 50%

In this scenario has been reduced the training set cutting of randomly the 50% of triples. Everything else is kept as in the previous cases. The goal of this study is to clarify if an increasing number of information in the KG helps to improve the accuracy of the model.

The results are encouraging. In fig. 5.19 is shown the accuracy of the model based on this sub-set of the KG. Increasing the number of information in the KG improves the significantly the accuracy of the model even if the score is not proportional to the dimension of the graph.

5.7 OGB

Hu et al. [17] measured the performance on LP of ogbl-biokg over different embedding models such as TransE, DistMult, ComplEx and RotatE. In this case, TransE is the worst model, instead ComplEx and RotatE are the better. An explanation behind this behaviour is the fact that TransE is not able to extract symmetric and asymmetric relations, in case all there results over these models are good, table 5.10. Using ogbl-biokg from Stanford university is possible to compare the quality of the biomedical KG presented in this work. To do so has been analyzed the prediction performances of the Stanford KG in the specific: every rela-

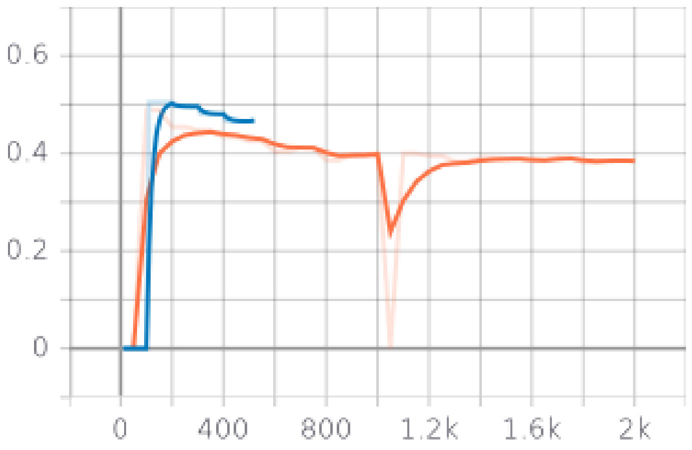


Figure 5.17: H@10 accuracy of TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In orange, from the training set are excluded the drug-drug relation, in blue the entire training-set.

tion that involves a drug is considered by itself to measure its accuracy of prediction using the corresponding type constraint. In table 5.11 is reported the results, relation by relation, excluding the relation with less than 500 occurrences except for drug-disease relation.

In the case of drug-disease relation, the accuracy that has been measured in terms of H@10 is 0.28, but only using 321 triples fig. 5.20. This table helps also to understand, based on how the KG is built, some tasks of LP are harder than others.

Model	Test MRR	Validation MRR
TransE	**0.8095**	**0.8105**
DistMult	0.8043	0.8055
ComplEx	0.7989	0.7997
RotatE	0.7452	0.7456

Table 5.10: MRR scores for Open Graph Benchmark (OGB) on validation and test set, using as embedding model Transe, DistMult, ComplEx and RotatE. Result by Hu et al. [17]. In bold the best results.

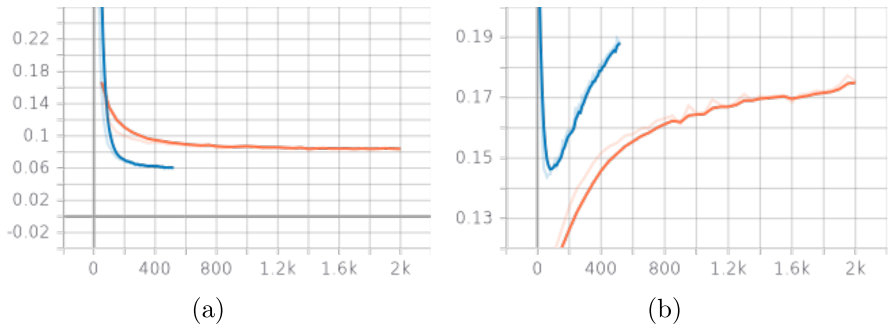


Figure 5.18: (a) Training loss and (b) validation loss for TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In orange, from the training set are excluded the drug-drug relation, in blue the entire training-set.

