arXiv:1911.03042v3 [cs.CL] 7 Apr 2020

Neural Graph Embedding Methods for Natural Language Processing

A THESIS
SUBMITTED FOR THE DEGREE OF
Doctor of Philosophy
IN THE
Faculty of Engineering

BY

Shikhar Vashishth

Computer Science and Automation
Indian Institute of Science
Bangalore – 560 012 (INDIA)

April, 2020

Declaration of Originality

I, **Shikhar Vashishth**, with SR No. **04-04-00-15-12-16-1-13374** hereby declare that the material presented in the thesis titled

**Neural Graph Embedding Methods for
Natural Language Processing**

represents original work carried out by me in the **Department of Computer Science and Automation** at **Indian Institute of Science** during the years **2016-2019**.
With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name: Advisor Signature

© Shikhar Vashishth

April, 2020

All rights reserved

DEDICATED TO

My Teachers

who enlightened me with all knowledge.

Acknowledgements

I want to offer my sincere thanks to my advisors Dr. Partha Pratim Talukdar and Prof. Chiranjib Bhattacharyya, who gave me the freedom to work in my area of interest and have been very supportive throughout the course of my PhD. I also want to thank Dr. Manaal Faruqui, who accepted the role of being my mentor and guided me in my research. I am very grateful to all my teachers at Indian Institute of Science for giving me a clear understanding of the basics which were essential for completing this work.

I feel fortunate to get the opportunity to collaborate with several researchers during my PhD. I want to thank Prince Jain, Shib Sankar Das, Swayambhu Nath, Rishabh Joshi, Sai Suman, Manik Bhandari, Prateek Yadav, Soumya Sanyal, Vikram Nitin, Shyam Upadhyay, Gaurav Singh Tomar and all the members of MALL Lab for their support and help. I am also thankful to my parents and friends for their support throughout my stay in Bangalore. Finally, I would like to thank Almighty God for all His blessings without which this would not have been possible.

Abstract

Graphs are all around us, ranging from citation and social networks to Knowledge Graphs (KGs). They are one of the most expressive data structures which have been used to model a variety of problems. Embedding graphs involve learning a representation of all nodes (and relations) in the graph which allows to effectively utilize graphs for various downstream problems. In this thesis, we explore such techniques for alleviating sparsity problem in knowledge graphs and for tasks such as document timestamping and word representation learning. We also list some of the limitations of existing graph embedding methods and propose solutions for addressing them.

Knowledge graphs are structured representations of facts in a graph, where nodes represent entities and edges represent relationships between them. Recent research has resulted in the development of several large KGs; examples include DBpedia, YAGO, NELL, and Freebase. However, all of them tend to be sparse with very few relations associated with each entity. For instance, NELL KG consists of only 1.34 facts per entity. In the first part of the thesis, we explore two solutions to alleviate this problem through neural graph embedding based techniques: (1) KG Canonicalization, i.e., identifying and merging duplicate entities in a KG, (2) Relation Extraction which involves densifying the knowledge graph by automatically extracting more relationships from unstructured text. For KG Canonicalization, we propose CESI (Canonicalization using Embeddings and Side Information), a novel approach that performs canonicalization over learned embeddings of KGs. The method extends recent advances in KG embedding by incorporating relevant NP and relation phrase side information in a principled manner. For relation extraction, we propose RESIDE, a distantly-supervised neural relation extraction method which utilizes additional side information from KGs for improved relation extraction. Both the approaches demonstrate the effectiveness of utilizing relevant side information along with graph embedding techniques for addressing knowledge graph sparsity. Through extensive experiments on multiple datasets, we demonstrate the effectiveness of our proposed methods.

Traditional Neural Networks like Convolutional Networks and Recurrent Neural Networks are constrained to learn representation of Euclidean data. However, graphs in Natural Language Processing (NLP) are prominent. Recently, Graph Convolutional Networks (GCNs) have been proposed to allow Deep Learning models to exploit graph structures by embedding them in an end-to-end fashion. GCNs have been successfully applied for several problems in NLP and computer vision. In the second part of the thesis, we explore application of GCNs in two prominent tasks: (1) Document Timestamping problem, which forms an essential component of tasks like document retrieval, and summarization. For this, we propose NeuralDater which leverages GCNs for jointly exploiting syntactic and temporal graph structures of document for obtaining state-of-the-art performance on the problem. (2) Word representation learning, which has been widely adopted across several NLP tasks. We propose SynGCN, a flexible Graph Convolution based method for learning word embeddings which utilize the dependency context of a word instead of linear context for learning more meaningful word embeddings.

Finally, in the last part of the thesis, we address two limitations of existing GCN models, viz, (1) The standard neighborhood aggregation scheme puts no constraints on the number of nodes that can influence the representation of a target node. This leads to a noisy representation of hub-nodes which covers almost the entire graph in a few hops. To address this shortcoming, we propose ConfGCN (Confidence-based GCN) which estimates confidences to determine the importance of a node on another during aggregation, thus restricting its influence neighborhood. (2) Most of the existing GCN models are limited to handle undirected graphs. However, a more general and pervasive class of graphs are relational graphs where each edge has a label and direction associated with it. Existing approaches to handle such graphs suffer from over-parameterization and are restricted to learning representation of nodes only. We propose CompGCN, a novel Graph Convolutional framework which jointly embeds entity and relations in a relational graph. CompGCN is parameter efficient and scales with the number of distinct relation types. It leverages a variety of entity-relation composition operations from KG Embedding techniques and achieves demonstrably superior results on node classification, link prediction, and graph classification tasks.

Publications based on this Thesis

The work in this dissertation is primarily related to the following peer-reviewed articles:

1. **Shikhar Vashishth**, Prince Jain, and Partha Talukdar. "CESI: Canonicalizing Open Knowledge Bases using Embeddings and Side Information". In Proceedings of the World Wide Web Conference (WWW), 2018.

2. **Shikhar Vashishth**, Shib Shankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. "Dating Documents using Graph Convolution Networks". In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), 2018.

3. **Shikhar Vashishth**, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. "RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information". In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018.

4. **Shikhar Vashishth***, Prateek Yadav*, Manik Bhandari*, and Partha Talukdar. "Confidence-based Graph Convolutional Networks for Semi-Supervised Learning". In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2019.

5. **Shikhar Vashishth**, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya, and Partha Talukdar. "Incorporating Syntactic and Semantic Information in Word Embeddings using Graph Convolutional Networks". In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL), 2019.

