AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

KATEDRA INFORMATYKI

PRACA DYPLOMOWA MAGISTERSKA

Link prediction based on classical techniques and dual convolutional neural network

Predykcja połączeń w grafie w oparciu o metody klasyczne oraz grafowe sieci konwolucyjne

Autor: ***Grzegorz Borkowski***
Kierunek studiów: ***Informatyka***
Typ studiów: ***Stacjonarne***
Opiekun pracy: ***prof. dr hab. inż. Witold Dzwinel***

Kraków, 2019

Oświadczenie studenta

Uprzedzony(-a) o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2018 r. poz. 1191 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony(-a) o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta.”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Jednocześnie Uczelnia informuje, że zgodnie z art. 15a ww. ustawy o prawie autorskim i prawach pokrewnych Uczelni przysługuje pierwszeństwo w opublikowaniu pracy dyplomowej studenta. Jeżeli Uczelnia nie opublikowała pracy dyplomowej w terminie 6 miesięcy od dnia jej obrony, autor może ją opublikować, chyba że praca jest częścią utworu zbiorowego. Ponadto Uczelnia jako podmiot, o którym mowa w art. 7 ust. 1 pkt 1 ustawy z dnia 20 lipca 2018 r. – Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.), może korzystać bez wynagrodzenia i bez konieczności uzyskania zgody autora z utworu stworzonego przez studenta w wyniku wykonywania obowiązków związanych z odbywaniem studiów, udostępniać utwór ministrowi właściwemu do spraw szkolnictwa wyższego i nauki oraz korzystać z utworów znajdujących się w prowadzonych przez niego bazach danych, w celu sprawdzania z wykorzystaniem systemu antyplagiatowego. Minister właściwy do spraw szkolnictwa wyższego i nauki może korzystać z prac dyplomowych znajdujących się w prowadzonych przez niego bazach danych w zakresie niezbędnym do zapewnienia prawidłowego utrzymania i rozwoju tych baz oraz współpracujących z nimi systemów informatycznych.

……

(czytelny podpis studenta)

Acknowledgment

I would like to express my sincere gratitude to my supervisor, Prof dr hab. inż. Witold Dzwinel for the continuous support during my study and research, encouragement, patient guidance and inspiration to explore the fascinating world of machine learning.

Abstract

Graphs are data representations used for modelling a variety of problems in many disciplines. One of the key challenges is finding a proper graph structure that not only will model real world accurately, but also work seamlessly (with no or little modification) with existing graph algorithms. This thesis presents the work on improving link prediction algorithms by leveraging a particular graph structure or some additional graph attributes.

Three different approaches are discussed. The first part addresses a link prediction on a graph of graph structure consisting of an external and internal graphs layers using dual convolutional neural network. The effectiveness of the model is evaluated on the dataset of interactions between drugs and the scientists citation dataset.

The next approach is focused on enhancing existing link prediction with textual node attributes using paragraph embedding methods. The experiment demonstrate that using tf-idf improves the node2vec on the scientists citaiton dataset.

The last part presents work on using dimensonality reduction algorithms to improve the link prediction precision by up to two percentage point predicting links between articles on Wikipedia, and finding scientists who are likely to collaborate in the future.

Streszczenie

Grafy służą do modelowania wielu rzeczywistych problemów. Istotnym wyzwaniem jest znalezienie odpowiedniej struktury grafu dobrze modelującej dany problem, tak aby równocześnie można było zastosować lub łatwo zmodyfikować istniejące algorytmy działające na grafach. W tej pracy przedstawiam różne podejścia do predykcji połączeń w grafach, wykorzystujących ich niestandardową strukturę lub pewne dodatkowe informacje o nich.

Praca ta składa się z trzech różnych części. Pierwsza z nich korzysta ze struktury grafu grafów składającej się z zewnętrznego i wewnętrznego grafu, badając predykcję linków w oparciu o model konwolucyjnej sieci neuronowej zaprojektowanej dla tej struktury grafów. Skuteczność tej metody jest ewaluowana na zbiorze zawierającym informacje o toksyczności reakcji między lekami oraz na zbiorze cytowań prac naukowych.

Kolejna część bada wykorzystanie atrybutów tekstowych w celu poprawy predykcji linków. Badane teksty to abstrakty artykułów naukowych ze zbioru cytowań prac naukowych. Eksperyment pokazuje, że odpowiednie zastosowanie indeksu tf-idf polepsza wyniki predykcji linków opartych o model node2vec.

Ostatnie eksperymenty badają wpływ zastosowania redukcji wymiarów na skuteczność metod predykcji linków. W pracy pokazano (na problemie predykcji czy artykuły w Wikipedii zawierają wzajemne odniesienia oraz na zbiorze cytowań artykułów naukowych), że zastosowanie metody PCA może podnieść precyzję klasyfikatora o dwa punkty procentowe.

Contents

List of symbols

General

$\mathbb{R}$	Real numbers
$\mathbb{R}^d$	d-dimensional Euclidean space

Matrices and vectors

A	Matrix A
$A_{m\times n} = [a_{ij}]$	Matrix A composed of elements a_{ij}, where $1 \leq i \leq m$ is the row number and $1 \leq j \leq n$ is the column number
$\mathbf{v}$	Row vector v
$\mathbf{v^T}$	Column vector v

Sets

$\mathbb{S}$	Set $\mathbb{S}$
$\mathbb{S} = \{a^{(1)}, a^{(2)}, ..., a^{(N)}\}$	Set $\mathbb{S}$ composed of N elements $a^{(1)}, a^{(2)}, ..., a^{(N)}$
$\forall$	universal quantifier
$\exists$	existential quantifier
$\cup$	set union
$\cap$	set intersection
$\lvert\mathbb{S}\rvert$	number of elements in $\mathbb{S}$

Graphs

$G = (\mathbb{V}, A)$	Graph, where $\mathbb{V}$ is a set of vertices and A is an adjacency matrix
$\mathfrak{G} = (\mathbb{V}_G, A_G)$	Graph of graphs $\mathfrak{G}$, where $\mathbb{V}_G$ is a set of vertices, where $\forall v_i \in \mathbb{V}_G$ is a graph G_i. A_G is an adjacency matrix.
$\Gamma(v)$	a set of all neighbours of vertex v

1. Introduction

Graphs are data representations used for modelling of a variety of real-world problems in many disciplines. Graphs are thoroughly studied in the field named of network theory [Newman, 2010]. One of its subfields of network theory is social network analysis, focused on studying the relationship between entities in graphs. One interesting topic in social network analysis is link prediction problem trying to answer questions: How does the connection patterns change over time? How likely is a connection between two particular nodes occurs in the future? This topic has been excessively studied, leading to promising outcomes in fields of bioinformatics [Lei and Ruan, 2013], e-commerce [Li et al., 2014a], security [Desmarais and Cranmer, 2013], and many more.

Graphs have many properties such as the size, degree distribution, graph structure, and many more. Link prediction research focuses on these properties, as the efficiency of the algorithms depends on leveraging them [Murata and Moriyasu, 2008]. There is no algorithm performing best for all kinds of graphs [Gao et al., 2015].

One type of graphs with a particular structure is a graph of graphs [Harada et al., 2018]. A graph of graphs is a multilevel graph made of an external and internal layers of nodes. Each node in external layer is a graph in an internal layer, and nodes in a particular internal layers are not interconnected.

Nevertheless, the link prediction on this kind of graph is a novel research topic, and there is a room for new ideas, implementations, evaluations, and benchmarks, leading to possible significant outcomes [Wang et al., 2014].

A link prediction on graph of this kind has many promising applications. It may be used in finding unknown interaction (e.g toxicity) between drugs and proteins, seeking similarity relations between group of researchers, and their papers, finding potentially valuable flights destinations, or to analyse the connection between communities in social networks.

Yet another interesting type of graphs are graphs of scientists collaboration [Tang et al., 2008]. Nodes of these graphs are scientific publications, whereas edges represent a citation relationship between papers. This graph has interesting structure, because much information about nodes is represented by its attribute - publication abstract. The challenge is how to extract and incorporate the publication abstract into the process of link prediction.

Link prediction problem using state-of-the-art methods relies on the quality of finding precise representations of graphs using graph embeddings, as the graph embeddings layer outputs are inputs to the link prediction layer.

Finding more accurate representations of graphs may help in other graph related problems (e.g graph visualisation) by increasing the efficiency of machine learning tasks (e.g dimensionality reduction) [Goyal and Ferrara, 2017, Cai et al., 2018].

One of the recent developments is the design of a dual convolutional neural network architecture [Harada et al., 2018] extracting the features from both internal and external graphs in an end-to-end matter, suited for solving a link prediction problem in graphs of graphs, showing promising results.

This thesis presents a work on evaluating classic link prediction methods to link prediction on a graph of graphs, a comparison with dual convolution neural network results, and its improvements on real-world datasets. The second part of this thesis is focused on link prediction in scientists collaboration network, and attempts to incorporate information about the papers into link prediction task. The main goal of this thesis is answering the following question: How to improve link prediction algorithms either leveraging a particular graph structure, or some additional graph attributes.

Problem formulation

This section presents a formal definitions of a link prediction problem described in this thesis.

Definition 1.0.1. *Graph:* A directed graph G is an ordered pair $G = (V, E)$, where V is a set of nodes, and E is a set of ordered pairs called edges. Undirected graph is a graph, where for every pair $(x, y) \in E$, also pair $(y, x) \in E$. Each node v_i contains a finited set of attributes $\{A_i^0, A_i^1, ...A_i^n\}$.

Definition 1.0.2. *Graph of graphs* [Harada et al., 2018]: A graph of graph $G = (V, E)$ is a undirected graph such as G is called an *external graph*, and each node $v_i \in V$ has an attribute A_i^0, *internal graph* $\mathbb{G}_i$. Each internal graph is an undirected graph made from predefined types of nodes. There is no connection between nodes in internal graphs.

Definition 1.0.3. *Link prediction*: Given a graph $G = (V, E)$, the set of edges E is divided into two disjoint sets: $E_{train\_positive}, E_{test\_positive}$. From sets of all pairs of nodes $(v_i, v_j) \in V^2, v_i \neq v_j \wedge (v_i, v_j) \notin E$, a subsets of pairs is chosen and divided into two disjoint sets $E_{train\_negative}, E_{test\_negative}$. Then $E_{test} = E_{test\_positive} \bigcup E_{test\_negative}$, and
$E_{train} = E_{train\_positive} \bigcup E_{train\_negative}$,
The link prediction problem is defined as: given $(V, E_{train\_positive}, E_{train\_negative})$ for each element $e_{test} \in E_{test}$ decide if $e_{test} \in E_{test\_positive}$ or $e_{test} \in E_{test\_negative}$

Definition 1.0.4. *Precision* [Goyal and Ferrara, 2017] is defined as the fraction of correct predictions in top k predictions, defined by a formula: $Pr@k = \frac{|E_{pred(1:k)} \bigcap E_{test\_positive}|}{k}$

Definition 1.0.5. *Mean Average Precision* [Goyal and Ferrara, 2017] estimates the precision for all nodes and averages the score as in formula.

Research hypotheses

The hypotheses are stated as follows:

1. *On drugs prediction dataset, if graph is defined as a graph of graphs, the dual convolutional neural network achieves higher precision than other methods*
2. *Enhancing existing link prediction model with paragraph embedding of abstracts of scientific publication may improve the precision of link prediction*
3. *Controlling the dimensonality of embedding can increase the link prediction accuracy*

Contributions

The main contributions of this thesis are:

- Preparing new datasets used for benchmarks
- The development, implementation and evaluation of a new algorithm for link prediction on collaboration network
- The implementation and evaluation of a link prediction model based on dual convolutional neural network on graph of graphs
- The development, implementation and evaluation of a new algorithm for link prediction combining dimensionality reduction technique
- Executing benchmarks comparing the quality of different link prediction models

Structure of the thesis

My thesis is structured as follows. The first chapter is an introduction. Second chapter is an overview of recent developments in link prediction domain. The chapter provides a background to algorithms, and models used later. The following one explains the implementation, and details of different solutions. The analysis of the results is discussed next. The last chapter contains conclusions, and a proposal of possible areas for future development.

The thesis main topic is about improving the accuracy of link prediction classifiers. However, it consists of three different distinct parts, each part representing different approach to tackle the main goal of the thesis. The first approach is using dual convolutional neural network model for the link prediction. The next approach is enhancing the link prediction with paragraph emebedding techniques. The last approach is using dimensionality reduction to improve the classifiers.

2. State of the art

The main aim of this Chapter is to present the state of the art of the link prediction problem and different approaches to tackle the link prediction problem, starting from the simplest solutions, and gradually introducing more complex ones.

Firstly, unsupervised topology-based metrics are described, followed by standard supervised learning approaches and graph-embedding methods, leading to the state-of-the-art deep-learning solutions.

2.1 Link prediction

2.1.1 Overview and applications

The social network contains information about social individuals and the similarity relationships between them. It can be represented by treating the network as a graph, in which a vertex corresponds to a specific person and an edge shows a broadly defined relation between two social network members [Otte and Rousseau, 2002]. The task that concerns the ability to predict non-existing connections in the network that may arise in the future, based on the knowledge regarding the present network's structure is called the link prediction (the formal definition of the link prediction problem is presented in definition 1.0.3) [Liben-Nowell and Kleinberg, 2007].

The link prediction problem can be also defined as a problem of finding missing links that are likely to exist, but are not marked as existing in the network. This approach is different

from the one predicting links, as it works with a static snapshot of network, rather than a network evolution. [Yan and Gregory, 2011].

The link prediction has many applications in social networks, and beyond. It can be used for recommending interesting and relevant content in social networks [Chiluka et al., 2011], recommend purchases in e-commerce [Li et al., 2014b], finding hidden groups of terrorists [Clauset et al., 2008], finding unknown interactions between drugs
[Fokoue et al., 2016, Harada et al., 2018]. Link prediction has been also applied to co-authorship networks finding scientists who are likely to collaborate in the future [Bhardwaj and Lu, 2019].

2.1.2 Similarity-based approach

In [Wang et al., 2014], the Authors proposed a generic framework for solving a link prediction problem, using similarity-based approaches and learning-based approaches. In similarity-based approach, for each non-connected pair in the graph, a score is calculated based on measures for analysing the proximity of nodes, where higher score means a higher probability that this pair of nodes is connected. The Authors [Wang et al., 2014] divided different techniques of measuring the similarity score into three categories: node-based metrics, topology-based metrics, and social-theory based metrics. However in this thesis, only topology-based metrics have been described, due to their usage in benchmarks, and being a base for more complex link prediction approaches.

The topology-based metrics can be categorised based on how many maximum hops of neighbours are being used when calculated the metrics, defined as either h-order heuristics (up to h neighbours), or high-order heuristics (entire network) [Zhang and Chen, 2018].