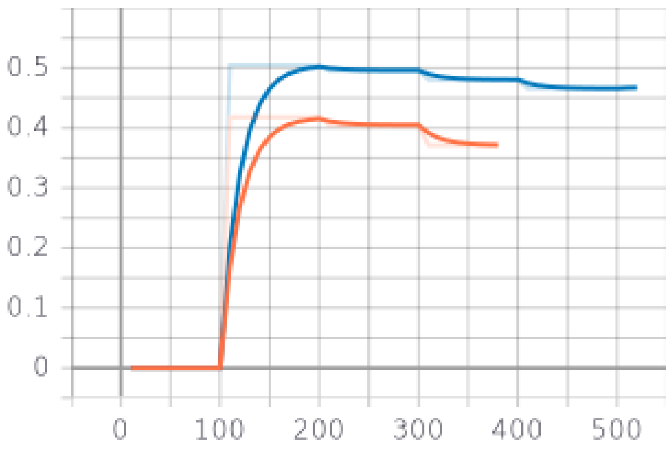


Figure 5.19: Validation loss of TransE with 128 as dimension of embedding, ADAM with 1e-04 as optimizer and 30 for the negative sampling on biomedical knowledge graph for drug repurposing. In orange, from the training set are excluded 50% of triples, in blue the entire training-set.

5.7.1 Drug-Disease Extension

Can be observed from table 3.2 and table 4.9 that the difference between the two KG in terms of triples between a drug and a disease is not negligible. For this reason, has been transferred the "TREAT" triple from the biomedical knowledge graph for this thesis to ogbl-biokg. To

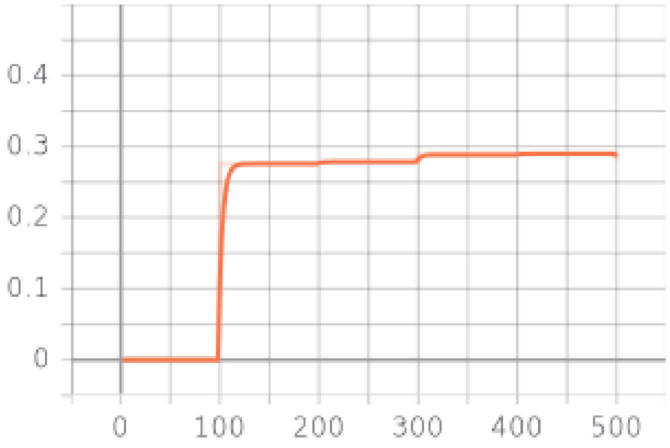


Figure 5.20: H@10 accuracy drug-disease relation of ogbl-biokg using TransE as embedding model on ogbl-biokg.

do so has been necessary a detailed work of vocabularies, trying to find a way to connect the two KG that use different IDs.

In fig. 5.21 is shown the result of this experiment: can be observed that the H@10 accuracy has significantly improved respect to the base case. That implies a better KG built around a specific goal like drug repurposing can achieve better results.

5.8 Overall Results

To sum up, in fig. 5.22 is presented the overall results of the experiment for drug repurposing applied to the biomedical knowledge graph. In our case ComplEx and TransE are the best models in terms of H@10 accuracy. The convergence of TransE is slower than ComplEx that has the best absolute result. It is very important to build an excellent KG rather than spend time on other tasks because it has a bigger impact in the performance: adding genes to the KG that are not deeply connect to the other entities of the graph does not bring better results, instead of removing an important relation such as drug-drug implies worst results. Finally, in fig. 5.22 in green, is possible to see the ogbl-biokg extended that does not perform well as the KG proposed in this thesis.