6. **Shikhar Vashishth***, Soumya Sanyal*, Vikram Nitin, and Partha Talukdar. "Composition-based Multi-Relational Graph Convolutional Networks". In Proceedings of 8th International Conference on Learning Representations (ICLR), 2020.

* Equal Contribution

The following articles have also been completed over the course of the PhD but are not discussed in the thesis:

7. **Shikhar Vashishth***, Soumya Sanyal*, Vikram Nitin, and Partha Talukdar. "InteractE: Improving Convolution-based Knowledge Graph Embeddings by Increasing Feature Interactions". In Proceedings of 34th Conference on Artificial Intelligence (AAAI), 2020.

8. Zhiqing Sun*, **Shikhar Vashishth***, Soumya Sanyal*, Partha Talukdar, and Yiming Yang. "A Re-evaluation of Knowledge Graph Completion Methods". In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), 2020.

9. **Shikhar Vashishth**, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. "Attention Interpretability Across NLP Tasks". *arXiv preprint arXiv:1909.11218*, 2019.

10. Prateek Yadav, Madhav Nimishakavi, Naganand Yadati, **Shikhar Vashishth**, Arun Rajkumar and Partha Talukdar. "Lovasz Convolutional Networks". In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2019.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Graphs are pervasive data structures that have been used to model a variety of problems. Embedding graphs involves learning a vector representation for each node (and relation) in the graph. This enables to effectively make inference over the graph and solve various tasks such as link prediction and node classification. Such techniques have also been extensively explored for Knowledge Graphs (KGs), which are a large storehouse of world facts in a graph format. Instances of KGs include Freebase [21], WordNet [127], YAGO [178], and NELL [132]. KGs find application in a variety of tasks, such as relation extraction [128], question answering [24, 25], recommender systems [221], and dialog systems [117]. Formally, Knowledge graphs are defined as a structured representation of facts in a graph, where nodes represent entities and edges denote relationships between them. This can be represented as a collection of triples (s, r, o), each representing a relation r between a "subject-entity" s and an "object-entity" o. A KG example is presented in Figure 1.1. Here, *John Lennon*, *Beatles*, and *Liverpool* are nodes in the knowledge graph, and edges indicate different facts about them. For instance, the fact that *"John Lennon was born in Liverpool"* is expressed through *(John Lennon, bornIn, Liverpool)* triple in the KG.

Recently, neural graph embeddings are also being exploited for several Natural Language Processing applications. Graphs are prevalent in NLP, starting from dependency parse to temporal event ordering graphs, constituency parse graphs, etc. Neural graph embeddings methods allow utilizing such graph structures for a variety of downstream applications. An instance of such methods is Graph Convolutional Networks (GCNs) [46, 87], which allows embedding graphs based on the final objective in an end-to-end fashion. GCNs are a generalization of Convolutional Neural Networks (CNNs), which have been pivotal to achieve beyond human-like performance for tasks such as object recognition [93, 224] and speech recognition [170]. GCNs have been shown to be effective for tasks such as neural machine translation [15], se-

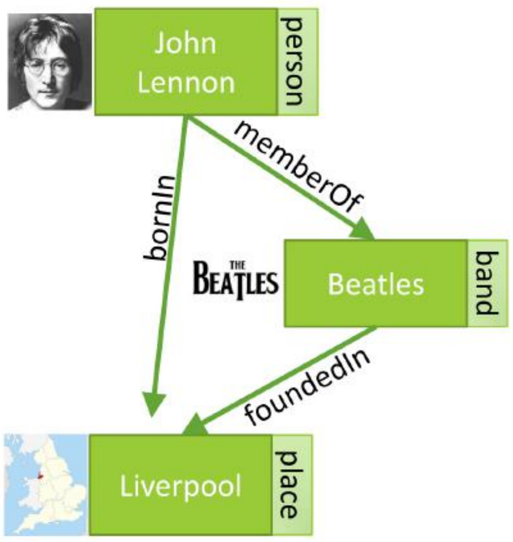


Figure 1.1: An instance of Knowledge Graph[1]. Here, nodes represent entities and edges indicate different relationships between them.

mantic role labeling [121], event detection [139]. However, the scope of their applicability for other NLP tasks is still an area of research. Furthermore, improving existing graph embedding methods also serves as an important research problem. Recent works [206, 207] have identified several limitations of Graph Convolutional Networks and have theoretically analyzed their potential.

Knowledge graphs have been utilized for a variety of tasks [117, 51]. However, most of the KGs are highly sparse with very few edges per entity, as also observed by [22]. For instance, NELL KG consists of only 1.34 facts per entity. This severely restricts their usage for several real-life applications. In the first part of the thesis, we present two graph embedding based techniques for addressing the sparsity problem in Knowledge Graphs. The proposed methods also demonstrate the effectiveness of utilizing relevant side information in a principled manner. We discuss each of them below.

1. **Knowledge Graph Canonicalization** involves identifying duplicate or redundant nodes in a KG and merging them as a single node. This can be explained through a concrete example. Given two triples in a KG: *(Barack Obama, was president of, US)* and *(Obama, born in, US)*. Identifying that *Barack Obama* and *Obama* refer to the same entity increases the overall facts per entity ratio. In our work, we focus on addressing such issues in Open KGs, which are automatically constructed knowledge graphs (using OpenIE algorithms [123, 39, 124]) without any pre-specified ontology. In spite of its importance, canonicalization is a relatively

[1]Image credits: https://kgtutorial.github.io/

unexplored problem, especially in the case of Open KGs. In this work, we propose CESI (Canonicalization using Embeddings and Side Information), a novel approach which performs canonicalization over learned embeddings of Open KBs. CESI extends recent advances in KB embedding by incorporating relevant noun and relation phrase side information in a principled manner. More details are provided in Chapter 3.

2. **Relation Extraction** involves automatically extracting semantic relationships between entity pairs from unstructured text. Most of the existing KGs like Wikidata and Freebase are human-curated. Relation extraction offers a mechanism for automatically constructing these KGs without any supervision. This allows it for further densifying existing KGs by extracting new facts from unstructured text. Since most supervised relation extraction methods require sizeable labeled training data which is expensive to construct, we utilize Distant Supervision (DS) [128] for automatically constructing a dataset. DS is based on the assumption that if two entities have a relationship in a KB, then all sentences mentioning those entities express the same relation. We propose a novel distantly-supervised neural relation extraction method, RESIDE which utilizes additional side information from KBs for improved relation extraction. It uses entity type and relation alias information for imposing soft constraints while predicting relations. RESIDE employs Graph Convolution Networks (GCN) to encode syntactic information from a text and improves performance even when limited side information is available. Please refer to Chapter 4 for more details.