All metrics listed below take as an input two nodes - X and Y.

Jaccard's coefficient [Jaccard, 1901] is a first-order heuristic, defined by a formula

$$J(X,Y) = \frac{|\Gamma(X) \cap \Gamma(Y)|}{|\Gamma(X) \cup \Gamma(Y)|} \tag{2.1}$$

.

Preferential Attachment Index [Yule, 1925] is a first-order heuristic, initially used for explaining a distribution of species. The application for link prediction problem is based on an idea that in social networks users with more friends tend to create more connections in the

future [Bianconi and Barabási, 2001], thus a probability of a new link between nodes x and y is proportional to $PAI(x,y)$. Then, the index is defined as:

$$PAI(X,Y) = |\Gamma(X)| \cdot |\Gamma(Y)|. \tag{2.2}$$

.

Adamic-Adard [Adamic and Adar, 2001] is a second-order heuristic, where elements with large neighbourhoods are less significant than elements shared between smaller neighbourhoods. The index is defined as

$$A(X,Y) = \sum_{u \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log(|\Gamma(u)|)} \tag{2.3}$$

.

Resource-allocation [Zhou et al., 2009] is a second-order heuristic motivated by the resource allocation process taking place in networks [Ou et al., 2007]. The index is defined by a formula:

$$RA(X,Y) = \sum_{u \in \Gamma(x) \cap \Gamma(y)} \frac{1}{|\Gamma(u)|} \tag{2.4}$$

.

The Katz descriptor [Katz, 1953] is a higher-order method, which sums over collection of paths between two nodes, exponentially damped by length to give greater for the shorter paths. The formula as follows:

$$K(X,Y) = \sum_{l=1}^{\infty} \beta^l |path^l_{x,y}| \tag{2.5}$$

where $path^l_{x,y}$ is a set of all paths between x and y with length l.

As shown in [Bhardwaj and Lu, 2019], topology-based metrics are useful in solving link prediction problem, where the Authors showed that the metrics achieve an accuracy orders of magnitude greater than random link prediction. The accuracy of link prediction varies a lot depending on some characteristics of the graphs [Zhou et al., 2009], such as clustering coefficient [Otte and Rousseau, 2002] or average node degree. However, in

[Liben-Nowell and Kleinberg, 2007] the Authors showed, that Katz method usually performs better than other metrics.

2.1.3 Learning-based approach

Solving link prediction problem is not limited to choosing one metric, and deciding whether a link is missed based solely on the metric value. Link prediction may be treated as a binary classification problem, where similarity between each pair of nodes represent a feature, and a class label is positive if the nodes are connected, otherwise class label is negative.

A key factor for efficient link prediction using classifier is extracting appropriate features. The topology-based metrics are popular, and useful features. However, in [Hasan et al., 2006], in addition to using topological features for supervised link prediction, the Authors [Hasan et al., 2006] use proximity-features, which are domain specific and represent form of proximity between the pair of vertices. For example, for co-authorship network, the proximity feature may be a size of identical keywords set in articles abstracts. Besides proximity features, the Authors defined aggregated features. The aggregate feature is a metric summarising a node based on features of its neighbours. For co-authorship network, authors constructed a new aggregate feature by calculating a sum of classification codes of research publications.

In [Wang et al., 2011], the Authors showed that supervised models trained on topology-based metrics outperform the unsupervised approach using those metrics (the best unsupervised metric Katz yields a score of precision 17.62%, where a decision tree trained on topology-based metrics achieve the precision score of 26.68%). In [Lichtenwalter et al., 2010], the Authors developed a framework based on supervised learning outperforming unsupervised link prediction methods by more than 30% area under the curve (AUC).

There are couple of reasons, described in [Lichtenwalter et al., 2010], why supervised learning is a better option to tackle link prediction problem than unsupervised learning. One of the reasons is that some metrics (e.g preferential attachment [Yule, 1925]) are better as global indicators where underlying local mechanisms such as neighbours recommendations are weaker. Yet another benefit of supervised learning is that it offers options for reducing variance such as bagging [Breiman, 1996].

2.2 Graph embeddings

Efficiency of learning-based approach relies heavily on feature extraction, and calculation. Often the features are domain-specific, and do not generalise well for different predictions. Nevertheless, it is possible to build an effective missing link classifier without manual feature engineering by using graph embeddings aiming to learn feature representation.

The goal of graph embedding is to convert a graph into a low-dimensional vector space [Bengio et al., 2013], where the graph structure, and properties are preserved in the Euclidean space.

Given graph $G = (\mathbb{V}, A)$ and dimensionality, d graph embedding δ may be defined as finding a mapping for each vertex in $\mathbb{V}$ such that some graph properties are preserved

$$\delta : \mathbb{V} \rightarrow \mathbb{R}^{d \times |V|} \tag{2.6}$$

Depending on the needs of finding an embedding, the output of graph embedding is a mapping of an entire graph, a subgraph, each node or edge. The nodes that are close in the graph (e.g belong to the same community) are embedded to have similar vectors. Graph embedding may be perceived as an unsupervised learning task, where only information about adjacency of the graph, and node attributes are used. However, finding the graph embedding may be also perceived as a supervised learning task, where the model seeking an embedding uses the additional information, e.g node classes are used. The vectors which are result of graph embedding, may be used as an input to another machine learning task, in our case link prediction classifier.

One has to be aware that learning graph embeddings has many challenges
[Wang et al., 2016]. The first challenge is that the structure of network is highly non-linear [Luo et al., 2011], so that a model capturing non-linearity well has to be used. Next challenge is that network embedding has to preserve the local and global structure of the network. Yet another challenge is a network sparsity - algorithms have to utilize a limited number of links. [Perozzi et al., 2014]

In a great overview about representational learning on graphs [Hamilton et al., 2017b], the Authors proposed a taxonomy of node embedding techniques called encoder-decoder

framework, containing an encoder mapping each node to a vector, and decoder retrieving the structural information about the graph from learned embedding on a given set of nodes. The flow for link prediction problem having an optimised encoder-decoder has to steps. Firstly, it uses the trained encoder to generate embeddings for two nodes. Later, the distance between the embeddings is calculated, or the vectors concatenated, and a classifier for finding missing links is applied.

The graph embedding methods may be divided into two groups based on encoding approach: direct encoding, and generalised encoder-decoder architectures
[Hamilton et al., 2017b].

2.2.1 Direct encodings

In direct encoding graph embeddings methods, encoder function simply maps node to vector embeddings, where Z is matrix containing the embedding for all nodes and $\mathbf{v^T}$ is a one-hot encoding vector representing particular column in matrix Z [Hamilton et al., 2017b]

$$ENC(i) = Z\mathbf{v^T} \tag{2.7}$$

The direct encodings methods are based either on random-walks, finding embeddings via encoding the neighbourhood found by random walks, and matrix-factorization methods representing graph property (e.g., node pairwise similarity) in the form of a matrix and factorise this matrix to obtain node embedding [Goyal and Ferrara, 2017].

2.2.2 Generalized encoder-decoder architectures

The direct encodings methods have many drawbacks [Hamilton et al., 2017b]:

- No parameters are shared between nodes in the encoder (sharing parameters may be a form of regularization, and without sharing the number of parameters grows linearly like with the number of nodes in a graph)
- Do not include node attributes information
- Cannot generate embeddings for previously unseen node during training phase

- Does not incorporate task-specific supervision

A number of approaches is facing the issues described above. The approaches use more complex encoders than direct encodings usually leveraging deep neural networks (graph neural networks).

2.3 Matrix factorization approaches

The matrix-based factorization methods show promising results on link prediction task [Menon and Elkan, , Ou et al., 2016]. The matrix-based factorization methods are unsupervised learning algorithms, finding low-dimensional embedding keeping the structure of neighbourhoods.

One of the method is Locally Linear Embedding [Roweis and Saul, 2000], based on assumption "that data point and its neighbours lie on or close to a locally linear patch of the manifold". This idea is calculated by minimizing the error given by a formula, where matrix W summarizes the contribution datapoints to reconstructions, and vectors $\mathbf{v_i}$ are datapoints.

$$\varepsilon(W) = \sum_i |\mathbf{v_i} - \sum_j W_{ij}\mathbf{v_j}|^2 \tag{2.8}$$

HOPE (High-Order Proximity preserved Embedding) [Ou et al., 2016] is an embedding method preserving higher-order proximity using asymmetric transitivity properties of paths in graphs based on the singular value decomposition method.

2.4 Random walk approaches

DeepWalk [Perozzi et al., 2014], and its generalized version node2vec
[Grover and Leskovec, 2016] are aiming to preserve the neighbourhood in an embedded space by maximizing a probability of observing the neighbourhood of a node conditioned on its embedding [Cai et al., 2018] [Hamilton et al., 2017b]:

$$DEC(\mathbf{z_i}, \mathbf{z_j}) \approx p_{G,T}(v_j|v_i), \tag{2.9}$$

where $p_{G,T}(v_j|v_i)$ is the probability of vising v_j on a length-T random walk starting at v_i.

The process of learning attempts to minimize the following cross-entropy loss : [Cai et al., 2018] [Hamilton et al., 2017b]

$$\mathscr{L} = \sum_{(v_i,v_j \in \mathscr{D})} -log(DEC(\mathbf{z_i}, \mathbf{z_j})), \tag{2.10}$$

where the training set $\mathscr{D}$ is generating by sampling random walks starting from each node

DeepWalk samples a set of paths from the graph using truncated random walk, and treats each path as an sentence, and a node as a word. Then a SkipGram with hierarchical softmax [Mikolov et al., 2013] is applied to maximize the probability. Node2vec is an improved DeepWalk algorithm, by introducing more flexibility in exploring the neighbourhood by interpolating between BFS and DFS, and using negative sampling instead of hierarchical softmax [Mikolov et al., 2013].

DeepWalk and node2vec showed results outperforming other methods in multi-label classification, link prediction, and other machine learning tasks [Grover and Leskovec, 2016], making it a good benchmark for link-prediction algorithm efficiency. Node2vec achieved the best AUC improvement on 12.6% on the arXiv dataset [Leskovec and Krevl, 2014] (scientist collaboration network dataset) over the best-performing baseline.

In [Qiu et al., 2018] the Authors showed that random-walk based methods: DeepWalk [Perozzi et al., 2014], and node2vec [Grover and Leskovec, 2016] are implicitly performing matrix-factorization and proposed a unified framework connecting both random-walk, and matrix-factorization based.

2.5 Graph neural networks

The usage of deep neural networks showed outstanding results in many areas such as visual reasoning [Santoro et al., 2017], object detection [Girshick, 2015], and speech generation [Oord et al., 2016]. The success is partially because of ability of deep learning architectures (i.e, convolutional neural network [Fukushima et al., 1988, Lecun et al., 1998]) to extract latent representation from images, text, and speech, and many more.

Deep learning also can be applied on graphs [Scarselli et al., 2009], and it proved to

be efficient on tasks such as: recommendation systems [van den Berg et al., 2017], text-categorization [Defferrard et al., 2016], predicting properties of molecules
[Gilmer et al., 2017], or link prediction [Kipf and Welling, 2016b, Schlichtkrull et al., 2018, Zhang and Chen, 2018].

A graph neural network is defined as "differentiable graph model which incorporates neural architecture" [Wu et al., 2019]. The Authors of [Wu et al., 2019] proposed a taxonomy to categorize graph neural networks into groups. The categories are: graph convolutional networks (GCN), graph auto-encoders, graph attention networks, graph generative networks, and graph spatial-temporal networks.

Only graph convolutional networks, and graph auto-encoders are described in this thesis, as other types of networks are better suited for other applications than link prediction. Both graph convolutionlal networks, and graph auto-encoders address previously mentioned problems of direct encodings methods.

2.5.1 Graphs autoencoders

Graph autoencoders (also known as neighbourhood autoencoder methods)
[Hamilton et al., 2017b] are based on autoencoders [Hinton and Salakhutdinov, 2006]. The methods compress information about node neighbourhood, by trying to minimize a difference between a low-dimensional neighbourhood vector and its reconstruction. This allows for keeping information about network structure hidden in autoencoders weights.

One such method is Deep Neural Graph Representation (DNGR) [Cao et al., 2016]. This method captures graph structure using the pointwise mutual information of nodes on random walk, and later compressing the structure using stacked denoising autoencoders. DNGR models outperforms state-of-the-art method (including DeepWalk) on various tasks, including clustering [Cao et al., 2016].

Yet another graph autoencoder method is Structural Deep Network Embedding (SDNE) [Wang et al., 2016]. The method aims to capture both first-order, and second-order proximity of the network by combining an unsupervised autoencoder finding embedding of nodes, and their neighbourhood, followed by a supervised training with a function cost similar to Laplacian Eigenmaps [Belkin and Niyogi, 2003], which forces the model to keep the similar

vertexes are mapped close in the embedding space.

2.5.2 Graph convolutional neural network

One problem with neighbourhood autoencoder methods is that the process of encoding node into a vector requires the entire graph. This issue is aimed to be resolved with methods generating embedding from local neighbourhood called neighbourhood aggregation [Hamilton et al., 2017b] or graph convolutional neural network [Kipf and Welling, 2016a]. They are called convolutional because filter parameters are shared over all locations in the graph [Kipf and Welling, 2016a].

Algorithm 1 Graph convolutional neural network algorithm. From [Hamilton et al., 2017a, Hamilton et al., 2017b]

Input: Graph $G = (\mathbb{V}, A)$, input features $\mathbf{x_v}, \forall v \in \mathbb{V})$, depth K, weight matrices $W^k, \forall k \in [1, K]$, non-linearity σ, differentiable aggregator function $\pi_k, \forall k \in [1, K]\}$, neighbourhood function $\theta : v \rightarrow 2^{\mathbb{V}}$
Output: Vector representation $\mathbf{z_v}, \forall v \in \mathbb{V}$,
Algorithm:

$\mathbf{h_v^0} \leftarrow \mathbf{x_v}, \forall v \in \mathbb{V}$
for $k \leftarrow 1$ to K **do**
  for $v \leftarrow \mathbb{V}$ **do**
    $\mathbf{h^k_{\theta(v)}} \leftarrow \pi_k(\{\mathbf{h_u^{k-1}}, \forall u \in \theta(v))$
    $\mathbf{h_v^k} \leftarrow \sigma(W^k \cdot COMBINE(\mathbf{h_v^{k-1}}, \mathbf{h^k_{\pi(v)}}))$
  end for
  $\mathbf{h_v^k} \leftarrow NORMALIZE(\mathbf{h_v^k}), \forall v \in \mathbb{V}$
end for
$\mathbf{z_v}, \leftarrow \mathbf{h_v^K}, \forall v \in \mathbb{V}$,

Having calculated embeddings for nodes, embeddings of subgraphs may be calculated [Duvenaud et al., 2015] by summing all nodes in subgraph.