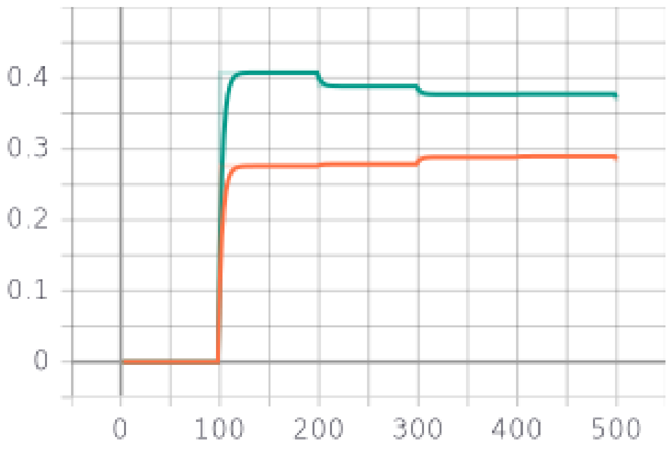


Figure 5.21: H@10 accuracy drug-disease relation of ogbl-biokg using TransE as embedding model in orange. In green the extended version, including in ogbl-biokg the "TREAT" triples from the biomedical KG of this thesis.

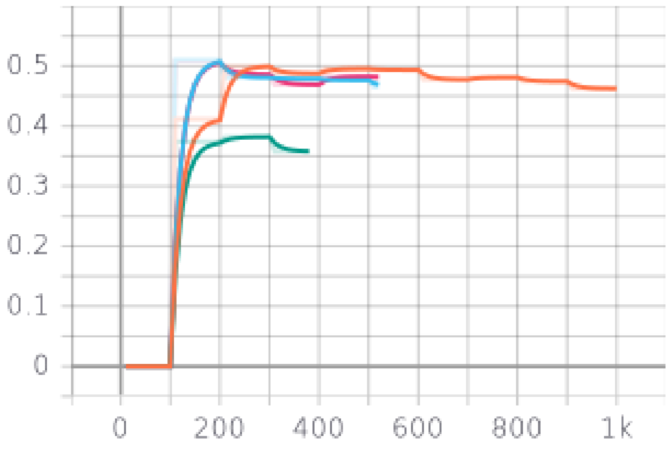


Figure 5.22: Biomedical knowledge graph embedding for drug repurposing using TransE as embedding model in orange, ComplEx in heavenly. In fuchsia, is TransE without the gene entities in the training set. In green is represented the extended version of OGB using TransE as model.

Relation	H@10	H@3	H@1	# Triples
Dg-disease	0.28	0.16	0.08	321
Dg-Dg.acquired.metabolic.disease	0.20	0.06	0.0	4,716
Dg-Dg.bacterial.infectious.disease	0.11	0.03	0.0	1,340
Dg-Dg.benign.neoplasm	0.15	0.04	0.0	2,292
Dg-Dg.cancer	0.19	0.05	0.0	3,632
Dg-Dg.cardiovascular.system.disease	0.33	0.09	0.0	6,994
Dg-Dg.cognitive.disorder	0.66	0.27	0.09	2,556
Dg-Dg.disorder.of.mental.health	0.61	0.21	0.06	1,074
Dg-Dg.endocrine.system.disease	0.85	0.41	0.16	4,131
Dg-Dg.fungal.infectious.disease	0.79	0.35	0.13	2,653
Dg-Dg.gastrointestinal.system.disease	0.96	0.60	0.25	6,248
Dg-Dg.hematopoietic.system.disease	0.94	0.54	0.21	5,962
Dg-Dg.immune.system.disease	0.74	0.31	0.10	2,512
Dg-Dg.inherited.metabolic.disorder	0.76	0.34	0.12	2,754
Dg-Dg.integumentary.system.disease	0.95	0.54	0.20	5,560
Dg-Dg.irritable.bowel.syndrome	0.51	0.20	0.07	608
Dg-Dg.musculoskeletal.system.disease	0.91	0.53	0.22	4,240
Dg-Dg.nervous.system.disease	0.96	0.57	0.23	5,888
Dg-Dg.reproductive.system.disease	0.71	0.35	0.14	1,279
Dg-Dg.respiratory.system.disease	0.94	0.53	0.20	6,031
Dg-Dg.sleep.disorder	0.72	0.29	0.09	1,967
Dg-Dg.struct.sim	0.96	0.88	0.16	962
Dg-Dg.urinary.system.disease	0.91	0.53	0.21	4,983
Dg-Dg.viral.infectious.disease	0.81	0.42	0.18	2,876