In the second part of the thesis, we focus on leveraging recently proposed Graph Convolutional Networks (GCNs) [46, 87] for exploiting different graph structures in NLP. Traditional neural network architectures like Convolutional Neural Networks (CNNs) [96] and Recurrent Neural Networks (RNNs) [74] are limited to handle Euclidean data. GCNs have been proposed to address this shortcoming and have been successfully employed for improving performance on tasks such as semantic role labeling [121], neural machine translation [15], relation extraction [223], shape segmentation [216], and action recognition [76]. In this work, we begin with utilizing GCNs for exploiting syntactic and event-ordering graph structures in Document Timestamping problem which involves predicting creation date of a given document. The task is at the core of many essential tasks, such as, information retrieval [145, 103], temporal reasoning [118, 113], text summarization [200], and analysis of historical text [44]. For this, we propose NeuralDater, a GCN-based approach which is to the best of our knowledge, the first application of deep learning for the problem. The model is more elaborately described in Chapter 5. Next, we propose to use GCNs for utilizing syntactic context while learning word embeddings. Most existing word embedding methods are restricted to using the sequential context of a word. In this work, we overcome this problem by proposing SynGCN, a flexible Graph Convolution based method which utilizes the dependency context of a word without increasing the vocabulary size. We also propose SemGCN, an effective framework for incorporating diverse semantic knowledge for further enhancing learned word representations. Refer to Chapter 6 for details.

In the third part of the thesis, we address some of the significant limitations of the current Graph Convolution based models. Most of the existing GCN methods are an instantiation of *Message Passing Neural Networks* [64] which uses neighborhood aggregation scheme which puts no constraints on the number of nodes that can influence the representation of a given target node. In a k-layer model, each node is influenced by all the nodes in its k-hop neighborhood. This becomes a concern for hub nodes which covers almost the entire graph with a few hop neighbors. To alleviate this shortcoming, we propose ConfGCN, a Graph Convolutional Network which models label distribution and their confidences for each node in the graph. ConfGCN utilizes label confidences to estimate the influence of one node on another in a label-specific manner during neighborhood aggregation, thus controlling the influence neighborhood of nodes during GCN learning. Please refer to Chapter 7 for details. Apart from this, we also propose an extension of GCN models for multi-relational graphs. Most of the existing GCN models are limited to handle undirected graphs. However, a more general and pervasive class of graphs are relational graphs where each edge has a label and direction associated with it. Existing approaches to handle such graph data suffer from overparameterization and are restricted to learning representation of nodes only. We propose CompGCN, a novel Graph Convolutional framework which jointly embeds entity and relations in a relational graph. CompGCN is parameter efficient and scales with the number of relations. It leverages a variety of entity-relation composition operations from Knowledge Graph Embedding techniques. CompGCN allows the application of GCNs for a problem which requires both node and edge embeddings such as drug discovery and KG link prediction. Through extensive experiments, we demonstrate the effectiveness of our proposed approaches. More details are presented in Chapter 8.

1.1 Summary of Contributions

Our contributions in the thesis can be grouped into the following three parts:

Addressing Sparsity in Knowledge Graphs: For addressing the sparsity problem in knowledge graphs, first we propose CESI (Canonicalization using Embeddings and Side Information), a novel method for canonicalizing Open KBs using learned embeddings. To the best of our knowledge, this is the first approach to use learned embeddings and side information for canonicalizing an Open KB. CESI models the problem of noun phrase (NP) and relation phrase canonicalization jointly using relevant side information in a principled manner. This is unlike prior approaches where NP and relation phrase canonicalization were performed sequentially. For densifying existing knowledge graphs using unstructured text, we propose RESIDE, a novel neural method which utilizes additional supervision from KB in a principled manner for improving distant supervised RE. RESIDE uses Graph Convolution Networks (GCN) for modeling syntactic information and has been shown to perform competitively even with limited side information. Through extensive evaluation on various benchmark datasets, we demonstrate the effectiveness of our proposed approaches.

Exploiting Graph Convolutional Networks in NLP: We leverage recently proposed Graph Convolutional Networks for exploiting several graph structures in NLP to improve performance on two tasks: Document Timestamping and Word embeddings. We propose NeuralDater, a Graph Convolution Network (GCN)-based approach for document dating. To the best of our knowledge, this is the first application of GCNs, and more broadly deep neural network-based methods, for the document dating problem. NeuralDater is the first document dating approach which exploits the syntactic as well as temporal structure of the document, all within a principled joint model. Next, we propose SynGCN, a Graph Convolution based method for learning word embeddings. Unlike previous methods, SynGCN utilizes syntactic context for learning word representations without increasing vocabulary size. We also present SemGCN, a framework for incorporating diverse semantic knowledge (e.g., synonymy, antonymy, hyponymy, etc.) in learned word embeddings, without requiring relation-specific special handling as in previous methods. Through experiments on multiple intrinsic and extrinsic tasks, we demonstrate that our proposed methods obtain substantial improvement over state-of-the-art approaches, and also yield advantage when used with methods such as ELMo.

GNN Enhancements: Finally, we address two limitations in existing Graph Convolutional Network (GCN) based methods. For this, We propose ConfGCN, a Graph Convolutional Network framework for semi-supervised learning which models label distribution and their confidences for each node in the graph. To the best of our knowledge, this is the first confidence enabled formulation of GCNs. ConfGCN utilizes label confidences to estimate the influence of one node on another in a label-specific manner during neighborhood aggregation of GCN learning. Next, we propose CompGCN, a novel framework for incorporating multi-relational information in Graph Convolutional Networks which leverages a variety of composition operations from knowledge graph embedding techniques. Unlike previous GCN based multi-relational graph embedding methods, COMPGCN jointly learns embeddings of both nodes and relations in the graph. Through extensive experiments on multiple tasks, we demonstrate the effectiveness of our proposed method. Through extensive experiments on multiple tasks, we demonstrate the effectiveness of our proposed methods.