The presented algorithm is an unsupervised one, but it is possible to incorporate supervision to the learning process by feeding through i.e logistic, or sigmoid function on top of the embeddings, calculating a predefined loss function, and backpropagate the gradient [Kipf and Welling, 2016a, Schlichtkrull et al., 2018].

2.5.3 Dual graph neural network

In the paper [Harada et al., 2018], the Authors proposed a solution for learning graph embeddings on graph of graphs (see definition 1.0.2) consisting of external, and internal layers and each node in external layer is an internal graph itself. They authors proposed an architecture called dual convolutional neural network, consisting of two graph convolutional neural networks extracting the features from a graph of graphs in an end-to-end manner, and a link prediction layer (see algorithm 2). To the best of my knowledge, this is the first approach of proposing a graph neural network for finding a graph embedding on such a graph structure.

The Authors evaluated the learning algorithm on three different datasets (drug-drug interaction network [Wishart et al., 2008], drug-function network [Takigawa et al., 2011], and metabolite reaction network [Kotera et al., 2014]). They showed that the proposed algorithm outperforms other benchmarks on the first dataset, which is a very dense network, whereas other methods are more efficent on more sparse graphs.

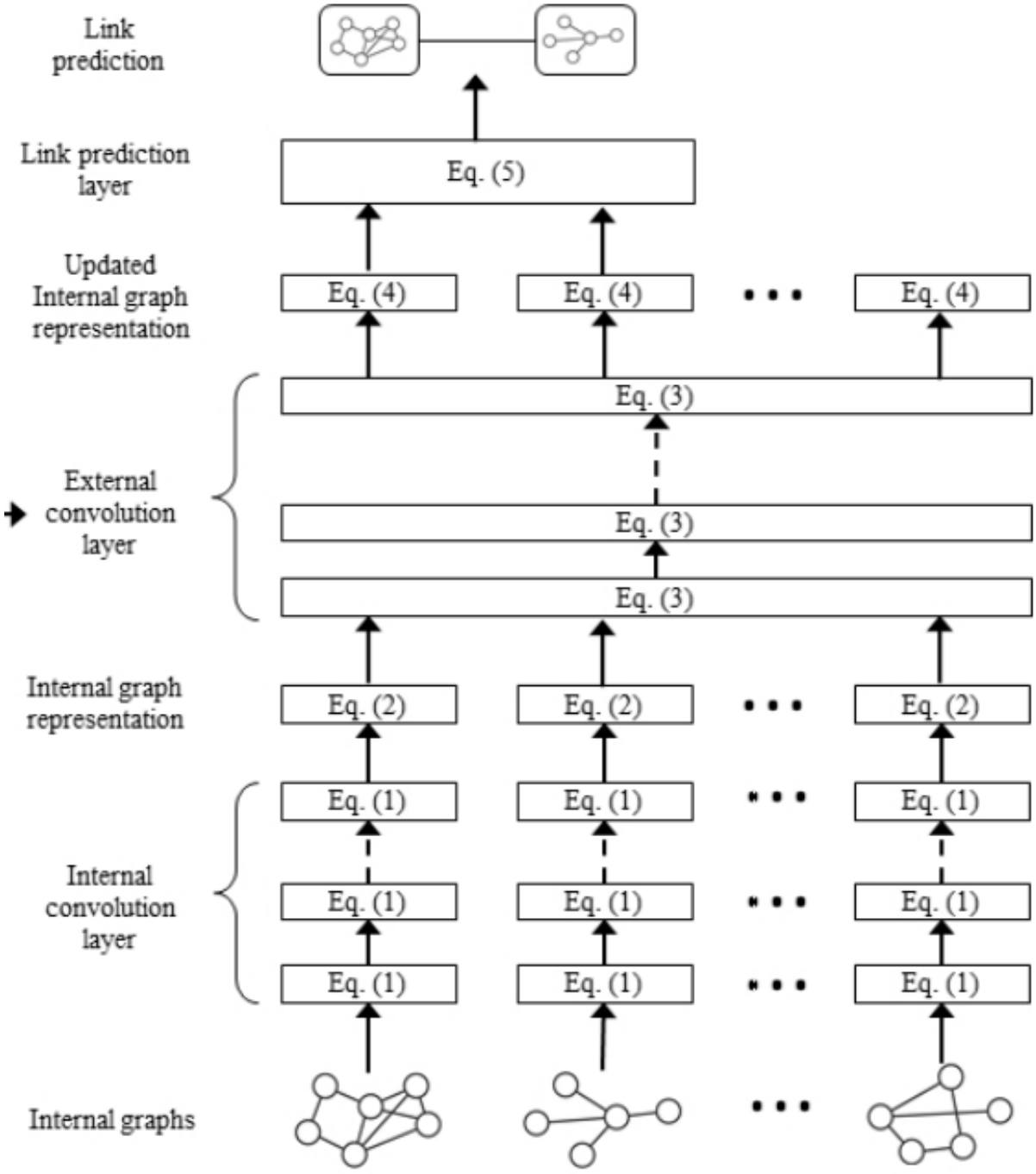

Figure 2.1: The dual convolution architecture for a graph of graphs. From: [Harada et al., 2018]

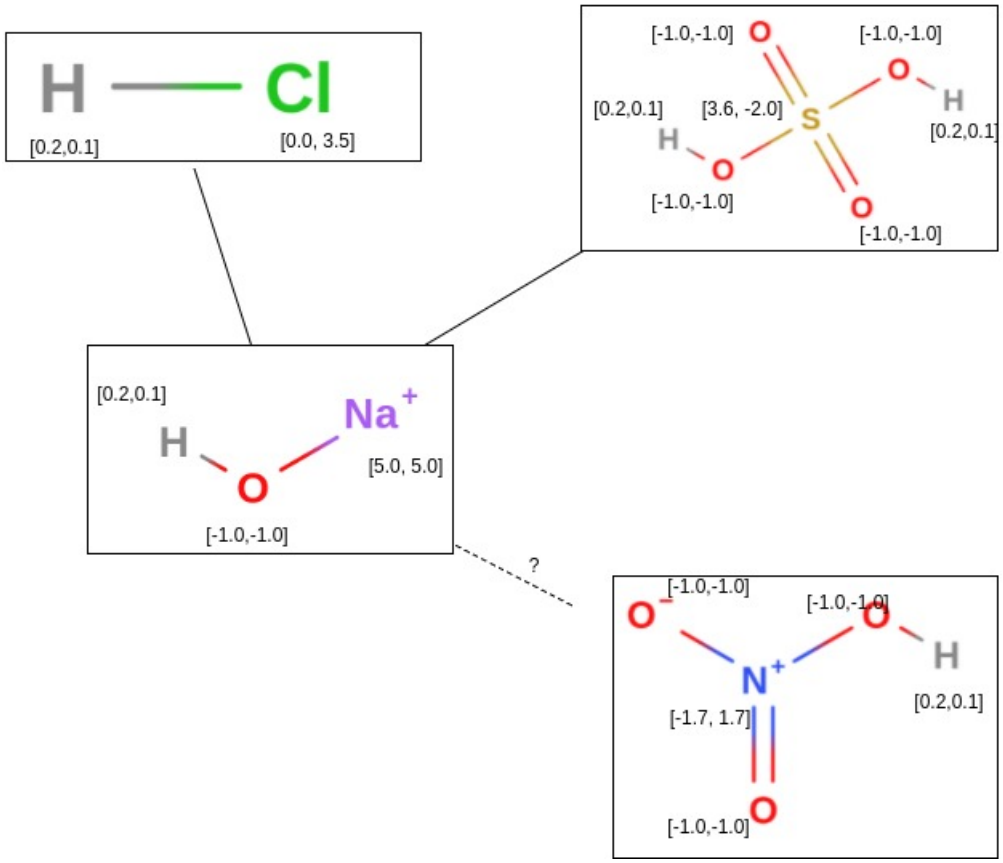

Figure 2.2: Link prediction on graph of graph

This figure presents an example of a graph of graphs. The external graph represents interaction of chemical compounds. Every link in external graph means two compounds interact with each other under standard conditions. The internal graphs represent chemical compounds. An edge in the internal graph represents a bond between two elements. All atoms of the same chemical elements share their vector representation.
In this case two edges are known to exist (the first between hydrochloric acid and sodium hydroxide, second between sodium hydroxide and sulfuric acid), the question is if an edge between sodium hydroxide and nitric acid exist (showed in the figure as a dashed line)

.

Algorithm 2 Finding an embedding for a node in external graph [Harada et al., 2018]

Input: An external graph : $\mathscr{G} = (\mathscr{V}, \mathscr{A})$, where $\mathscr{V}$ is a set of nodes, $\mathscr{A}$ is an adjacency list
An internal graph G = (V, A), where V is a set of nodes, and A is an adjacency list
Each node in the external graph is an internal graph.
f_G, σ_G, s_G, d_G are non-linear activation functions
Index j indicates for which node the embedding is calculated

Parameters: $\forall v_k \in V : \mathbf{v_k} \in \mathbb{R}^d$ (each $\mathbf{v_k}$ is initialized differently depending on type of node and trained using backpropagation).
W, M, U, V - weight matrices

Output: A fixed dimensional embedding for a given node in the external graph considering its internal graph and external graph
Algorithm:

1: For each node $v_k \in V$, starting from $\mathbf{v_k^0}$ and T iterations:
$$\mathbf{v_k^{t+1}} = f_G(W\mathbf{v_k^t} + \sum_{\mathbf{v_m} \in A_k} M\mathbf{v_m^t})$$
2: Then the internal graph representation is calculated as summing all nodes features calculated in previous step over all T steps:
$$\mathbf{g_j^T} = \sum_{\mathbf{v_k} \in V} \sigma_G(\sum_{t=0}^{T} \mathbf{v_k^t})$$
3: The representation obtained from internal graph representation are updated with the external convolutional information to incorporate information about the structure of an external graph, by making L updates, starting from l=0:
$$\mathbf{g_j^{T+l+1}} = s_G(U\mathbf{g_j^{T+l}} + \sum_{\mathbf{g_m} \in \mathscr{A}_|} V\mathbf{g_m^{T+l}})$$
4: The final representation $\mathbf{h_j^{textbfT+L}}$ finally is calculated as summing all nodes features calculated in previous step over all L steps (like in step number 2):
$$\mathbf{h_j^{T+l}} = d_G(\sum_{l=0}^{L} \mathbf{g_j^{T+l}})$$

Algorithm 3 End-to-end learning algorithm [Harada et al., 2018]

Input: Two nodes i, j from an external graph $\mathscr{G}$.
Output: A probability of link existence between nodes i, j
Algorithm:

1: Calculate embeddings h_i, h_j for nodes i, j using algorithm 2
2: Concatenate vectors h_i, h_j
3: Use this concatenated vectors as an input to feedforward neural network with a softmax function at the end giving the link probability
4: Minimize the cross-entropy loss function with respect to parameters used in algorithm 2
5: Repeat steps 1-4 for all edges from the training sets

2.6 Link prediction challenges

As there are many link prediction approaches [Wang et al., 2014], not all of them can be described in this Chapter. The selection of the methods has been made based on a criteria of popularity, usage in benchmarks, and their role in research and progress in link prediction area. The example of different approaches to link prediction are such categories of methods as probabilistic graph models, maximum likelihood methods, stochastic relational models [Wang et al., 2014, Lü and Zhou, 2011] and many more.

In this thesis, the evaluation is focused on classic link prediction definition, however link predicion problem has many variants [Wang et al., 2014]. One link prediction problem setup is finding temporal links [Dunlavy et al., 2011] - combining information when a link was formed. Another interesting problem is finding links in heterogeneous networks - finding links in network containing different types of links, and nodes.

The key challenges in designing, and evaluating link prediction algorithms are [Wang et al., 2014]:

- Overcoming imbalance datasets (number of present link is orders of magnitudes smaller than number of possible links). The datasets are very sparse so it is difficult to build statistical models, design algorithm working in reasonable time, and space complexity, quantify the confidence of evaluation, and there is still room for improvement in performance of the algorithms.
- Incorporating research from social theory, and social networks analysis
- Leveraging the information about network structure, node, and edges attributes (including case when the edges, and nodes are of different kind - heterogeneous networks)
- Standardised benchmarks and datasets - currently evaluation of algorithms is not standardised. It is very often done on non-reproducible datasets often chosen for supporting authors statement or claim. Even the same datasets vary because of their various versions, and preprocessing steps, leading to difficulties in fair comparison of algortihms.

2.7 Community detection

In many real world networks a distribution of nodes degree follow a power law [Barabasi and Albert, 1999], and the distribution of edges is not globally even, where a significant percentage of edges are concentrated within particular groups of vertices and much fewer between these groups. This features is called community structure [Girvan and Newman, 2002].

The community structures are important in finding missing links [De Bacco et al., 2017]. The reason is that a probability of forming a link within a community is higher than a probability of a link between communities. Knowing a structure of communities may help in analysing the predictions of a algorithm finding missing links by analysing a confusion matrix of link prediction algorithm within and between communities.

A problem of finding communities based on a given graph is called community detection. The communites structures, properties, and community detection algorithms has been thoroughly studied, and described in many excellent survey papers such as: [Fortunato, 2010, Samatova et al., 2014, Coscia et al., 2011].

An algorithm which is used in this thesis for finding communities is Girvan-Newman algorithm - a classic community detection algorithm [Girvan and Newman, 2002] (see algorithm 3).

However, in this thesis we do not use the community detection in the link prediction classifier tasks.

Algorithm 4 Girvan-Newman algorithm for community detection [Girvan and Newman, 2002]

Input: A graph
Output: A list of communities
Algorithm:

1: Compute betweenness centrality [Freeman, 1977] for each edge
2: Remove edges with highest betweenness
3: Repeat steps 1-2 until no edges are left
4: One edge removal splits the network into two distinct communities

2.8 Improving classifiers using dimensionality reduction

Classifiers based on high dimensional data can run the risk of overfitting [Trunk, 1979]. Many of the dataset attributes may be redundant or correlated, which leads to decrease of accuracy. There are techniques such as Principal Component Analysis (PCA) [Abdi and Williams, 2010] which may help with this problem by reducing the dimensionality of the dataset.

It has been shown [Grünauer and Vincze, 2015, Howley et al., 2005, Taloba et al., 2018, Janecek et al., 2008, Nasution et al., 2018, **?**] on a variety of datasets and experiment setups, that using PCA may improve the accuracy of classification.

The reasons are that PCA removes the irrelevant, noisy and redundant features from the feature vector [Taloba et al., 2018]. The accuracy of most classifiers tends to be less sensitive to the number of features when principal components are used instead of subsets of the original features [Janecek et al., 2008]. What is more, percentage of the total variability of the data captured is not correlated with the classification accuracy [Janecek et al., 2008].

The experiment have been conducted on either tabular-classification or text-classification dataset. To the best of my knowledge, no experiments on combining the dimensionality reduction and link prediction have been done yet.