Table 5.11: H@1, H@3 and H@10 score for OGB's relations with the number of triples that use a certain relationship in the knowledge graph. Only the relations that involve a drug with more than 500 occurrences are displayed except for drug-disease relation. Dg stands for Drug.

Conclusions 6

La conclusione é
il punto
dove ti sei stufato
di pensare.

This final chapter reviews the work that has been done within this thesis and draws conclusions from what has been presented. The contributions of our work are summarized in section 6.1, while section 6.2 discusses current limitations. Finally, section 6.3 introduces future directions of research.

6.1 Contributions

The major contributions brought by this work are mainly four.

The first concerns the creation of the Knowledge Graph (KG). Starting from resources freely available online, it is possible to have enough information to be able to create a biomedical knowledge graph that is able to compete with other KG created for similar purposes. The problems that are often encountered in the creation of a biomedical KG are due to the availability of the KG as a whole due to the limitations imposed by the databases that compose it and the fact that the same entity can be identified through multiple IDs. For this reason, creating this biomedical KG was confirmed to be a feasible procedure, and creating one specifically for drug repurposing brought better results than a generic one. To do this it was necessary to create a series of vocabularies that mapped the various entities used in different databases.

The second-largest contribution of this work is related to the Knowledge Graph Embedding (KGE). There are dozens of embedding models, each specific for a specific task promising to significantly improve the results in terms of accuracy compared to the previous ones. In this, a summary of the characteristics of the main models was made by analyzing, in practice, their behaviour in a real situation and not building an ad-hoc scenario. Often, applying an embedding method that has excellent results on complete, well-constructed graphs is different than applying it to a different, incomplete domain where the link prediction task is therefore much more difficult. For this reason, simple, general models in this kind of more real situations are often better than other models that instead refer to certain characteristics of the graph.

The last major contribution of this work is related to the actual feasibility in applying machine learning techniques, such as KGE, to complete a domain or generate a new biomedical knowledge that allows, for example, to better understand the importance of some entities in the graph and of some relationships. The results in terms of accuracy for drug repurposing are very good and considering the fact that the time needed to generate them is a few hours, it allows us to imagine applying this method as initial filtering to more traditional drug repurposing techniques, reducing the complexity of the problem. Furthermore, as it has been shown, enriching the KG involves an improvement in performance and for this reason, the results are very encouraging since this approach is easily extendable by aggregating other databases.

6.2 Limits of the present work

The limitations of this work are mainly two. These limitations are posterior observations and no shortcomings with respect to the state of the art at the start of the project since it was a pioneering research project in its approach and domain.

The first limitation is due to the impossibility of really checking the quality of the predictions. The accuracy results were obtained by removing true triples from the training set and verifying if the embedding model is able to predict them correctly. However, there is no way to verify if any new predictions are at least viable.

The second limitation is due to the use of embedding models that are not specific for these applications or for this domain which could instead bring benefits in the accuracy of the predictions.

6.3 Future work

In the future, there is the intention to further improve this project. This research was an exploratory project in all its phases: from the creation of the knowledge graph, through the use of embedding to the predictions for drug repurposing. For this reason, in the coming months, there is a desire to improve this project in all three of its main aspects.