1.2 Organization of Thesis

The rest of the thesis is organized as follows: In Chapter 2, we review some background on Knowledge Graphs and Graph Convolutional Networks. Then in Part 1 of the thesis, we present two methods for addressing sparsity problem in Knowledge Graph: Canonicalization (Chapter 3) and Relation Extraction (Chapter 4). In Part 2, we present two novel applications of Graph Convolutional Networks in NLP for Document Timestamping (Chapter 5) and Word embedding (Chapter 6) tasks. Then, we address two limitations in existing GCN models in Part 3. We present ConfGCN for controlling influence neighborhood in GCN learning in Chapter 7 and an extension of GCNs for relational graphs in Chapter 8. Finally, we conclude in Chapter 9 by summarizing our contributions and discussing future directions.

Chapter 2

Background: Graph Convolutional Networks

In this chapter, we provide an overview of Graph Convolutional Networks (GCNs) which forms a necessary background material required for understanding the subsequent chapters of this work.

2.1 Introduction

Convolutional Neural Networks (CNNs) have lead to major breakthroughs in the era of deep learning because of their ability to extract highly expressive features. However, CNNs are restricted to Euclidean data such as images and text which have grid like structure. Non-Euclidean data such as graphs are more expressive and have been used to model a variety of problems. Graph Convolutional Networks (GCNs) address this shortcoming by generalizing CNNs' property of local receptive field, shared weights, and multiple layers to graphs. GCNs have been successfully applied to several domains such as social networks [37], knowledge graphs [165], natural language processing [121, 15], drug discovery [159] and natural sciences [105, 59]. In this chapter. we describe how CNN model for Euclidean graphs can be generalized for non-Euclidean data using Spectral Graph theory [174]. We acknowledge that most of the content of this chapter is adopted from Shuman et al. [169], Defferrard et al. [46], Kipf and Welling [87].

2.2 Preliminaries

Notations: We denote an undirected and connected graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W})$, where $\mathcal{V}$ refers to the set of nodes ($N = |\mathcal{V}|$), $\mathcal{E} = \{(u, v) \mid u, v \in \mathcal{V}\}$ indicates the set of edges, and $\boldsymbol{W}$ is a weighted adjacency matrix of the graph. If there does not exist an edge between node i and j

then W_{ij} is set to 0.

Graph Signal refers to a function defined on the vertices of a graph $\mathcal{G}$, i.e., $f : \mathcal{V} \to \mathbb{R}$. For the entire graph, it can be represented as a vector $\boldsymbol{x} \in \mathbb{R}^N$, where x_i denotes the function value at the i^{th} vertex. Figure 2.1 shows an illustration of a graph signal over a graph.

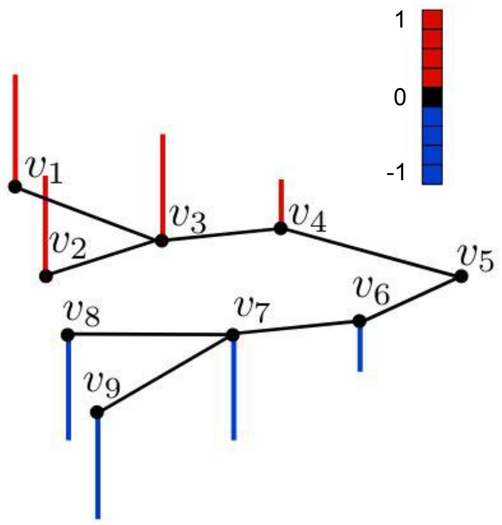


Figure 2.1: An Illustration of Graph signal over a graph. Refer to Section 2.2 for details.

Graph Laplacian (Δ) for any graph signal f is defined as:

$$(\Delta f)(i) = \sum_{j \in \mathcal{N}_i} W_{i,j}[f(i) - f(j)],$$

where $\mathcal{N}_i$ is the set of immediate neighbors of vertex i in $\mathcal{G}$. Graph Laplacian measures the difference between f and its local average. It is small for a smooth signal, i.e., connected vertices have similar values and is large when f frequently oscillates between connected vertices. Graph Laplacian can be represented as Laplacian matrix, i.e.,

$$\Delta = D - \boldsymbol{W},$$

where D is a degree matrix, i.e., $D = diag\left(\sum_{i \neq j} W_{i,j}\right)$. Δ is a real symmetric matrix, therefore, it has a complete set of orthonormal eigenvectors which we denote by $\{\phi_0, \phi_1, ..., \phi_{N-1}\}$. Moreover, all its eigenvalues are real and non-negative, i.e., $\lambda_0, \lambda_1, ..., \lambda_{N-1} \geq 0$. Further, graph

Laplacian (Δ) can be decomposed (Spectral Decomposition) as

$$\Delta = \Phi^T \Lambda \Phi,$$

where $\Phi = [\phi_0, \phi_1, ..., \phi_{N-1}]$ and $\Lambda = diag(\lambda_0, \lambda_1, ..., \lambda_{N-1})$. In the graph setting, eigenvalues and eigenvectors provide a notion of frequency. The eigenvector corresponding to smaller eigenvalues are smoother compared to the eigenvectors with larger eigenvalues. For instance, if we count cross edges ($\mathcal{Z}_{\mathcal{G}}(f)$), i.e., the number of edges connecting vertices with opposite signal value which is defined as:

$$\mathcal{Z}_{\mathcal{G}}(f)) = \{e = (i, j) \in \mathcal{E} : f(j)f(j) < 0\},$$

then, we obtain the plot as shown in Figure 2.2. This shows that with the increase in eigenvalue, the number of such edges also increases.

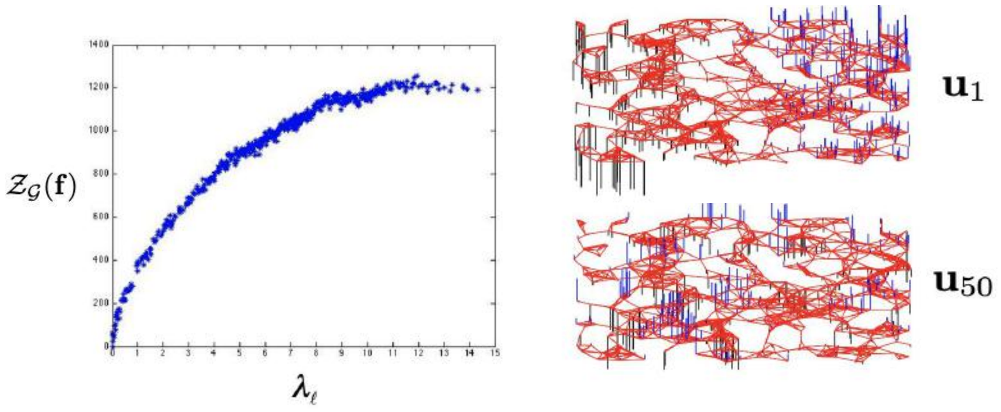


Figure 2.2: **(Left)** Shows change in cross edges with the increase in eigenvalues of graph Laplacian. **(Right)** demonstrates that an eigenvector corresponding to a smaller eigenvalue is smoother compared to the eigenvector corresponding to a larger eigenvalue.