Summary

In Chapter 2 we present an overview of different link prediction methods. There is no best method which achieves best results on any dataset. The efficiency of algorithms varies depending on the structure, and characteristics of the graph. Different types of methods have different advantages, and drawbacks. Similarity-based metrics are simple to calculate, but usually are outperformed by other metrics, and limited in terms of complexity of the model. Learning-based approach improves similarity-based metrics, but often requires domain-specific feature engineering, and careful classifier training, and evaluation. The problem with feature engineering is solved using graph embedding problems, but the graph embedding methods have their challenges - capturing, and preserving the structure of the network, and scalability of the algorithms. The simplest graph embedding solutions are matrix-based factorization, and random walk methods, however these methods are not perfect, and

the issues are addressed with more complex algorithms, based on deep learning - graph neural networks. The graph neural networks solve some problems with simpler approaches, however there is still a room for improvement in areas such as interpretability, scalability, designing new architectures, proper evaluation of models and so forth, incorporating graph including node attributes. As stated in the Chapter 1, the link prediction problem is not the only one application of graph embeddings found by described methods, and progress on the link prediction methods is transferable to different areas.

3. Proposed solution

In this Chapter, we present different approaches to tackle the problem defined in a previous chapter. The main research question underpinning the evaluation described in this chapter is stated as follows: How can one leverage the information defined in node attributes to increase the efficiency of link prediction algorithms? This problem is tackled on two datasets using two different approaches, one of the datasets is a dataset of drugs interaction, where the node attributes are information about chemical bonds, and components composing a drug. The second dataset is a dataset of citation of scientific articles, where the node attributes are abstracts of these articles. This chapter is divided into sections, one per each method, on datasets which details are described in Appendix A.

3.1 Dual convolutional neural network - drugs interaction

The experiment in this section are performed on drugs interaction dataset
[Marinka Zitnik and Leskovec, 2018]. The goal is to predict links between drugs, representing possibly harmful interactions. The drugs obviously are chemical compounds, and a challenge was to leverage that information in link prediction problem. In [Harada et al., 2018] authors proposed a dual convolutional neural network architecture learning embeddings and predicting links (algorithms 1 and 2) on graphs given that graphs are represented in a graph of graphs structure. For the purpose of this thesis, the model has been implemented from scratch, and evaluated against different link prediction methods.

The model is implemented in PyTorch [Paszke et al., 2017] framework. The the implementation is available at: `https://github.com/grzegorzborkowski/gcn`.

The graph of graph is created using algorithm 4. The node representation size is set to 64. The weight matrices were initialized using Xavier initialization [Glorot and Bengio, 2010]. The activation functions between layers is sofmax, in aggregation steps relu. The model has been trained for 20 epochs with learning rate 0.1 using Adam optimizer [Kingma and Ba, 2014].

In the experiment a number of negative edges in dataset has been equal to number of positive edges in the dataset.

For benchmark, link prediction using Jaccard Coefficent, Adamic-Adar, Preferential attachment, and node2vec with a logistic classifier for link prediction layer are used. The benchmark script has been based on the script [1] [Hu et al., 2018]. The results of the experiment are presented in chapter 5.1.

3.2 Dual convolutional neural network - DBLP dataset

An interesting research question is how to use, modify, or leverage the model of dual convolutional neural network clearly designed for link prediction on drugs on other link prediction task. The main issue is a construction of graph of graph structure, so that the embedding of node in an external graph depends on the internal graph nodes, and their connections. DBLP citation network may be transformed into a graph of graph structure. The external nodes are articles, the edges in external graph are citations. Each external node is made from an internal graph, where nodes in an external graph are represented by a word in the abstract of a particular document, and edge in internal graph means that words occured in a common window (represented by n-grams). The main problem with that approach is the time, and space complexity of this solution, as the requirements grow lineary with respect to number of words chosen from abstracts. For the experiment, only a small subset of network has been chosen with 501 nodes, and 679 edges, and from abstract all words which occured fewer than 50 times were removed.

[1] The original version available at: `https://github.com/lucashu1/link-prediction`

3.3 DBLP dataset - graph embeddings with paragraph embeddings

There are some problems with applying dual convolutional neural network for citation dataset. One of them is the tradeoff between a number of words for which an embedding is being found, and the training, and evaluation time of the model, as the memory complexity of the model grows lineary with a number of types of internal nodes, leading to scalability problems.

There are two key problems in using the abstracts. One of the problem is how to create a reasonable feature representation from abstracts, so that the representation retains as much information as possible, but on the other hand it is possible to incorporate the information in link prediction task. Representing the abstract in a form of text is troublesome when trying to use this information in link prediction task. An intuitive solution to this problem is to find a vector representation of each abstract, and reuse this vector in the process of link prediction algorithm training.

The key requirement in finding the vector representations of documents is that the semantic similarity should be kept, because articles on similar topics tend to cite articles in the same area. This problem may be tackled using doc2vec [Le and Mikolov, 2014]. Doc2vec algorithm after training on the entire corpus of abstracts, should be able to represent each article as a vector keeping the semantic similarity.

The second challenge is having vector representation of articles, how the vectors may be used to improve the link prediction algorithms. One possible solution is the articles vector can improve the embedding of graph which is an input to the link prediction layer. The question is how to enhance existing embedding with articles vector. One possible solution is to just concatenate the vectors, creating the embedding of nodes by stacking together an embedding of a particular node from some graph embedding method, and corresponding node embedding being an output of doc2vec on this particular article.

This solution is tested on the DBLP dataset, and the enhanced graph embedding method is node2vec [Grover and Leskovec, 2016]. For clarity this algorithm is referred as *node2vec+doc2vec*.

Algorithm 5 *node2vec+doc2vec* algorithm for link prediction

Input: A Graph G = (V,E), a set of training, and test edges. Each node in graph has additional attribute representing an abstract of the document.
Output: A list of predictions for all edges in test edges dataset.
Algorithm:

1: Train a doc2vec algorithm on a corpus of all abstracts
2: For each abstract in the graph retrieve the embedding from just trained doc2vec
3: Train a node2vec as in original paper [Grover and Leskovec, 2016], retrieving the embedding for each node
4: For each node concatenate two vectors (one coming from node2vec embedding, the other from doc2vec embedding) into one result vector
5: The edges are reconstructed using link prediction algorithm as in [Grover and Leskovec, 2016] using Hadamard product of two nodes and comparing with a threshold

This algorithm may be modified by using a different way of finding embeddings for abstract. Another way of finding the embeddings for articles may use the tf-idf index
[Salton and Buckley, 1988]. Each word in each article is given its tf-idf index for a particular article, or zero if the word does not occur in the article. Then each document may be represented as a long vector (its length is a number of words in the entire corpus), representing tf-idf for all words in the document. Obviously, due to the dimensionality, the embedding vector is not useful in link prediction task, however after reducing the dimensionality using e.g tSNE [van der Maaten and Hinton, 2008], or UMAP
[McInnes et al., 2018] to 16 or 32 dimensions this vector for each abstract may be used in link prediction task.

The result of experiment combining node2vec, node2vec+doc2vec, node2vec+tfidf on link prediction are described in chapter 5.3. The experiments are built on top of GEM [2] framework [Goyal and Ferrara, , Goyal and Ferrara, 2017]. The source code is available at: `https://github.com/grzegorzborkowski/GEM`

[2]Original GEM framework repository: `https://github.com/palash1992/GEM`

3.4 Wikipedia dataset - dimensionality reduction

One of the possible problem with link prediction methods efficiency is not that the graph embeddings vectors are missing underlying graph neighborhood and structure information, but there is a problem with retrieving this information, and using it in the final link prediction layer. One possible solution which may tackle this problem is reducing the dimensionality of embedding vectors, even accepting the fact that some information about graph may be lost. The dimensionality reduction step may decrease the sparsity of the input to the link prediction layer, leading to capturing more graph structure information, possibly leading to increase of the efficiency of link prediction method.

This hypothesis is a basis for an experiment, where accuracy of link prediction on node2vec embeddings is compared with an accuracy of link prediction on the vectors returned from a node2vec procedure, followed by a dimensionality reduction step using UMAP [McInnes et al., 2018], and node2vec followed by a dimensionality reduction step using PCA [Jolliffe, 2002].

For the purpose of the experiment, link prediction on top of 32 dimensional vectors trained using node2vec is compared with link prediction on 16 dimensional vectors being a result of applying either PCA or UMAP on top of the node2vec vectors.

Two different types of classifier are trained - Support Vector Machine classifier [Cortes and Vapnik, 1995] and Logistic Regression classifier.

The results of the experiments are desribed in chapter 5.4. The experiment script has been based on script [Hu et al., 2018]. The source is available at: `https://github.com/grzegorzborkowski/umap_node2vec_linkprediction`

3.5 DBLP dataset - dimensionality reduction

The idea, setup and execution model of this experiment is exactly the same as the previous one. The only difference is the dataset. This experiment is conducted on DBLP citation dataset [Tang et al., 2008]. The purpose of this experiment is to check whether the results obtained in experiment on link prediction and dimensionality reduction on Wikipedia dataset may be a general phenomenon or the results may be caused by the dataset structure itself.

4. Evaluation of the results

In this Chapter we present a summary and an analysis of the results of experiments described in the previous Chapter. The experiments show how does leveraging the information defined in node attributes increase the accuracy of link prediction, and that managing the dimensionality of graph embedding changes the link prediction accuracy.

4.1 Dual convolutional neural network - drugs interaction

The initial goal of the first experiment has been to validate if my implementation of the Dual Convolutional Neural Network [Harada et al., 2018] architecture can achieve higher precision scores on the link prediciton task than the baseline methods. The methods have been evaluated on the drugs interaction dataset (described in appendix A). The dataset has been chosen because it is similar in its structure to the dataset used by researchers in the original paper [Harada et al., 2018]. The similarity means that the drugs interaction dataset may be easily transformed into the graph of graphs structure. In order to validate the utility of the dual convolutional neural network model I compare its results with simple link prediction methods. Showing that the dual convolutional neural network outperforms other link prediction methods has been a crucial point in deciding whether this method may be successfully applied on link prediction task on dataset with different structure.

Method	Precision score
Dual Convolutional Neural Net	**0.7052**
Adamic-Adar	0.5037
Jaccard Coefficent	0.5037
Preferential Attachment	0.6934
node2vec	0.5273

Table 4.1: Link prediction on drugs interaction test dataset results

The experiments show that the dual convolutional neural network outperforms the standard link prediction algorithms. This may be caused because it leverages the information about chemical bonds of each drug, whereas this information is not used in predictions made by other algorithms. This experiment also shows that representing a graph in the graph of graphs structure may improve the link prediction precision. Surprisingly, preferential attachment achieves a noticeably higher score than other baseline metrics.

The reasons that dual convolutional network model outperforms other methods are is that during training uses information about the chemical bonds between elements in each drug, whereas other methods do not.

The experiment has been an inspiration to conducting other experiments. The graph of graphs structure is one way to leverage the node attributes information, in this case information about the presence of chemical bonds. The outcomes of this experiment lead to the research questions if this graph of graph structure may be successfully applied to other datasets, and what are the other possible ways to use the node attributes information to improve the link prediction algorithms.

4.2 Dual convolutional neural network - DBLP dataset

The purpose of this experiment has been to check whether transforming the citation dataset [Tang et al., 2008] (including the abstracts of research papers) into a graph of graph structure and training the dual convolutional neural network outperfoms the classic link prediction algorithms. The reason is that the promising results of the architecture of dual convolutional neural network model on the drugs interaction dataset may not generalize well on a dataset

with other structures than the drugs interaction dataset. The graph of graphs is constructed that the nodes of the external graphs are the articles and edges represent a citation relationship. The internal graph nodes are vector mappings of words in the abstracts, whereas edges in the internal graph represent a co-occurrence of the words (the words form an n-gram)

Method	**Precision score**
Dual Convolutional Neural Net	0.8082
Adamic-Adar	0.6933
Jaccard Coefficent	0.6834
Preferential Attachment	0.6704
node2vec	**0.8966**

Table 4.2: Link prediction on DBLP dataset results

The results of the experiment show that the dual convolutional neural network outperforms the topology-based link prediction methods, however node2vec achieves a higher precision score. The conclusion is that the dual convolutional neural network and the graph of graphs structure may be applied to varying datasets. For the citation dataset, node2vec achieves the highest precision score. However, in this experiment no information about the abstract content is used in the link prediction using node2vec, which leads to the next experiment.

The reasons that dual convolutional neural network model does not perform better that node2vec is that dual convolutional neural network model may not be well-suited for solving the link prediction problem on a graph of the structure as DBLP dataset. The main issue is that the DBLP dataset has hundreds of times more types of internal nodes (in that case words in abstracts) that the drugs interaction dataset which leads to significant increase in the number of training parameters and problem with model convergence.

4.3 DBLP dataset - graph embeddings with paragraph embeddings

The goal of this experiment has been to test how does combining node2vec embedding with paragraph embedding methods influence the result of the link prediction task. The baseline is the link prediction using node2vec graph embedding. The baseline has been enhanced with embeddings of the abstracts of the scientific papers. The underlying assumptions are that similar abstracts would produce similar paragraph embeddings and that articles with similar abstracts would cite each other by. Combining these two assumptions leads to the idea that, if a distance between two embeddings is low then that citation relationship is likely to occur. Two different paragraph embeddings methods have been chosen. Although doc2vec is able to represent articles more precisely, it requires a through hyperparameter tuning and many training examples. Because of that, less complex paragraph embedding method - tf-idf has been also chosen as a benchmark. Concatenating the paragraph embedding and initial node2vec embedding constructed a new embedding - an input to the link prediction layer.

This table presents the result of experiment on subset of graph with 487 nodes, and 698 positive edges.

Method	**Mean Average Precision score**
node2vec	0.0866
node2vec and doc2vec	0.0154
node2vec and tfidf	**0.0889**

Table 4.3: Link prediction on DBLP dataset with 487 nodes

As presented in the table, combining the graph embedding representing the network neighborhoods of nodes, and pargraph embedding (obtained using tf-idf) may improve the overall link prediction precision score. Combining node2vec with tf-idf slightly increases the node2vec score, whereas combining node2vec with doc2vec decreases the precision score. It may be caused because training doc2vec requires a significant number of training samples, and with so few samples, the doc2vec embedding is not trained and is noisy, making it more difficult to predict links correctly. Tf-idf does not need any training at all but adds meaningful

information about each abstract, thus the link prediction layer can leverage the information hidden in abstracts. This experiment shows that analysis of the node attributes not specific to the graph structure itself may improve the accuracy of the link prediction algorithms.

The results of the experiment may be explained by evaluating the quality of embedding. Such evaluation is described in Appendix B - datasets visualization. The quality of visualization of abstracts in DBLP dataset using tf-idf as abstract embedding method is higher than the quality of visualization using doc2vec. The quality of visualization depends on the quality of embedding which explains the fact that the doc2vec significantly decreases the link prediction accuracy.