First, it will be necessary to improve the KG that represents our starting domain. Some ideas to do this can concern the integration of other entities from other databases such as gene ontology, side effects, etc.. Besides, it would be good to apply the filtering techniques of the KG before using it in the following steps. It may be that some parts of KG or some entities, for some reason, stand on their own, and not be connected in any way to the rest of the entities. These situations do not bring additional information but only computational heaviness. This idea has already been used with databases like WN [31] and FB [48] where there have been versions with elements that have at least thousands of links.

The second step to improve this project is to develop, on the basis of the previous models, an ad-hoc model that is able to learn better the biomedical connections. An example above all could be to introduce weighted links in entities since some triples are not manually curated but are also inferred by other links.

Finally, another step to improve this research is to introduce a method to evaluate the quality of the new predictions. To do this we could take up an idea already discussed previously: finding links, and therefore studies, between the drug and the disease in the medical-scientific literature such as PubMed. Another valid alternative could be to involve industry experts in order to get feedback on the best predictions.

6.4 Final considerations

In conclusion, this thesis illustrated the challenges of creating a good system able to help in finding a treatment for a disease.

The new knowledge graph that has been built for this project represents an extensible tool to represent a biomedical domain that provides better prediction respect to other biomedical databases if measured on the task of drug repurposing.

In general, the combination of a knowledge graph and a graph embedding model can represent a different approach to solve a generic problem on which there is a complex amount of data to analyze and is necessary to infer a new link between the entities in the KG.

Still, there exist many improvements that could be made, either as extensions to the domain or as improvements to the core machine learning part, as presented in section 6.3. As of today, the field of drug repurposing is under the highlight due to the promise of reducing cost and time in drug development. However, no single approach managed to become the de facto standard in the field. Many challenges are still open, and in the following years, we will most certainly see great advancements in computational techniques for drug repurposing.

Bibliography

[1] Ivana Balazevic, Carl Allen, and Timothy Hospedales. «TuckER: Tensor Factorization for Knowledge Graph Completion». In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (2019). DOI: `10.18653/v1/d19-1522`. URL: `http://dx.doi.org/10.18653/v1/D19-1522`.

[2] Kurt Bollacker et al. «Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge». In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. SIGMOD '08. Vancouver, Canada: Association for Computing Machinery, 2008, 1247â1250. ISBN: 9781605581026. DOI: `10.1145/1376616.1376746`. URL: `https://doi.org/10.1145/1376616.1376746`.

[3] Antoine Bordes et al. «A semantic matching energy function for learning with multi-relational data». In: *Machine Learning* 94.2 (May 2013), 233â259. ISSN: 1573-0565. DOI: `10.1007/s10994-013-5363-6`. URL: `http://dx.doi.org/10.1007/s10994-013-5363-6`.

[4] Antoine Bordes et al. «Translating Embeddings for Modeling Multi-Relational Data». In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPSâ13. Lake Tahoe, Nevada: Curran Associates Inc., 2013.

[5] Gene Ontology Consortium. «The Gene Ontology (GO) database and informatics resource». In: *Nucleic Acids Research* 32.suppl$_1$ (Jan. 2004), pp. D258–D261. ISSN: 0305-1048. DOI: `10.1093/nar/gkh036`. eprint: `https://academic.oup.com/nar/article-pdf/32/suppl\_1/D258/7621365/gkh036.pdf`. URL: `https://doi.org/10.1093/nar/gkh036`.

[6] The UniProt Consortium. «UniProt: a worldwide hub of protein knowledge». In: *Nucleic Acids Research* 47.D1 (Nov. 2018), pp. D506–D515. ISSN: 0305-1048. DOI: 10.1093/nar/gky1049. eprint: https://academic.oup.com/nar/article-pdf/47/D1/D506/27437297/gky1049.pdf. URL: https://doi.org/10.1093/nar/gky1049.