2.3 Convolution in Euclidean space

Given two functions $f, g : [-\pi, \pi] \to \mathbb{R}$, their convolution is defined as

$$(f \star g)(x) = \int_{-\pi}^{\pi} f(t)g(x - t)dt, \tag{2.1}$$

the above formulation satisfies the following properties:

1. **Shift-invariance:** The convolution result remains unchanged on translating either of the function, i.e., $f(x - x_0) \star g(x) = (f \star g)(x - x_0)$.

2. **Convolutional Theorem:** Fourier transform diagonalizes the convolution operator which allows it to be computed in the Fourier domain as

$$\widehat{(f \star g)} = \hat{f} \cdot \hat{g},$$

where $\hat{\cdot}$ indicates the fourier transform of the function. Similarly, the convolution of two vectors $\mathbf{f} = (f_1, f_2, ..., f_n)$ and $\mathbf{g} = (g_1, g_2, .., g_n)$ can be defined as

$$\mathbf{f} \star \mathbf{g} = \mathbf{\Phi}(\mathbf{\Phi}^T \mathbf{g} \circ \mathbf{\Phi}^T \mathbf{f}) \tag{2.2}$$

3. **Computational Efficient:** Using Fast-Fourier Transform (FFT) [63], the Fourier transform can be computed efficiently in $O(\log n)$.

2.4 Convolution in non-Euclidean space

The definition of convolution as given in Equation 2.1 cannot be directly generalized for the graph setting. This is because translation, $f(x - t)$ is not defined on graphs. However, by analogy, one can define convolution operation for two vectors $\mathbf{f}, \mathbf{g} : \mathcal{V} \to \mathbb{R}^N$ as

$$\begin{aligned}\mathbf{f} \star \mathbf{g} &= \mathbf{\Phi}(\mathbf{\Phi}^T \mathbf{g} \circ \mathbf{\Phi}^T \mathbf{f}) \\ &= \mathbf{\Phi} \ \mathrm{diag}(\hat{g}_1, \hat{g}_2, ..., \hat{g}_N) \mathbf{\Phi}^T \mathbf{f} \\ &= \mathbf{\Phi} \hat{g}(\mathbf{\Lambda}) \mathbf{\Phi}^T \mathbf{f} \\ &= \hat{g}(\mathbf{\Phi} \mathbf{\Lambda} \mathbf{\Phi}^T) \mathbf{f} \\ &= \hat{g}(\Delta) \mathbf{f}\end{aligned}$$

The above formulation unlike for Euclidean space suffers from lack of shift invariance. Moreover, the filter coefficients $(\hat{g}(\Delta))$ depend on Fourier basis $\{\phi_1, \phi_2, ..., \phi_N\}$, which is expensive to compute $O(n^2)$ as FFT algorithm is not directly applicable. The above reduction allows to address this shortcoming, as $\hat{g}(\Lambda)$ can be approximated as a polynomial function of Laplacian eigenvalues, i.e.,

$$\hat{g}(\Lambda) = \sum_{k=1}^{K} \alpha_k \lambda^k,$$

where $\boldsymbol{\alpha} = (\alpha_1 \in \mathbb{R}, \alpha_2 \in \mathbb{R}, ..., \alpha_K \in \mathbb{R})^T$ is a vector of filter parameters. The filters represented by K^{th}-order polynomials of the Laplacian are exactly K-localized, i.e., restricted to capture K-hop neighborhood. Moreover, this also reduces the learning complexity to $O(K)$, the support size of the filter which is the same complexity as the standard CNNs. The above formulation, however requires $O(n^2)$ operation as $\mathbf{\Phi}\hat{g}(\mathbf{\Lambda})\mathbf{\Phi}^T\mathbf{f}$ involves multiplication with Fourier basis. One solution for this is to use Chebyshev polynomial to parameterize $\hat{g}(\Delta)$ and recursively compute it from Δ, i.e.,

$$\hat{g}(\Delta) = \sum_{k=0}^{K} \theta_k T_k(\tilde{\Delta})\mathbf{f}. \tag{2.3}$$

Here, $T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$. $T_k(\tilde{\Delta})$ denotes the k^{th}-order Chebyshev polynomial evaluated at $\tilde{\Delta} = 2\Delta/\lambda_{max} - I_N$, Laplacian with eigenvalues constrained to $[-1, 1]$. This reduces the time complexity from $O(n^2)$ to $O(K|\mathcal{E}|)$ as it involves K multiplication with sparse Δ matrix.

Kipf and Welling [87] defines a first-order approximation of the above formulation by taking $K = 1$. This reduces Equation 2.3 to

$$\hat{g}(\Delta)\mathbf{f} = (\theta_0 + \theta_1\tilde{\Delta})\mathbf{f}.$$

Now, approximating $\lambda_{max} \approx 2$ as neural network parameters will adapt to this change in scale during training and taking $\theta_0 = -\theta_1 = \theta$ gives $(\theta_0 + \theta_1\tilde{\Delta}) = (\theta - \theta(2\Delta/\lambda_{max} - I_N)) = \theta(I_N - (\Delta - I_N))$. Thus, the above equation reduces to

$$\hat{g}(\Delta)\mathbf{f} = \theta(I_N + D^{-1/2}AD^{-1/2})\mathbf{f}, \tag{2.4}$$

here, Δ is replaced with $I_N - D^{-1/2}AD^{-1/2}$, the normalized Laplacian operator. Note that since $I_N + D^{-1/2}AD^{-1/2}$ has eigenvalues in range $[0, 2]$, repeated application of this operator can lead to numerical instabilities. To address this, we use re-normalization which replaces $I_N + D^{-1/2}AD^{-1/2}$ with $\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$, where $\tilde{A} = A + I_N$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Thus, Equation 2.4 is reduces to

$$\hat{g}(\Delta)\mathbf{f} = \theta(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2})\mathbf{f}.$$