4.4 Wikipedia dataset - dimensionality reduction

The goal of this experiment has been to measure the impact of managing the dimensionality of graph embedding on link prediction task. The hypothesis underpinning the experiment is that when a graph embedding is found, the link prediction layer may not be able to extract the necessary information about network structure, because of the sparsity of the structure. Reducing the dimensionality (e.g using PCA) removes some information about network, but also removes the sparsity of the embedding, easing the link prediction task for the link prediction layer.

The base node2vec embedding size are 32 and 16. The dimensionality reduction methods reduce the dimensionality of vectors from 32 to 16 dimensions.

Node2vec trained to 16 dimensions is also used because finding embedding and then applying the dimensionality reduction to 16 dimensions is different from finding an embedding to 16 dimensions.

Different methods of dimensionality reduction have been chosen, because of their fundamentally different assumption about the underlying structure of the data.

The experiment has been conducted on a variety of different subgraphs, varying in size. The reason is that the fewer nodes the graph has, the less information the embedding has to encode in its structure, changing the possible impact of dimensionality reduction technique on link prediction.

Tables 5.4 - 5.9 present detailed results gathered from link prediction experiments based

on different subsets of the Wikipedia dataset, each of different size. Two classifiers have been applied in the experiments, Support Vector Machine and Logistic Regression Classifier.

For all charts listed below, the dimensionality of embedding is placed in parentheses. SVM stands for Support Vector Machine classifier, whereas LR stands for Logistic Regression classifier.

Name of the method	**ROC score**	**Precision score**
Adamic-Adar	0.7887	0.7892
Jaccard Coefficient	0.7860	0.7811
Preferential Attachment	0.7366	0.7536
node2vec (32)	0.9226	0.9302
node2vec (16)	0.9127	0.9168
node2vec+UMAP (16)	0.9003	0.8887
node2vec+PCA (16)	**0.9311**	**0.9329**

Table 4.4: Link prediction on Wikipedia dataset containing 5783 nodes, 30144 training edges and 15071 testing edges,SVM classifier

Name of the method	**ROC score**	**Precision score**
Adamic-Adar	0.7904	0.7911
Jaccard Coefficient	0.7878	0.7838
Preferential Attachment	0.7401	0.7588
node2vec (32)	0.9196	0.9270
node2vec (16)	0.9082	0.9132
node2vec+UMAP (16)	0.8894	0.8870
node2vec+PCA (16)	**0.9272**	**0.9348**

Table 4.5: Link prediction on Wikipedia dataset containing 5783 nodes, 30144 training edges and 15071 testing edges,LR classifier

Tables 5.4 and 5.5 show the results obtained on the largest Wikipedia dataset subset, containing 5783 nodes, obtained using SVM and LR classifiers respectively. Node2vec + PCA outperforms other methods in both ROC score and precision score. It is followed by node2vec (32 dimensions) and node2vec (16 dimensions). Replacing PCA with UMAP decreases the ROC score by almost 4 percentage points regarding LR and over 3 percentage points regard-

ing SVM. The precision score drops by over 4 percentage points regarding SVM and almost 5 percentage points when using LR.

Name of the method	ROC score	Precision score
Adamic-Adar	0.7868	0.7845
Jaccard Coefficient	0.7838	0.7751
Preferential Attachment	0.7329	0.7540
node2vec (32)	0.9135	**0.9223**
node2vec (16)	0.9030	0.9082
node2vec+UMAP (16)	0.6668	0.6886
node2vec+PCA (16)	**0.9181**	0.9209

Table 4.6: Link prediction on Wikipedia dataset containing 2299 nodes, 10181 training edges and 5090 testing edges,SVM classifier

Name of the method	ROC score	Precision score
Adamic-Adar	0.7879	0.7841
Jaccard Coefficient	0.7844	0.7731
Preferential Attachment	0.7382	0.7543
node2vec (32)	0.9033	0.9074
node2vec (16)	0.8876	0.8912
node2vec+UMAP (16)	0.6613	0.6856
node2vec+PCA (16)	**0.9217**	**0.9255**

Table 4.7: Link prediction on Wikipedia dataset containing 2299 nodes, 10181 training edges and 5090 testing edges,LR classifier

Tables 5.6 and 5.7 present the results for the second largest subset of the Wikipedia dataset, that contains nearly 40% of nodes from the largest dataset. Regarding both, SVM and LR, classifiers, node2vec + PCA outperforms other methods in ROC score. When using LR classifier, the precision score gained by node2vec + PCA is also better than scores gained by other methods. Regarding SVM, node2vec (32 dimensions) achieves a slightly higher precision score than node2vec + PCA. Although, the difference is less than 0.2 percentage point. An interesting observation is that node2vec + UMAP achieves much worse results than any other method. The difference between results obtained by node2vec + UMAP and the second

worst performing method (Preferential Attachment) is about 7 percentage points in the ROC score and about 8 percentage points in the precision score.

Name of the method	**ROC score**	**Precision score**
Adamic-Adar	0.7632	0.7642
Jaccard Coefficient	0.7607	0.7558
Preferential Attachment	0.7239	0.7349
node2vec (32)	0.8728	0.8838
node2vec (16)	0.8705	0.8756
node2vec+UMAP (16)	0.8492	0.8362
node2vec+PCA (16)	**0.8901**	**0.8921**

Table 4.8: Link prediction on Wikipedia dataset containing 1195 nodes, 3898 training edges and 1948 testing edges,SVM classifier

Name of the method	**ROC score**	**Precision score**
Adamic-Adar	0.7536	0.7502
Jaccard Coefficient	0.7506	0.7405
Preferential Attachment	0.7137	0.7184
node2vec (32)	0.8629	0.8698
node2vec (16)	0.8484	0.8542
node2vec+UMAP (16)	0.7913	0.7921
node2vec+PCA (16)	**0.8839**	**0.8914**

Table 4.9: Link prediction on Wikipedia dataset containing 1195 nodes, 3898 training edges and 1948 testing edges,LR classifier

Tables 5.8 and 5.9 show the results for the smallest subset of the Wikipedia dataset, containing about 20% of the edges from the largest dataset. The trend here is similar to the one observed for the largest dataset, however, all methods achieve relatively worse results. Node2vec + PCA outperforms other methods, followed by node2vec (32 dimensions) and node2vec (16 dimensions). Node2vec + UMAP does perform better than Adamic-Adar, Jaccard Coefficient, and Preferential Attachment methods.

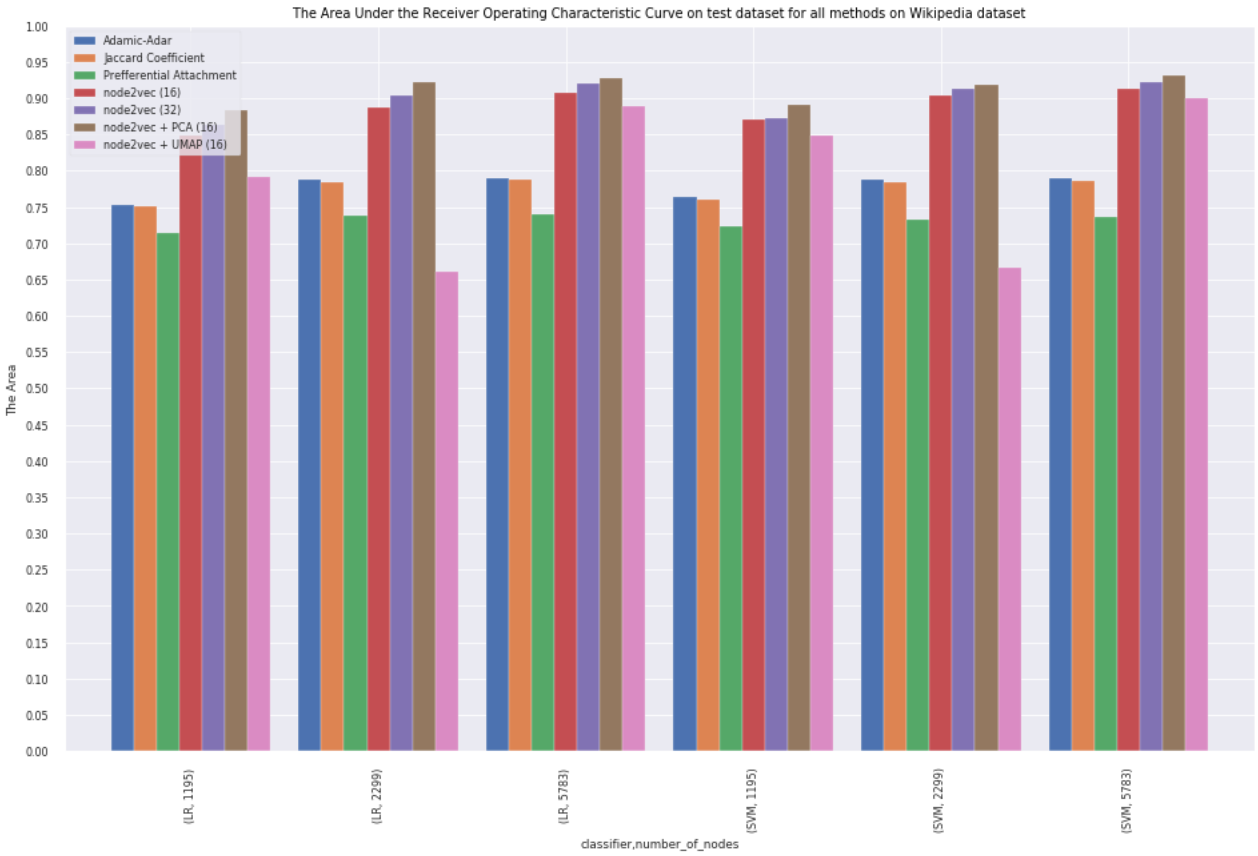

Figure 4.1: The Area Under the ROC on the link prediction task on Wikipedia dataset on subsets of different sizes and using SVM and LR classifier

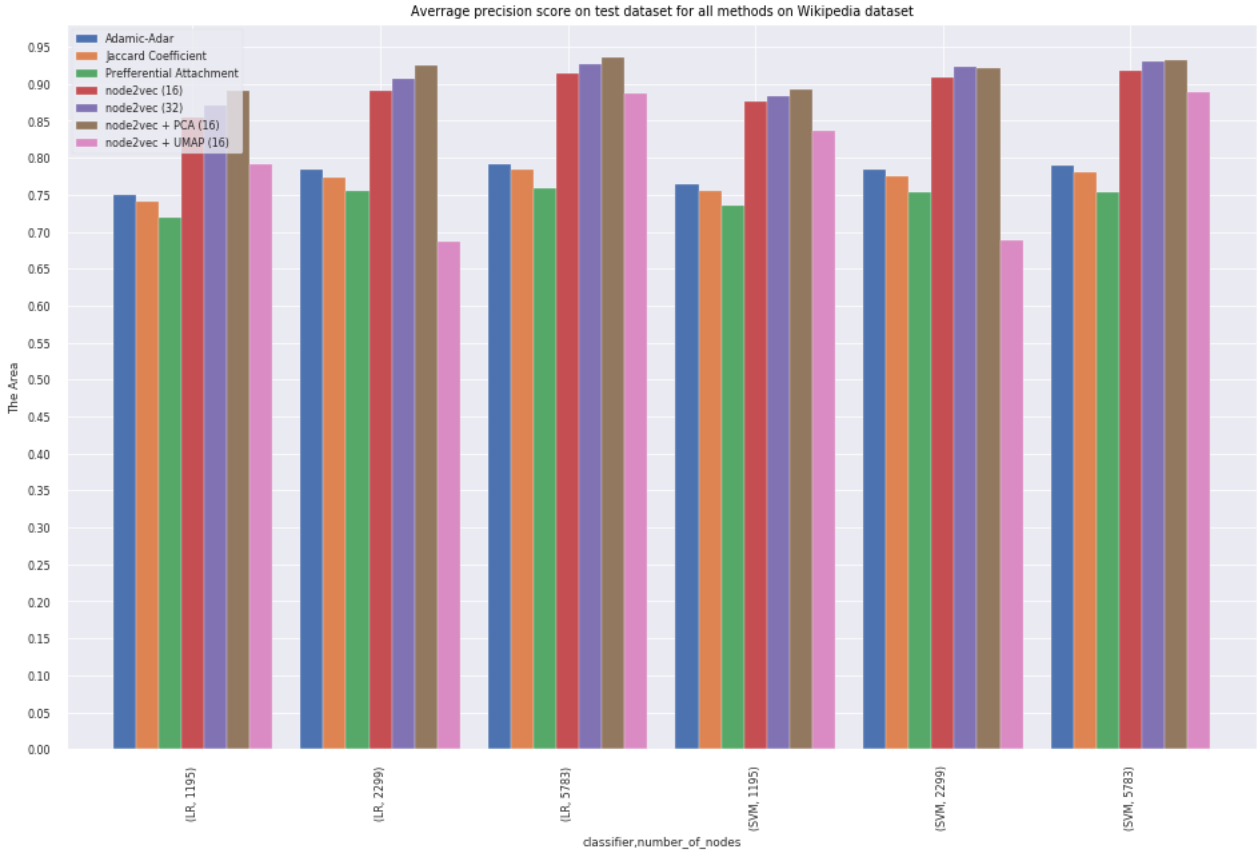

Figure 4.2: The average precision score on the link prediction task on Wikipedia dataset on subsets of different sizes and using SVM and LR classifier

Figures 5.1 and 5.2 present a visualization of the data shown in Tables 5.4 - 5.9.

The outcome of this experiment shows that reducing the dimensionality of graph embedding may increase the precision of link prediction algorithm using the embedding. While applying UMAP does not lead to any accuracy improvements, using the PCA increases both the precision score and ROC score for all the subgraphs.