[7] Luca Costabello et al. *AmpliGraph: a Library for Representation Learning on Knowledge Graphs.* Mar. 2019. DOI: 10.5281/zenodo.2595043. URL: https://doi.org/10.5281/zenodo.2595043.

[8] Allan Peter Davis et al. «The Comparative Toxicogenomics Database: update 2019». In: *Nucleic acids research* 47 (Sept. 2018). DOI: 10.1093/nar/gky868.

[9] Tim Dettmers et al. *Convolutional 2D Knowledge Graph Embeddings.* 2017. arXiv: 1707.01476 [cs.LG].

[10] Daniel A. Dias, Sylvia Urban, and Ute Roessner. «A Historical Overview of Natural Products in Drug Discovery». In: *Metabolites* 2.2 (Apr. 2012), 303â336. ISSN: 2218-1989. DOI: 10.3390/metabo2020303. URL: http://dx.doi.org/10.3390/metabo2020303.

[11] Boyang Ding et al. «Improving Knowledge Graph Embedding Using Simple Constraints». In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2018). DOI: 10.18653/v1/p18-1011. URL: http://dx.doi.org/10.18653/v1/P18-1011.

[12] John Duchi, Elad Hazan, and Yoram Singer. «Adaptive Subgradient Methods for Online Learning and Stochastic Optimization». In: *Journal of Machine Learning Research* 12 (July 2011), pp. 2121–2159.

[13] Takuma Ebisu and Ryutaro Ichise. *TorusE: Knowledge Graph Embedding on a Lie Group.* 2017. arXiv: 1711.05435 [cs.AI].

[14] Lingbing Guo et al. *Recurrent Skipping Networks for Entity Alignment.* 2018. arXiv: 1811.02318 [cs.CL].

[15] Ada Hamosh et al. «Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders». In: *Nucleic Acids Research* 33.suppl$_1$ (Jan. 2005), pp. D514–D517. ISSN: 0305-1048. DOI: 10.1093/nar/gki033. eprint: https://academic.oup.com/nar/article-pdf/33/suppl_1/D514/7621651/gki033.pdf. URL: https://doi.org/10.1093/nar/gki033.

[16] Xu Han et al. «OpenKE: An Open Toolkit for Knowledge Embedding». In: *Proceedings of EMNLP*. 2018, pp. 139–144.

[17] Weihua Hu et al. *Open Graph Benchmark: Datasets for Machine Learning on Graphs*. 2020. arXiv: 2005.00687 [cs.LG].

[18] Francesco Iorio et al. «Identification of small molecules enhancing autophagic function from drug network analysis». In: *Autophagy* 6 (Nov. 2010), pp. 1204–5. DOI: 10.4161/auto.6.8.13551.

[19] Francesco Iorio et al. «Transcriptional data: A new gateway to drug repositioning?» In: *Drug discovery today* 10.1016/j.drudis.2012.07.014 (Aug. 2012). DOI: 10.1016/j.drudis.2012.07.014.

[20] Guoliang Ji et al. «Knowledge Graph Embedding via Dynamic Mapping Matrix». In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, July 2015, pp. 687–696. DOI: 10.3115/v1/P15-1067. URL: https://www.aclweb.org/anthology/P15-1067.

[21] Shaoxiong Ji et al. «A Survey on Knowledge Graphs: Representation, Acquisition and Applications». In: (Feb. 2020).

[22] Xiaotian Jiang, Quan Wang, and Bin Wang. «Adaptive Convolution for Multi-Relational Learning». In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 978–987. DOI: 10.18653/v1/N19-1103. URL: https://www.aclweb.org/anthology/N19-1103.

[23] Minoru Kanehisa et al. «KEGG as a reference resource for gene and protein annotation». In: *Nucleic Acids Research* 44.D1 (Oct. 2015), pp. D457–D462. ISSN: 0305-1048. DOI: 10.1093/nar/gkv1070. eprint: https://academic.oup.com/nar/article-pdf/44/D1/D457/9482226/gkv1070.pdf. URL: https://doi.org/10.1093/nar/gkv1070.