The above formulation can be generalized for a graph signal $X \in \mathbb{R}^{N\times d}$ with d-dimensional feature vector for every node and F filters as

$$\boldsymbol{H} = f(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}X\boldsymbol{W}), \tag{2.5}$$

where $\boldsymbol{W} \in \mathbb{R}^{d\times F}$ is a filter parameter, f is any non-linearity and $\boldsymbol{H} \in \mathbb{R}^{N\times F}$ is the convoluted signal matrix. For an undirected graph $\mathcal{G}$, the above equation can be re-written as

$$h_v = f\left(\sum_{u \in \mathcal{N}(v)} (\boldsymbol{W} x_u + b)\right), \quad \forall v \in \mathcal{V}. \tag{2.6}$$

Here, $\mathcal{N}(v)$ refers to the set of neighbors of v and and $b \in \mathbb{R}^F$ are learned in a task-specific setting using first-order gradient optimization. In order to capture nodes many hops away, multiple GCN layers may be stacked one on top of another. In particular, h_v^{k+1}, representation of node v after k^{th} GCN layer can be formulated as

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(W^k h_u^k + b^k\right)\right), \forall v \in \mathcal{V}. \tag{2.7}$$

2.5 GCNs for Directed and Labeled Graphs

In this section, we consider GCN formulations over graphs where each edge is labeled as well as directed, proposed by Marcheggiani and Titov [121]. In this setting, an edge from node u to v with label $l(u, v)$ is denoted as $(u, v, l(u, v))$. Based on the assumption that the information in a directed edge need not only propagate along its direction, following Marcheggiani and Titov [121] we define an updated edge set $\mathcal{E}'$ which expands the original set $\mathcal{E}$ by incorporating inverse, as well self-loop edges.

$$\mathcal{E}' = \mathcal{E} \cup \{(v, u, l(u, v)^{-1}) \mid (u, v, l(u, v)) \in \mathcal{E}\} \cup \{(u, u, \top) \mid u \in \mathcal{V})\}.$$

$$h_v^{k+1} = f\left(\sum_{u \in \mathcal{N}(v)} \left(W_{l(u,v)}^k h_u^k + b_{l(u,v)}^k\right)\right). \tag{2.8}$$

We note that the parameters $W_{l(u,v)}^k$ and $b_{l(u,v)}^k$ in this case are edge label specific.

Incorporating Edge Importance: In many practical settings, we may not want to give equal importance to all the edges. For example, in case of automatically constructed graphs, some of the edges may be erroneous and we may want to automatically learn to discard them. Edge-wise gating may be used in a GCN to give importance to relevant edges and subdue the noisy ones. Nguyen and Grishman [139], Marcheggiani and Titov [121] used gating for similar reasons and obtained large performance gain. At k^{th} layer, we compute gating value for a particular edge $(u, v, l(u, v))$ as:

$$g^k_{u,v} = \sigma\left(h^k_u \cdot \hat{w}^k_{l(u,v)} + \hat{b}^k_{l(u,v)}\right),$$

where, $\sigma(\cdot)$ is the sigmoid function, $\hat{w}^k_{l(u,v)}$ and $\hat{b}^k_{l(u,v)}$ are label specific gating parameters. Thus, gating helps to make the model robust to the noisy labels and directions of the input graphs. GCN embedding of a node while incorporating edge gating may be computed as follows.

$$h^{k+1}_v = f\left(\sum_{u \in \mathcal{N}(v)} g^k_{u,v} \times \left(W^k_{l(u,v)} h^k_u + b^k_{l(u,v)}\right)\right). \tag{2.9}$$

We utilize the GCN formulation for directed and labeled graph with (Equation 2.9) and without edge-wise gating (Equation 2.8) for most of the works in this thesis.

Part I

Addressing Sparsity in Knowledge Graphs

Chapter 3

Open Knowledge Base Canonicalization using Embeddings and Side Information

3.1 Introduction

In this chapter, we present our first solution to address the sparsity problem in Knowledge Graphs. Recent research has resulted in the development of several large *Ontological* Knowledge Bases (KBs), examples include DBpedia [6], YAGO [178], and Freebase [21]. These KBs are called ontological as the knowledge captured by them conform to a fixed ontology, i.e., pre-specified Categories (e.g., *person, city*) and Relations (e.g., *mayorOfCity(Person, City)*). Construction of such ontological KBs require significant human supervision. Moreover, due to the need for pre-specification of the ontology, such KB construction methods can't be quickly adapted to new domains and corpora. While other ontological KB construction approaches such as NELL [131] learn from limited human supervision, they still suffers from the quick adaptation bottleneck.

In contrast, Open Information Extraction (OpenIE) methods need neither supervision nor any pre-specified ontology. Given unstructured text documents, OpenIE methods readily extract triples of the form *(noun phrase, relation phrase, noun phrase)* from them, resulting in the development of large Open Knowledge Bases (Open KBs). Examples of Open KBs include TextRunner [11], ReVerb [54], and OLLIE [39, 164, 124]. While this makes OpenIE methods highly adaptable, they suffer from the following shortcoming: unlike Ontological KBs, the Noun Phrases (NPs) and relation phrases in Open KBs are not *canonicalized.* This results in storage of redundant and ambiguous facts.

Let us explain the need for canonicalization through a concrete example. Please consider the two sentences below.

Barack Obama was the president of US.
Obama was born in Honolulu.

Given the two sentences above, an OpenIE method may extract the two triples below and store them in an Open KB.

(Barack Obama, was president of, US)
(Obama, born in, Honolulu)

Unfortunately, neither such OpenIE methods nor the associated Open KBs have any knowledge that both *Barack Obama* and *Obama* refer to the same person. This can be a significant problem as Open KBs will not return all the facts associated with *Barack Obama* on querying for it. Such KBs will also contain redundant facts, which is undesirable. Thus, there is an urgent need to *canonicalize* noun phrases (NPs) and relations in Open KBs.

In spite of its importance, canonicalization of Open KBs is a relatively unexplored problem. In [61], canonicalization of Open KBs is posed as a clustering problem over *manually* defined feature representations. Given the costs and sub-optimality involved with manual feature engineering, and inspired by recent advances in knowledge base embedding [22, 142], we pose canonicalization of Open KBs as a clustering over *automatically learned* embeddings. We make the following contributions in this chapter.

- We propose Canonicalization using Embeddings and Side Information (CESI), a novel method for canonicalizing Open KBs using learned embeddings. To the best of our knowledge, this is the first approach to use learned embeddings and side information for canonicalizing an Open KB.