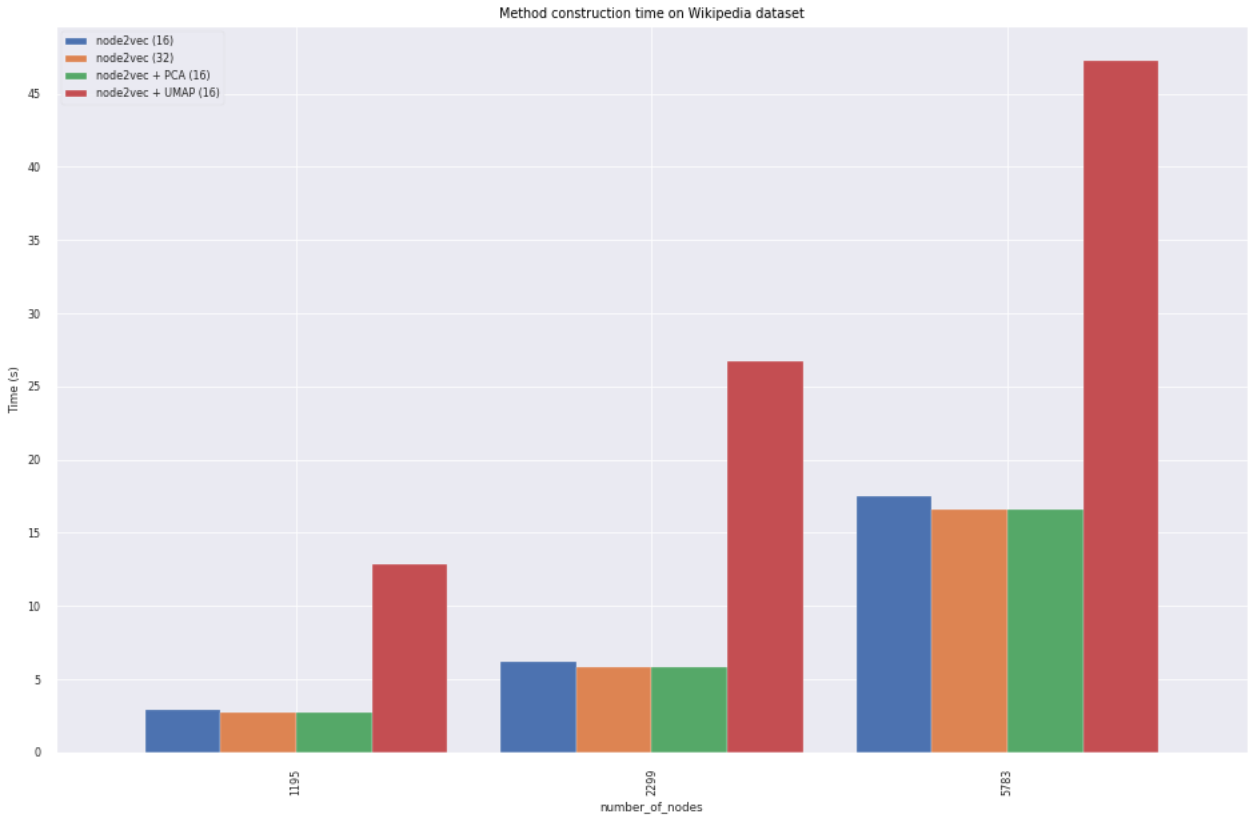

Figure 4.3: Time of the training of models on Wikipedia dataset on subsets of varying size

Figure 5.3 shows the training time for node2vec (16 and 32 dimensions) as well as node2vec + UMAP and PCA models, for all three dataset sizes. One important insight is that applying PCA on top of node2vec almost does not affect the training time, whereas finding UMAP embedding is time-consuming.

4.5 DBLP dataset - dimensionality reduction

The goal of this experiment is to verify whether dimensonality reduction improves the link prediction classifier accuracy. The setup of this experiment is the same as in the previous experiment.

The experiment has been conducted on two different subgraphs of DBLP dataset. The

classification method is SVM or Logistic Regression classifier. .

Name of the method	ROC score	Precision score
Adamic-Adar	0.6934	0.6940
Jaccard Coefficient	0.6928	0.6905
Preferential Attachment	0.7244	0.7282
node2vec (32)	0.8175	0.8497
node2vec (16)	0.8003	0.8168
node2vec+UMAP (16)	**0.8365**	0.8298
node2vec+PCA (16)	0.8201	**0.8576**

Table 4.10: Link prediction on DBLP dataset 2426 nodes, 3057 training edges and 1527 testing edges, SVM classifier

Table 5.10 presents the results for the larger DBLP subgraph. Node2vec + PCA outperforms other methods in the precision score, followed by node2vec + UMAP and node2vec (32 dimensions). Node2vec + UMAP outperforms other methods in the ROC score. Replacing UMAP with PCA decreases the ROC score by about 1 percentage point.

Name of the method	ROC score	Precision score
Adamic-Adar	0.7527	0.7504
Jaccard Coefficient	0.7520	0.7485
Preferential Attachment	0.6382	0.6576
node2vec (32)	0.8664	0.8844
node2vec (16)	0.8651	0.8818
node2vec+UMAP (16)	0.8853	0.8608
node2vec+PCA (16)	**0.8886**	**0.8994**

Table 4.11: Link prediction on DBLP dataset 614 nodes, 1122 training edges and 560 testing edges, SVM classifier

Table 5.11 presents the results obtained with the smaller subgraph of DBLP dataset. Node2vec + PCA outperforms other methods in both the ROC score and the precision score. However, the ROC score of the node2vec + UMAP is lower by only 0.3 percentage point.

The results listed in tables below are achieved using Logistic Regression classifier.

Name of the method	ROC score	Precision score
Adamic-Adar	0.6851	0.6849
Jaccard Coefficient	0.6849	0.6835
Preferential Attachment	0.7180	0.7166
node2vec (32)	0.8101	0.8455
node2vec (16)	0.7982	0.8173
node2vec+UMAP (16)	**0.8291**	0.8440
node2vec+PCA (16)	0.7998	**0.8488**

Table 4.12: Link prediction on DBLP dataset 2426 nodes, 3057 training edges and 1527 testing edges, Logistic Regression classifier

Name of the method	ROC score	Precision score
Adamic-Adar	0.7348	0.7344
Jaccard Coefficient	0.7343	0.7322
Preferential Attachment	0.6455	0.6597
node2vec (32)	0.8328	0.8638
node2vec (16)	0.8224	0.8435
node2vec+UMAP (16)	**0.8661**	0.8637
node2vec+PCA (16)	0.8500	**0.8760**

Table 4.13: Link prediction on DBLP dataset 614 nodes, 1122 training edges and 560 testing edges, Logistic Regression classifier

Tables 5.12 and 5.13 show the results of the experiments with the same datasets as above, but with different classifier. The overall trend in the results remains the same. The only significant difference is, that for the smaller dataset (Table 5.13) node2vec + UMAP outperforms node2vec + PCA.

In general, the ROC scores, as well as the precision scores obtained for the DBLP dataset, are lower than the ROC scores and precision scores obtained for the Wikipedia dataset. For DBLP node2vec + UMAP performs almost equally well as node2vec + PCA, whereas for Wikipedia node2vec + PCA gains significantly better results.

The results of this experiment confirm the conclusions derived from the previous experiment. Reducing the dimensionality of graph embedding may increase the link prediction algorithm on DBLP dataset.

5. Conclusions and future works

5.1 Achieved goals and observations

Conducted experiments prove that the quality of link prediction algorithms relies heavily on a graph structure, and its characteristics. Given that a graph may be represented in a graph of graphs structure, the dual convolutional neural network model may significantly improve the accuracy of link prediction under certain circumstances. A dual convolutional neural network has been an interesting starting point into other link prediction experiments in a broader perspective - leveraging the information in graph attributes to improve the link prediction algorithms. However one issue with the model is that only some datasets are feasible to be evaluated using this model. This the reason that the dual convolutional neural network model has not been trained on Wikipedia dataset - there was no reasonable graph of graph interpretation. Experiments on combining node2vec, and paragraph embeddings method prove that existing link prediction algorithms may be successfully enhanced with embeddings trained on graph attributes, rather than graph structure itself. However, as shown in the results section, one has to carefully choose an auxiliary embedding method, because even if the methods are based on the same input, they may have completely different quality of the output (tfidf vs doc2vec embeddings). Yet another interesting finding is that reducing the dimensionality of graph embedding layer, being an input to link prediction layer, may finally increase the accuracy of the entire link prediction pipeline.

Nevertheless, the newly proposed approaches have their drawbacks. They require careful

deliberation in choosing building block methods. Combining two methods in link prediction, increases the number of hyperparameters necessary to test, increasing the number of degrees of freedom, as well as the complexity, and unpredictability of the entire model. Just because the dual convolutional neural network may have higher accuracy than other link prediction models, it has a long training time, memory complexity, and other issues which deep learning methods have such as interpretability issues.

The key contributions and highlights of this thesis are as follows:

- The dual convolutional neural network on drugs interaction dataset model achieves precision score higher than the best baseline method by one percentage point.
- Dual convolutional neural network on DBLP dataset achieves higher precision score that topology-based baseline methods (more than ten percentage point than the best), and lower than node2vec (by nine percentage point).
- Mean average precision on link prediction task on DBLP dataset may be improve by combining vector representing the abstract calcualted using tf-idf with ndoe2vec (by about 0.2 percentage point). Using doc2vec decreases the mean average precision score significantly.
- Reducing the dimensonality of the embedding produced by node2vec may improve the precision and ROC score by up to 2 percentage point (the score depends on the size of graph and the dimensionality reduction technique) on DBLP and Wikipedia datasets.

The presented algorithms and experiments show very promising results leaving a possible room for improvement and enhancement.

5.2 Future work

The research and development in tested models is easily transferable from the link prediction to other machine learning tasks such as data visualization, regression, and so on. A very interesting problem would be applying a dual convolutional neural network for other machine learning tasks. Obviously, there is still a room for improvement within the model itself, an

example of such possible improvement is implementing the graph of graph operations using GPU operations, increasing the training, and evaluation time. There are other datasets, and real-world problems which may be modeled as a graph of graph structure, so it would be interesting to evaluate the model on different problems.

Yet another room for improvement is testing the combination of node2vec, and other graph embedding methods with different paragraph embedding methods and testing the quality of link prediction, and other learning tasks. Such methods require a different, better suited for the combined model, a way of reconstructing edges, as the standard edge reconstruction method proposed in node2vec may not be sufficient for the combined model.

Finally, there is a room for improvement in the survey of combining dimensionality reduction on graph embeddings on link prediction tasks. Other dimensionality reduction methods should be tested, as well as it requires some in-depth theoretical analysis. The experiments has been conducted on rather small graphs (with fewer than 6000 nodes) and interesting research questions is whether and how the dimensionality reduction influences the link prediction on larger graphs.

A. Datasets

In this chapter details of the datasets used for evaluation of the algorithms are described.

Drug-drug interaction network [Marinka Zitnik and Leskovec, 2018]

The dataset is identified at BioSNAP repository as ChCh-Miner dataset. The summary of the dataset at BioSNAP is described as follows: "The dataset is a network of interactions betweeen drugs, which are approved by the U.S. Food and Drug Administration. Nodes represent drugs and edges represent drug interactions. The interactions occur when the pharmacologic effect of a one drug is altered by the action of another drug, leading to unpredictable clinical effects such as adverse drug reactions." [Marinka Zitnik and Leskovec, 2018]

The dataset has 1200 unique nodes, and 10000 edges. Each drug is represented by its unique ID in DrugBank. For the experiments a subset of 1345 edges has been chosen. [Wishart et al., 2008].

The internal graphs for all nodes are created using following algorithm:

Algorithm 6 Finding an internal graph given a node in external graph for the drug-drug interaction dataset

Input: A DrugBank id of a node in external graph
Output: An internal graph of the node
Algorithm:

1: Retrieve from DrugBank a SMILES [Weininger, 1988] code for the drug with a given id
2: From the SMILES code retrieve the information about chemical elements in the drug and bonds between them
3: Each bond is an edge in the internal graph. Each element is a node in the internal graph.

DBLP dataset [Tang et al., 2008]

The dataset is a citation dataset extracted from DBLP. In this thesis, the eighth version of the dataset was analyzed. Each entry in the dataset is defined by a paper title, authors, year of publication, the publication venue, references of this paper, and abstract. Some of the fields may be empty. The dataset has been preprocessed similary to preprocessing steps defined in [Yadati et al., 2019]. The preprocessing steps are: removing articles missing datafields, counting a number of articles written by all authors, and keeping articles written only by top authors (defined by number of articles written), removing unfrequent words, and removing articles who are unfrequently cited. In the end a number of nodes, and edges in final dataset varied depending on parameters of preprocessing steps.

Wikipedia dataset

This dataset is a dataset of references between articles in English Wikipedia. The dataset contains articles in categories: art, entertainment, movies, theatre and related. The algorithm for retrieving relveant articles has been described in [Dzwinel et al., 2017]. Because of time, and memory complexity of evaluated algorithms only the nodes with the highest numbers of edges are preserved within the largest connected component.

B. Datasets visualization

DBLP dataset visualization

A set of abstracts has been visualized by performing two steps. The first step is finding a relevant class represeting each abstract. The second steps is finding an embedding for each abstract, then reducing the dimensionality to two dimensions, and then plotting as a scatter plot. The abstracts do not have preassigned class. Finding a class for each abstract increases the quality of the visualization, beacause classes enable to evaluate the quality of outcoming clusters. The classes are found using topic modelling via latent Dirichlet allocation [Blei et al., 2003]. The algorithm return a list of topic with words, and their weights making up constituting a particular topic. The topics then are assigned to real-world classes by evaluating the words, weights, and articles about given topic. The embedding is found in the same way as in chapter 3.3 - finding tf-idf [Salton and Buckley, 1988], or doc2vec [Le and Mikolov, 2014] representation for each article, and then reducing the dimensionality using UMAP [McInnes et al., 2018] or tSNE [van der Maaten and Hinton, 2008].

For each visualization two metrics describing the quality of embedding are calculated. The first metric calculates for each datapoint what fraction of nearest neighbours belong to the same class as the datapoint. This metric is calculated using k-nearest neighbours algorithms [Cover and Hart, 1967] for different settings of parameter k (1, 10, 100) averaged over all samples. This metric is calculated for the original embedding, as well as after dimensionality reduction.

The second metric returns for each point a fraction of nearest neighbours in both original dataset, and the dataset after dimensionality reduction to parameter k.