[24] Seyed Mehran Kazemi and David Poole. *SimplE Embedding for Link Prediction in Knowledge Graphs*. 2018. arXiv: 1802.04868 [stat.ML].

[25] Michael Keiser et al. «Predicting new molecular targets for known drugs». In: *Nature* 462 (Nov. 2009), pp. 175–81. DOI: 10.1038/nature08506.

[26] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization.* 2014. arXiv: 1412.6980 [cs.LG].

[27] Hailun Lin et al. «Learning Entity and Relation Embeddings for Knowledge Resolution». In: *Procedia Computer Science* 108 (2017). International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland, pp. 345–354. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.05.045. URL: http://www.sciencedirect.com/science/article/pii/S1877050917305628.

[28] Hanxiao Liu, Yuexin Wu, and Yiming Yang. *Analogical Inference for Multi-Relational Embeddings.* 2017. arXiv: 1705.02426 [cs.LG].

[29] F. Mahdisoltani, J. Biega, and Fabian M. Suchanek. «YAGO3: A Knowledge Base from Multilingual Wikipedias». In: *CIDR*. 2015.

[30] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[31] George A. Miller. «WordNet: A Lexical Database for English». In: *Commun. ACM* 38.11 (Nov. 1995), pp. 39–41. ISSN: 0001-0782. DOI: 10.1145/219717.219748. URL: https://doi.org/10.1145/219717.219748.

[32] Sameh K. Mohamed, Aayah Nounu, and Vít Nováček. «Drug Target Discovery Using Knowledge Graph Embeddings». In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing.* SAC â19. Limassol, Cyprus: Association for Computing Machinery, 2019, 11â18. ISBN: 9781450359337. DOI: 10.1145/3297280.3297282. URL: https://doi.org/10.1145/3297280.3297282.

[33] Sameh K Mohamed, Vit Novacek, and Aayah Nounu. «Discovering protein drug targets using knowledge graph embeddings». In: *Bioinformatics* 36.2 (Aug. 2019), pp. 603–610. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btz600. eprint: https://academic.oup.com/bioinformatics/article-pdf/36/2/603/31962922/btz600.pdf. URL: https://doi.org/10.1093/bioinformatics/btz600.

[34] David Nadeau and Satoshi Sekine. «A Survey of Named Entity Recognition and Classification». In: *Lingvisticae Investigationes* 30 (Jan. 2007). DOI: 10.1075/li.30.1.03nad.

[35] Dai Quoc Nguyen et al. «A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization». In: *Proceedings of the 2019 Conference of the North* (2019). DOI: 10.18653/v1/n19-1226. URL: http://dx.doi.org/10.18653/v1/N19-1226.

[36] Dai Quoc Nguyen et al. «A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network». In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)* (2018). DOI: 10.18653/v1/n18-2053. URL: http://dx.doi.org/10.18653/v1/N18-2053.

[37] Dat Quoc Nguyen et al. «STransE: a novel embedding model of entities and relationships in knowledge bases». In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2016). DOI: 10.18653/v1/n16-1054. URL: http://dx.doi.org/10.18653/v1/N16-1054.

[38] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. *Holographic Embeddings of Knowledge Graphs.* 2015. arXiv: 1510.04935 [cs.AI].

[39] Sudeep Pushpakom et al. «Drug repurposing: Progress, challenges and recommendations». In: *Nature Reviews Drug Discovery* 18 (Oct. 2018). DOI: 10.1038/nrd.2018.168.

[40] Andrea Rossi et al. *Knowledge Graph Embedding for Link Prediction: A Comparative Analysis.* 2020. arXiv: 2002.00819 [cs.LG].

[41] Sebastian Ruder. *An overview of gradient descent optimization algorithms.* 2016. arXiv: 1609.04747 [cs.LG].