- CESI models the problem of noun phrase (NP) and relation phrase canonicalization *jointly* using relevant side information in a principled manner. This is unlike prior approaches where NP and relation phrase canonicalization were performed sequentially.

- We build and experiment with ReVerb45K, a new dataset for Open KB canonicalization. ReVerb45K consists of 20x more NPs than the previous biggest dataset for this task. Through extensive experiments on this and other real-world datasets, we demonstrate CESI's effectiveness (Section 3.4).

CESI's source code and datasets used in the chapter are available at `https://github.com/malllabiisc/cesi`.

3.2 Related Work

Entity Linking: One traditional approach to canonicalizing noun phrases is to map them to an existing KB such as Wikipedia or Freebase. This problem is known as Entity Linking (EL) or Named Entity Disambiguation (NED). Most approaches generate a list of candidate entities for each NP and re-rank them using machine learning techniques. Entity linking has been an active area of research in the NLP community [187, 108, 160]. A major problem with these kind of approaches is that many NPs may refer to new and emerging entities which may not exist in KBs. One approach to resolve these noun phrases is to map them to NIL or an OOKB (Out of Knowledge Base) entity, but the problem still remains as to how to cluster these NIL mentions. Although entity linking is not the best approach to NP canonicalization, we still leverage signals from entity linking systems for improved canonicalization in CESI.

Canonicalization in Ontological KBs: Concept Resolver [92] is used for clustering NP mentions in NELL [131]. It makes "one sense per category" assumption which states that a noun phrase can refer to at most one concept in each category of NELL's ontology. For example, the noun phrase "Apple" can either refer to a company or a fruit, but it can refer to only one company and only one fruit. Another related problem to NP canonicalization is Knowledge Graph Identification [156], where given a noisy extraction graph, the task is to produce a consistent Knowledge Graph (KG) by performing entity resolution, entity classification and link prediction jointly. Pujara et al. [156] incorporate information from multiple extraction sources and use ontological information to infer the most probable knowledge graph using probabilistic soft logic (PSL) [26]. However, both of these approaches require additional information in the form of an ontology of relations, which is not available in the Open KB setting.

Relation Taxonomy Induction: SICTF [144] tries to learn relation schemas for different OpenIE relations. It is built up on RESCAL [141], and uses tensor factorization methods to cluster noun phrases into *categories* (such as "person", "disease", etc.). We, however, are interested in clustering noun phrases into entities.

There has been relatively less work on the task of relation phrase canonicalization. Some of the early works include DIRT [107], which proposes an unsupervised method for discovering inference rules of the form "*X is the author of Y* $\approx$ *X wrote Y*" using paths in dependency trees; and the PATTY system [137], which tries to learn subsumption rules among relations (such as *son-of* $\subset$ *child-of*) using techniques based on frequent itemset mining. These approaches are more focused on finding a taxonomy of relation phrases, while we are looking at finding equivalence between relation phrases.

Knowledge Base Embedding: KB embedding techniques such as TransE [22], HolE [142] try to learn vector space embeddings for entities and relations present in a KB. TransE makes the assumption that for any ⟨*subject, relation, object*⟩ triple, the relation vector is a translation from the subject vector to the object vector. HolE, on the other hand, uses non-linear operators to model a triple. These embedding methods have been successfully applied for the task of link prediction in KBs. In this work, we build up on HolE while exploiting relevant side information for the task of Open KB canonicalization. We note that, even though KB embedding techniques like HolE have been applied to ontological KBs, CESI might be the first attempt to use them in the context of Open KBs.

Canonicalizing Open KBs: The RESOLVER system [214] uses string similarity based features to cluster phrases in TextRunner [11] triples. String similarity features, although being effective, fail to handle synonymous phrases which have completely different surface forms, such as *Myopia* and *Near-sightedness*.

KB-Unify [48] addresses the problem of unifying multiple Ontological and Open KBs into one KB. However, KB-Unify requires a pre-determined sense inventory which is not available in the setting CESI operates.

The most closely related work to ours is [61]. They perform NP canonicalization by performing Hierarchical Agglomerative Clustering (HAC) [183] over manually-defined feature spaces, and subsequently perform relation phrase clustering by using the AMIE algorithm [62]. CESI significantly outperforms this prior method (Section 3.4).

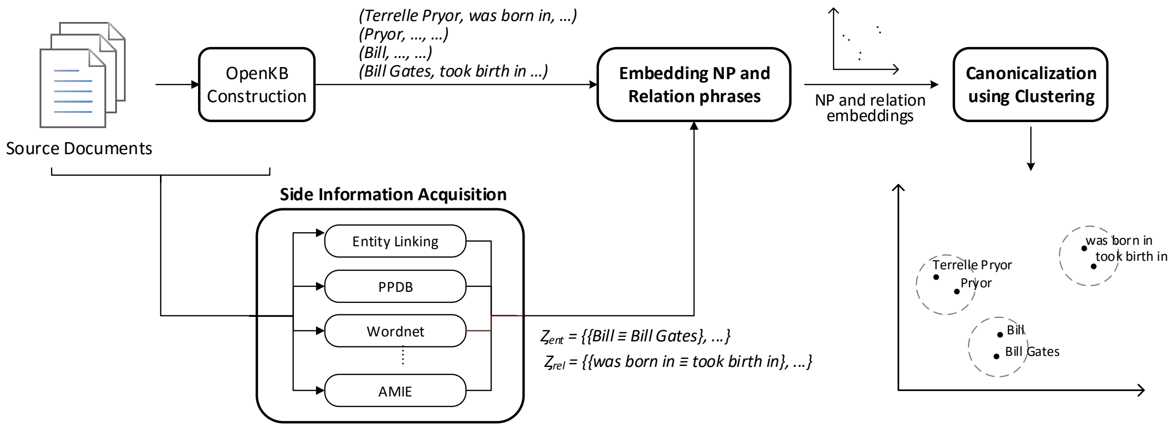
3.3 Proposed Approach: CESI

3.3.1 Overview

Overall architecture and dataflow of CESI is shown in Figure 3.1. The input to CESI is an un-canonicalized Open Knowledge Base (KB) with source information for each triple. The output is a list of canonicalized noun and relation phrases, which can be used to identify equivalent entities and relations or canonicalize the KB. CESI achieves this through its three step procedure:

1. **Side Information Acquisition:** The goal of this step is to gather various NP and relation phrase side information for each triple in the input by running several standard algorithms on the source text of the triples. More details can be found in Section 3.3.2.