The source code for visualization and the metrics is available at `https://github.com/grzegorzborkowski/visualization-dblp`

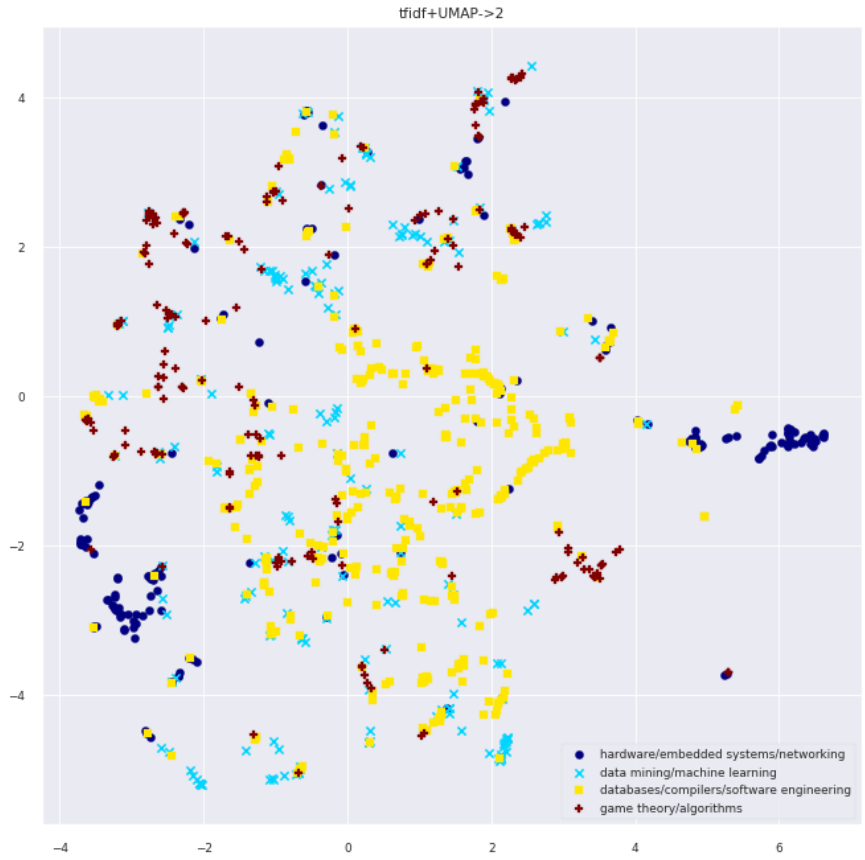

Figure B.1: Visualization of DBLP articles using tf-idf, and UMAP to two dimensions

Number of nearest neighbours (k)	**Metric no 1 before dimensionality reduction**	**Metric no 1 after dimensionality reduction**	**Metric no 2**
1	0.7201	0.6670	0.3299
10	0.6387	0.5951	0.4154
100	0.4712	0.4605	0.2664

Table B.1: Evaluation of visualization of DBLP articles using tf-idf and UMAP

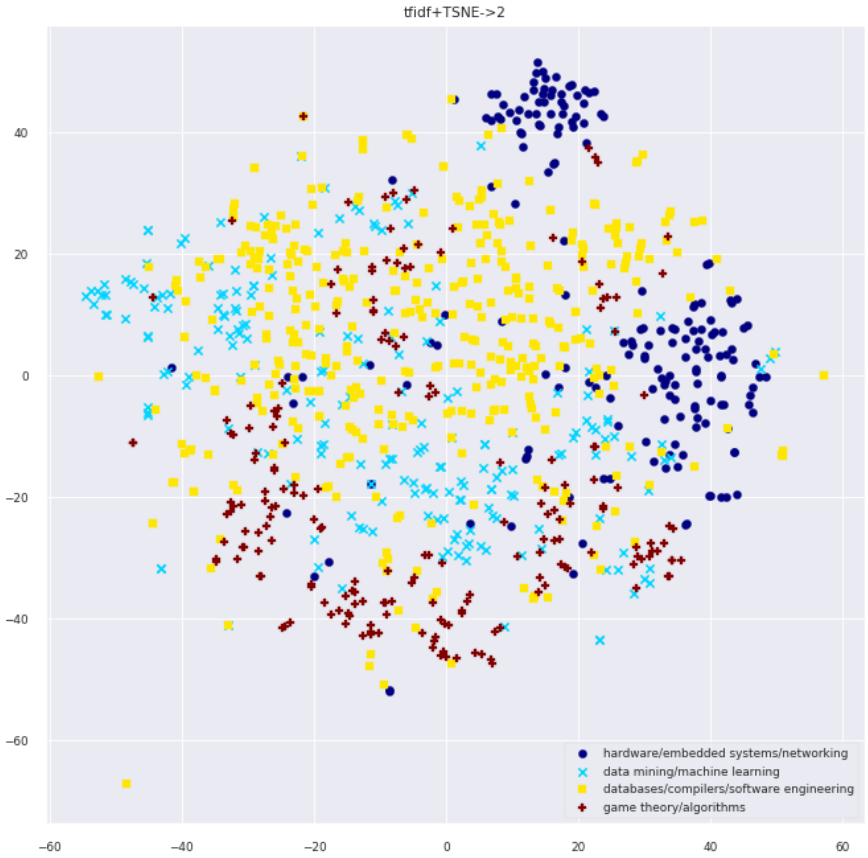

Figure B.2: Visualization of DBLP articles using tf-idf, and tSNE to two dimensions

Number of nearest neighbours (k)	**Metric no 1 before dimensionality reduction**	**Metric no 1 after dimensionality reduction**	**Metric no 2**
1	0.7201	0.7160	0.7170
10	0.6387	0.6010	0.4165
100	0.4712	0.4609	0.2937

Table B.2: Evaluation of visualization of DBLP articles using tf-idf and tSNE

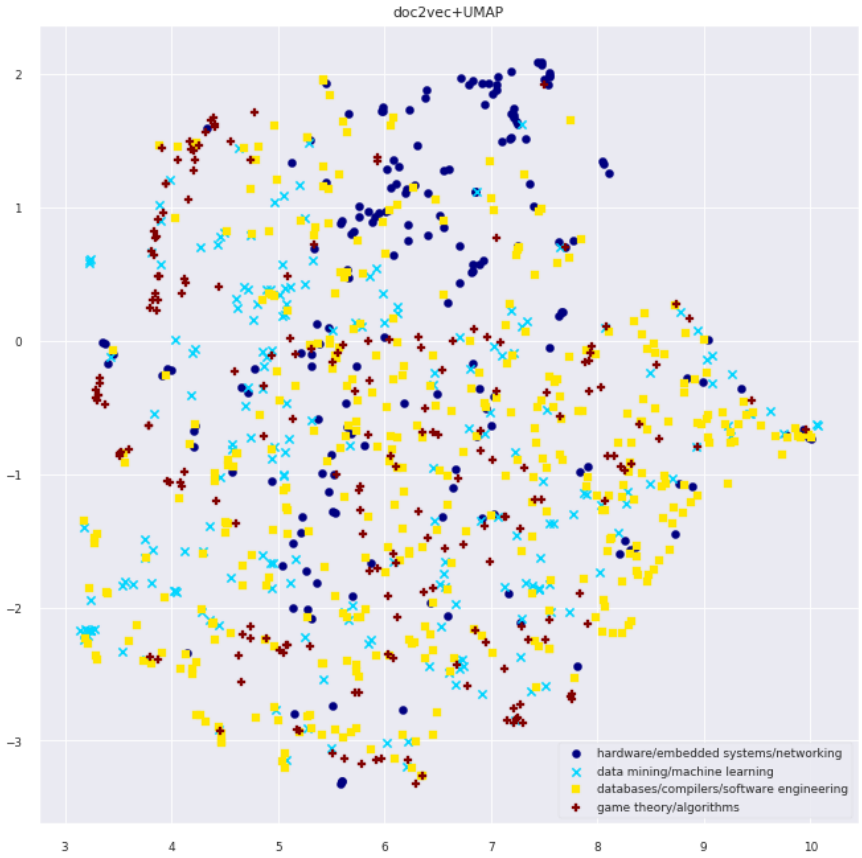

Figure B.3: Visualization of DBLP articles using doc2vec, and UMAP to two dimensions

Number of nearest neighbours (k)	**Metric no 1 before dimensionality reduction**	**Metric no 1 after dimensionality reduction**	**Metric no 2**
1	0.5199	0.4862	0.2227
10	0.4361	0.4477	0.1834
100	0.3633	0.3523	0.2118

Table B.3: Evaluation of visualization of DBLP articles using doc2vec and UMAP

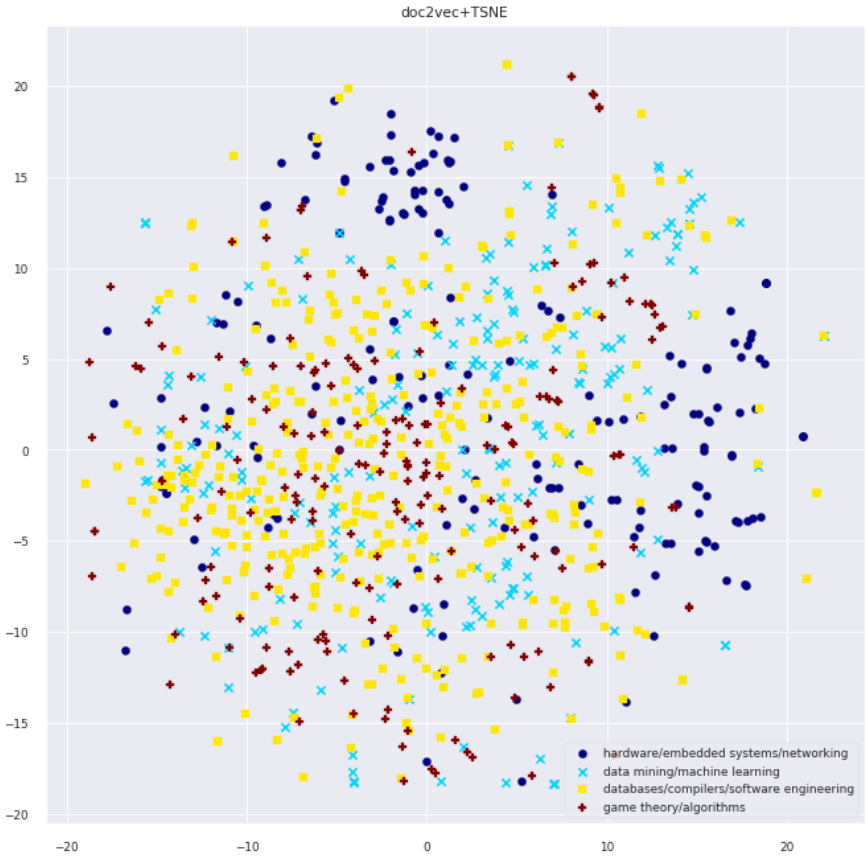

Figure B.4: Visualization of DBLP articles using doc2vec, and tSNE to two dimensions

Number of nearest neighbours (k)	**Metric no 1 before dimensionality reduction**	**Metric no 1 after dimensionality reduction**	**Metric no 2**
1	0.5199	0.5403	0.3994
10	0.4361	0.4402	0.1996
100	0.3633	0.3350	0.2330

Table B.4: Evaluation of visualization of DBLP articles using doc2vec and tSNE

C. Analysis of the dimensonality reduction experiment

An important research question is why does the dimensonality reduction improve the accuracy of link prediction classifiers.

One possible hypothesis is that the dimensionality reduction step removes the irrelevant and noisy features from the feature vector and transforms them to more useful ones [Taloba et al., 2018]. This hypothesis explains the dimensonality reduction improving classic classifiers on tabular data or text data [Taloba et al., 2018, Grünauer and Vincze, 2015].

To verify this hypothesis I did an analysis using Local Interpretable Model-agnostic Explanations (LIME) tool [Ribeiro et al., 2016] - a novel technique to explain predictions of classifiers. The tool returns a metric indicating how does each feature influence classifier decision for every sample. Summing and normalizing the metric for all samples, yields the importance score for each feature. This algorithm is conducted on all classifiers.

In the figures below the distribution of feature importance for all embedding techniques used in experiment 5.4 (dimensionality reduction on Wikipedia dataset with 5783 nodesand SVM classifier).

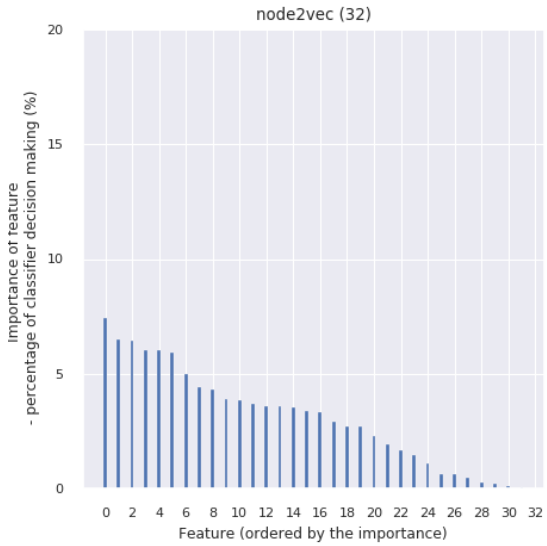

Figure C.1: The distribution of feature importance for SVM classifier when the embedding is produced by 32 dimensional node2vec vectors

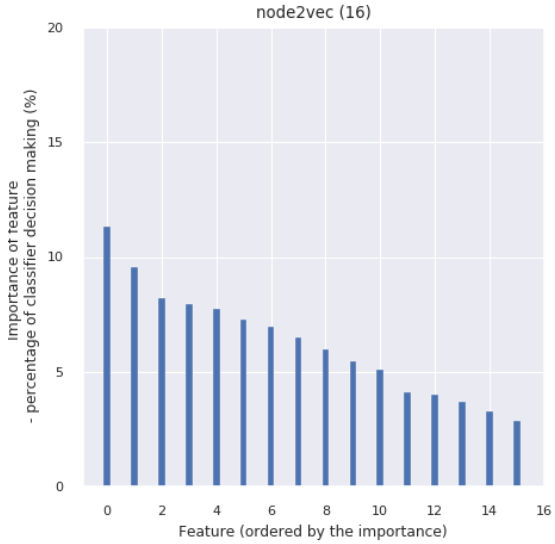

Figure C.2: The distribution of feature importance for SVM classifier when the embedding is produced by 16 dimensional node2vec vectors

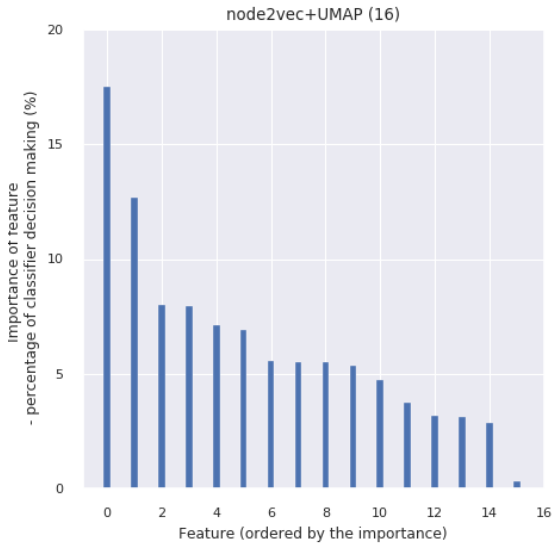

Figure C.3: The distribution of feature importance for SVM classifier when the embedding is produced by 32 dimensional node2vec vectors followed by UMAP reduction to 16 dimensions

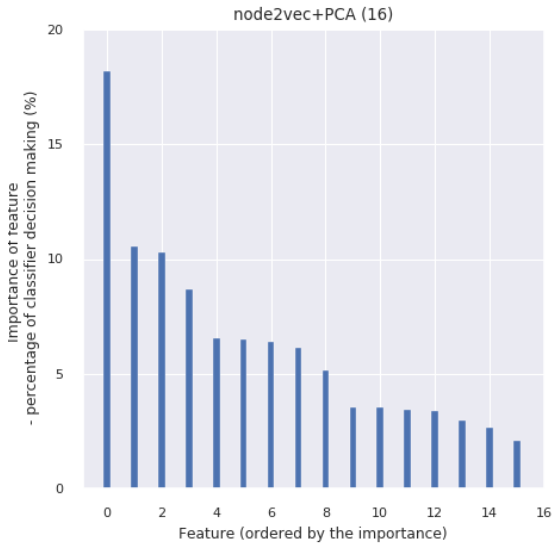

Figure C.4: The distribution of feature importance for SVM classifier when the embedding is produced by 32 dimensional node2vec vectors followed by PCA reduction to 16 dimensions

The analysis of the distributions may lead to two conclusions explaining the fact that the dimensionality reduction improves the link prediction classifiers. For 32 dimensional node2vec vectors there are obsolete features. The top three features (in terms of importance) in PCA based classifier sum up to almost 40 percent of classifier decision making process. The distributions of feature importance explain why does the dimensionality reduction based classifiers achieve higher accuracy than the classifiers without the reduction.