[42] Sandeep Sinha and Divya Vohora. «Chapter 2 - Drug Discovery and Development: An Overview». In: *Pharmaceutical Medicine and Translational Clinical Research.* Ed. by Divya Vohora and Gursharan Singh. Boston: Academic Press, 2018, pp. 19–32. ISBN: 978-0-12-802103-3. DOI: https://doi.org/10.1016/B978-0-

12-802103-3.00002-X. URL: http://www.sciencedirect.com/science/article/pii/B978012802103300002X.

[43] Daniel Sosa et al. «A Literature-Based Knowledge Graph Embedding Method for Identifying Drug Repurposing Opportunities in Rare Diseases». In: (Aug. 2019). DOI: 10.1101/727925.

[44] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. «Yago: A Core of Semantic Knowledge». In: *Proceedings of the 16th International Conference on World Wide Web*. WWW '07. Banff, Alberta, Canada: Association for Computing Machinery, 2007, 697â706. ISBN: 9781595936547. DOI: 10.1145/1242572.1242667. URL: https://doi.org/10.1145/1242572.1242667.

[45] Zhiqing Sun et al. *RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space*. 2019. arXiv: 1902.10197 [cs.LG].

[46] Pedro Tabacof and Luca Costabello. *Probability Calibration for Knowledge Graph Embedding Models*. 2019. arXiv: 1912.10000 [cs.AI].

[47] Alan Talevi and Carolina L. Bellera. «Challenges and opportunities with drug repurposing: finding strategies to find alternative uses of therapeutics». In: *Expert Opinion on Drug Discovery* 15.4 (2020). PMID: 31847616, pp. 397–401. DOI: 10.1080/17460441.2020.1704729. URL: https://doi.org/10.1080/17460441.2020.1704729.

[48] Kristina Toutanova et al. «Representing Text for Joint Embedding of Text and Knowledge Bases». In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 1499–1509. DOI: 10.18653/v1/D15-1174. URL: https://www.aclweb.org/anthology/D15-1174.

[49] Théo Trouillon et al. *Complex Embeddings for Simple Link Prediction*. 2016. arXiv: 1606.06357 [cs.AI].

[50] D. S. Wishart et al. «DrugBank 5.0: a major update to the DrugBank database for 2018». In: *Nucleic Acids Res.* 46.D1 (Jan. 2018), pp. D1074–D1082.

[51] Yanrong Wu and Zhichun Wang. «Knowledge Graph Embedding with Numeric Attributes of Entities». In: *Proceedings of The Third Workshop on Representation Learning for NLP*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 132–136. DOI: 10.18653/v1/W18-3017. URL: https://www.aclweb.org/anthology/W18-3017.

[52] Hanqing Xue et al. «Review of Drug Repositioning Approaches and Resources». In: *International journal of biological sciences* 14 (July 2018), pp. 1232–1244. DOI: 10.7150/ijbs.24612.

[53] Bishan Yang et al. *Embedding Entities and Relations for Learning and Inference in Knowledge Bases*. 2014. arXiv: 1412.6575 [cs.CL].

[54] Wen Zhang et al. «Interaction Embeddings for Prediction and Explanation in Knowledge Graphs». In: *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (Jan. 2019). DOI: 10.1145/3289600.3291014. URL: http://dx.doi.org/10.1145/3289600.3291014.

[55] Y. Zhang et al. «NSCaching: Simple and Efficient Negative Sampling for Knowledge Graph Embedding». In: *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. 2019, pp. 614–625.

[56] Yongjun Zhu et al. «Knowledge-driven drug repurposing using a comprehensive drug knowledge graph». In: *Health Informatics Journal* 0.0 (0). PMID: 32674665, p. 1460458220937101. DOI: 10.1177/1460458220937101. eprint: https://doi.org/10.1177/1460458220937101. URL: https://doi.org/10.1177/1460458220937101.