List of Tables

List of Figures

List of Algorithms

Bibliography

[Abdi and Williams, 2010] Abdi, H. and Williams, L. J. (2010). Principal component analysis. *WIREs Comput. Stat.*, 2(4):433–459.

[Adamic and Adar, 2001] Adamic, L. A. and Adar, E. (2001). Friends and neighbors on the web. *SOCIAL NETWORKS*, 25:211–230.

[Barabasi and Albert, 1999] Barabasi, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

[Belkin and Niyogi, 2003] Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396.

[Bengio et al., 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828.

[Bhardwaj and Lu, 2019] Bhardwaj, O. and Lu, X. (2019). Experiments with link prediction on dblp coauthorship network.

[Bianconi and Barabási, 2001] Bianconi, G. and Barabási, A.-L. (2001). Competition and multiscaling in evolving networks. *EPL (Europhysics Letters)*, 54(4):436.

[Blei et al., 2003] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

[Cai et al., 2018] Cai, H., Zheng, V. W., and Chang, K. (2018). A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering*.

[Cao et al., 2016] Cao, S., Lu, W., and Xu, Q. (2016). Deep neural networks for learning graph representations. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 1145–1152. AAAI Press.

[Chiluka et al., 2011] Chiluka, N., Andrade, N., and Pouwelse, J. (2011). A link prediction approach to recommendations in large-scale user-generated content systems. In Clough, P., Foley, C., Gurrin, C., Jones, G. J. F., Kraaij, W., Lee, H., and Mudoch, V., editors, *Advances in Information Retrieval*, pages 189–200, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Clauset et al., 2008] Clauset, A., Moore, C., and Newman, M. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98–101.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273–297.

[Coscia et al., 2011] Coscia, M., Giannotti, F., and Pedreschi, D. (2011). A classification for community discovery methods in complex networks. *Stat. Anal. Data Min.*, 4(5):512–546.

[Cover and Hart, 1967] Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. 13:21– 27.

[De Bacco et al., 2017] De Bacco, C., Power, E. A., Larremore, D. B., and Moore, C. (2017). Community detection, link prediction, and layer interdependence in multilayer networks. *Physical Review E*, 95(4):042317.

[Defferrard et al., 2016] Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In Lee, D. D.,

Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc.

[Desmarais and Cranmer, 2013] Desmarais, B. A. and Cranmer, S. J. (2013). Forecasting the locational dynamics of transnational terrorism: a network analytic approach. *Security Informatics*, 2(1):8.

[Dunlavy et al., 2011] Dunlavy, D. M., Kolda, T. G., and Acar, E. (2011). Temporal link prediction using matrix and tensor factorizations. *ACM Trans. Knowl. Discov. Data*, 5(2):10:1–10:27.

[Duvenaud et al., 2015] Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 2224–2232. Curran Associates, Inc.

[Dzwinel et al., 2017] Dzwinel, W., Wcislo, R., and Strzoda, M. (2017). ivga: visualization of the network of historical events. In *IML*, pages 38:1–38:7. ACM.

[Fokoue et al., 2016] Fokoue, A., Hassanzadeh, O., Sadoghi, M., and Zhang, P. (2016). Predicting drug-drug interactions through similarity-based link prediction over web data. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, pages 175–178, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

[Fortunato, 2010] Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174.

[Freeman, 1977] Freeman, L. C. (1977). A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41.

[Fukushima et al., 1988] Fukushima, K., Miyake, S., and Ito, T. (1988). Artificial neural networks: Theoretical concepts. chapter Neocognitron: A Neural Network Model for a

Mechanism of Visual Pattern Recognition, pages 136–144. IEEE Computer Society Press, Los Alamitos, CA, USA.

[Gao et al., 2015] Gao, F., Musial, K., Cooper, C., and Tsoka, S. (2015). Link prediction methods and their accuracy for different social networks and network metrics. *Scientific Programming*, 2015.

[Gilmer et al., 2017] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, International Convention Centre, Sydney, Australia. PMLR.

[Girshick, 2015] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1440–1448, Washington, DC, USA. IEEE Computer Society.

[Girvan and Newman, 2002] Girvan, M. and Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826.

[Glorot and Bengio, 2010] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*.

[Goyal and Ferrara,] Goyal, P. and Ferrara, E. Gem: A python package for graph embedding methods. *Journal of Open Source Software*, 3(29):876.

[Goyal and Ferrara, 2017] Goyal, P. and Ferrara, E. (2017). Graph embedding techniques, applications, and performance: A survey. *CoRR*, abs/1705.02801.

[Grover and Leskovec, 2016] Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[Grünauer and Vincze, 2015] Grünauer, A. and Vincze, M. (2015). Using dimension reduction to improve the classification of high-dimensional data. *CoRR*, abs/1505.06907.

[Hamilton et al., 2017a] Hamilton, W., Ying, Z., and Leskovec, J. (2017a). Inductive representation learning on large graphs. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 1024–1034. Curran Associates, Inc.

[Hamilton et al., 2017b] Hamilton, W. L., Ying, R., and Leskovec, J. (2017b). Representation learning on graphs: Methods and applications. *CoRR*, abs/1709.05584.

[Harada et al., 2018] Harada, S., Akita, H., Tsubaki, M., Baba, Y., Takigawa, I., Yamanishi, Y., and Kashima, H. (2018). Dual convolutional neural network for graph of graphs link prediction. *CoRR*, abs/1810.02080.

[Hasan et al., 2006] Hasan, M. A., Chaoji, V., Salem, S., and Zaki, M. (2006). Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*.

[Hinton and Salakhutdinov, 2006] Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

[Howley et al., 2005] Howley, T., Madden, M., O'Connell, M.-L., and Ryder, A. (2005). The effect of principal component analysis on machine learning accuracy with high dimensional spectral data. pages 209–222.

[Hu et al., 2018] Hu, L., Kipf, T., and Eraslan, G. (2018). lucashu1/link-prediction: v0.1: FB and Twitter Networks.

[Jaccard, 1901] Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579.

[Janecek et al., 2008] Janecek, A., Gansterer, W., Demel, M., and Ecker, G. (2008). On the relationship between feature selection and classification accuracy. In Saeys, Y., Liu,

H., Inza, I., Wehenkel, L., and de Pee, Y. V., editors, *Proceedings of the Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery at ECML/PKDD 2008*, volume 4 of *Proceedings of Machine Learning Research*, pages 90–105, Antwerp, Belgium. PMLR.

[Jolliffe, 2002] Jolliffe, I. (2002). *Principal component analysis*. Springer Verlag, New York.

[Katz, 1953] Katz, L. (1953). A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Kipf and Welling, 2016a] Kipf, T. N. and Welling, M. (2016a). Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907.

[Kipf and Welling, 2016b] Kipf, T. N. and Welling, M. (2016b). Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*.

[Kotera et al., 2014] Kotera, M., Tabei, Y., Yamanishi, Y., Muto, A., Moriya, Y., Tokimatsu, T., and Goto, S. (2014). Metabolome-scale prediction of intermediate compounds in multistep metabolic pathways with a recursive supervised approach. *Bioinformatics*, 30(12):i165–i174.

[Le and Mikolov, 2014] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org.

[Lecun et al., 1998] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324.

[Lei and Ruan, 2013] Lei, C. and Ruan, J. (2013). A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity. *Bioinformatics*, 29(3):355–364.

[Leskovec and Krevl, 2014] Leskovec, J. and Krevl, A. (2014). SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`.

[Li et al., 2014a] Li, J., Zhang, L., Meng, F., and Li, F. (2014a). Recommendation algorithm based on link prediction and domain knowledge in retail transactions. *Procedia Computer Science*, 31:875–881.

[Li et al., 2014b] Li, J., Zhang, L., Meng, F., and Li, F. (2014b). Recommendation algorithm based on link prediction and domain knowledge in retail transactions. *Procedia Computer Science*, 31:875–881.

[Liben-Nowell and Kleinberg, 2007] Liben-Nowell, D. and Kleinberg, J. (2007). The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.*, 58(7):1019–1031.

[Lichtenwalter et al., 2010] Lichtenwalter, R., Lussier, J. T., and Chawla, N. V. (2010). New perspectives and methods in link prediction. In *KDD*, pages 243–252. ACM.

[Lü and Zhou, 2011] Lü, L. and Zhou, T. (2011). Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6):1150–1170.

[Luo et al., 2011] Luo, D., Ding, C., Nie, F., and Huang, H. (2011). Cauchy graph embedding. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 553–560, USA. Omnipress.

[Marinka Zitnik and Leskovec, 2018] Marinka Zitnik, Rok Sosič, S. M. and Leskovec, J. (2018). BioSNAP Datasets: Stanford biomedical network dataset collection. `http://snap.stanford.edu/biodata`.

[McInnes et al., 2018] McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*.

[Menon and Elkan,] Menon, A. and Elkan, C. Link prediction via matrix factorization. In Gunopulos, D., Hofmann, T., Malerba, D., and Vazirgiannis, M., editors, *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 437–452. Springer Berlin / Heidelberg.

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

[Murata and Moriyasu, 2008] Murata, T. and Moriyasu, S. (2008). Link prediction based on structural properties of online social networks. *New Generation Comput.*, 26:245–257.

[Nasution et al., 2018] Nasution, M. Z. F., Sitompul, O. S., and Ramli, M. (2018). PCA based feature reduction to improve the accuracy of decision tree c4.5 classification. *Journal of Physics: Conference Series*, 978:012058.

[Newman, 2010] Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press.

[Oord et al., 2016] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. cite arxiv:1609.03499.

[Otte and Rousseau, 2002] Otte, E. and Rousseau, R. (2002). Social network analysis: a powerful strategy, also for the information sciences. *Journal of Information Science*, 28(6):441–453.

[Ou et al., 2016] Ou, M., Cui, P., Pei, J., Zhang, Z., and Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1105–1114, New York, NY, USA. ACM.

[Ou et al., 2007] Ou, Q., Jin, Y. D., Zhou, T., Wang, B. H., and Yin, B. Q. (2007). Power-law strength-degree correlation from resource-allocation dynamics on weighted networks. *Phys. Rev. E*, 75:021102.

[Paszke et al., 2017] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

[Perozzi et al., 2014] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA. ACM.

[Qiu et al., 2018] Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., and Tang, J. (2018). Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 459–467, New York, NY, USA. ACM.

[Ribeiro et al., 2016] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA. ACM.

[Roweis and Saul, 2000] Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326.

[Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. In *INFORMATION PROCESSING AND MANAGEMENT*, pages 513–523.

[Samatova et al., 2014] Samatova, N., Padmanabhan, K., Seay, R., Harlalka, J., Ranshous, S., Gjeltema, L., Bello, G., and Harenberg, S. (2014). Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439.

[Santoro et al., 2017] Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. (2017). A simple neural network module for relational reasoning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan,

S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 4967–4976. Curran Associates, Inc.

[Scarselli et al., 2009] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *Trans. Neur. Netw.*, 20(1):61–80.

[Schlichtkrull et al., 2018] Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2018). Modeling relational data with graph convolutional networks. In *ESWC*.

[Takigawa et al., 2011] Takigawa, I., Tsuda, K., and Mamitsuka, H. (2011). Mining Significant Substructure Pairs for Interpreting Polypharmacology in Drug-Target Network. *PLoS ONE*, 6(2):e16999+.

[Taloba et al., 2018] Taloba, A. I., Eisa, D. A., and Ismail, S. S. I. (2018). A comparative study on using principle component analysis with different text classifiers. *CoRR*, abs/1807.03283.

[Tang et al., 2008] Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., and Su, Z. (2008). Arnetminer: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 990–998, New York, NY, USA. ACM.

[Trunk, 1979] Trunk, G. V. (1979). A problem of dimensionality: A simple example. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(3):306–307.

[van den Berg et al., 2017] van den Berg, R., Kipf, T. N., and Welling, M. (2017). Graph convolutional matrix completion. *CoRR*, abs/1706.02263.

[van der Maaten and Hinton, 2008] van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

[Wang et al., 2016] Wang, D., Cui, P., and Zhu, W. (2016). Structural deep network embedding. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1225–1234, New York, NY, USA. ACM.

[Wang et al., 2011] Wang, D., Pedreschi, D., Song, C., Giannotti, F., and Barabási, A. (2011). Human mobility, social ties, and link prediction. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'11*, pages 1100–1108.

[Wang et al., 2014] Wang, P., Xu, B., Wu, Y., and Zhou, X. (2014). Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58.

[Weininger, 1988] Weininger, D. (1988). Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36.

[Wishart et al., 2008] Wishart, D. S., Knox, C., Guo, A., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B., and Hassanali, M. (2008). Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic Acids Research*, 36(Database-Issue):901–906.

[Wu et al., 2019] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2019). A comprehensive survey on graph neural networks. *CoRR*, abs/1901.00596.

[Yadati et al., 2019] Yadati, N., Nitin, V., Nimishakavi, M., Yadav, P., Louis, A., and Talukdar, P. (2019). Link prediction in hypergraphs using graph convolutional networks.

[Yan and Gregory, 2011] Yan, B. and Gregory, S. (2011). Finding missing edges and communities in incomplete networks. *Journal of Physics A: Mathematical and Theoretical*, 44(49):495102.

[Yule, 1925] Yule, G. U. (1925). A mathematical theory of evolution, based on the conclusions of dr. j. c. willis, f.r.s. *Royal Society of London Philosophical Transactions Series B*, 213:21–87.

[Zhang and Chen, 2018] Zhang, M. and Chen, Y. (2018). Link prediction based on graph neural networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 5171–5181. Curran Associates, Inc.

[Zhou et al., 2009] Zhou, T., Lü, L., and Zhang, Y. (2009). Predicting missing links via local information. *The European Physical Journal B-Condensed Matter and Complex Systems*, 71(4):623–630.