2. **Embedding NP and Relation Phrases:** In this step, CESI learns specialized vector embeddings for all NPs and relation phrases in the input by making principled use of side information available from the previous step.



Source Documents
OpenKB Construction
(Terrelle Pryor, was born in, …)
(Pryor, …, …)
(Bill, …, …)
(Bill Gates, took birth in …)
Embedding NP and Relation phrases
NP and relation embeddings
Canonicalization using Clustering
Side Information Acquisition
Entity Linking
PPDB
Wordnet
AMIE
Zent = {{Bill ≡ Bill Gates}, …}
Zrel = {{was born in ≡ took birth in}, …}
Terrelle Pryor
Pryor
was born in
took birth in
Bill
Bill Gates

Figure 3.1: Overview of CESI. CESI first acquires side information of noun and relation phrases of Open KB triples. In the second step, it learns embeddings of these NPs and relation phrases while utilizing the side information obtained in previous step. In the third step, CESI performs clustering over the learned embeddings to canonicalize NP and relation phrases. Please see Section 3.3 for more details.

3. **Clustering Embeddings and Canonicalization:** Goal of this step is to cluster the NPs and relation phrases on the basis of their distance in the embedding space. Each cluster represents a specific entity or relation. Based on certain relevant heuristics, we assign a representative to each NP and relation phrase cluster.

Details of different steps of CESI are described next.

3.3.2 Side Information Acquisition

Noun and relation phrases in Open KBs often have relevant side information in the form of useful context in the documents from which the triples were extracted. Sometimes, such information may also be present in other related KBs. Previous Open KB canonicalization methods [61] ignored such available side information and performed canonicalization in isolation focusing only on the Open KB triples. CESI attempts to exploit such side information to further improve the performance on this problem. In CESI, we make use of five types of NP side information to get equivalence relations of the form $e_1 \equiv e_2$ between two entities e_1 and e_2. Similarly, relation phrase side information is used to derive relation equivalence, $r_1 \equiv r_2$. All equivalences are used as soft constraints in later steps of CESI (details in Section 3.3.3).

3.3.2.1 Noun Phrase side Information

In the present version of CESI, we make use of the following five types of NP side information:

1. **Entity Linking**: Given unstructured text, entity linking algorithms identify entity mentions and link them to Ontological KBs such as Wikipedia, Freebase etc. We make use of Stanford CoreNLP entity linker which is based on [175] for getting NP to Wikipedia entity linking. Roughly, in about 30% cases, we get this information for NPs. If two NPs are linked to the same Wikipedia entity, we assume them to be equivalent as per this information. For example, *US* and *America* can get linked to the same Wikipedia entity *United_States*.

2. **PPDB Information**: We make use of PPDB 2.0 [150], a large collection of paraphrases in English, for identifying equivalence relation among NPs. We first extracted high confidence paraphrases from the dataset while removing duplicates. Then, using union-find, we clustered all the phrases which are stated equivalent with high confidence and randomly assigned a representative to each cluster. Using an index created over the obtained clusters, we find cluster representative for each NP. If two NPs have the same cluster representative then they are considered to be equivalent. NPs not present in the dataset are skipped. This information helps us identifying equivalence between NPs such as *management* and *administration*.

3. **WordNet with Word-sense Disambiguation**: Using word-sense disambiguation [10] with Wordnet [127], we identify possible synsets for a given NP. If two NPs share a common synset, then they are marked as similar as per this side information. For example, *picture* and *image* can get linked to the same synset *visualize.v.01*.

4. **IDF of Overlapping Toke**: NPs sharing infrequent terms give a strong indication of them referring to the same entity. For example, it is very likely for *Warren Buffett* and *Buffett* to refer to the same person. In [61], IDF token overlap was found to be the most effective feature for canonicalization. We assign a score for every pair of NPs based on the standard IDF formula:

$$score_{idf}(n, n') = \frac{\sum_{x \in w(n) \cap w(n')} \log\left(1 + f(x)\right)^{-1}}{\sum_{x \in w(n) \cup w(n')} \log\left(1 + f(x)\right)^{-1}}$$

   Here, $w(\cdot)$ for a given NP returns the set of its terms, excluding stop words. $f(\cdot)$ returns the document frequency for a token.

5. **Morph Normalization**: We make use of multiple morphological normalization operations like tense removal, pluralization, capitalization and others as used in [54] for finding

out equivalent NPs. We show in Section 3.4.2.2 that this information helps in improving performance.

3.3.2.2 Relation Phrase Side Information

Similar to noun phrases, we make use of PPDB and WordNet side information for relation phrase canonicalization as well. Apart from these, we use the following two additional types of side information involving relation phrases.

1. **AMIE Information**: AMIE algorithm [62] tries to learn implication rules between two relations r and r' of the form $r \Rightarrow r'$. These rules are detected based on statistical rule mining [61]. It declares two relations r and r' to be equivalent if both $r \Rightarrow r'$ and $r' \Rightarrow r$ satisfy support and confidence thresholds. AMIE accepts a semi-canonicalized KB as input, i.e., a KB where NPs are already canonicalized. Since this is not the case with Open KBs, we first canonicalized NPs morphologically and then applied AMIE over the NP-canonicalized KB. We chose morphological normalization for this step as such normalization is available for all NPs, and also because we found this side information to be quite effective in large Open KBs.

2. **KBP Information**: Given unstructured text, Knowledge Base Population (KBP) systems detect relations between entities and link them to relations in standard KBs. For example, *"Obama was born in Honolulu"* contains *"was born in"* relation between *Obama* and *Honolulu*, which can be linked to *per:city_of_birth* relation in KBs. In CESI, we use Stanford KBP [180] to categorize relations. If two relations fall in the same category, then they are considered equivalent as per this information.

The given list can be further extended based on the availability of other side information. For the experiments in this chapter, we have used the above mentioned NP and relation phrase side information. Some of the equivalences derived from different side information might be erroneous, therefore, instead of using them as hard constraints, we try to use them as supplementary information as described in the next section. Even though side information might be available only for a small fraction of NPs and relation phrases, the hypothesis is that it will result in better overall canonicalization. We find this to be true, as shown in Section 3.4.2.

3.3.3 Embedding NP and Relation Phrases

For learning embeddings of NPs and relation phrases in a given Open KB, CESI optimizes HolE's [142] objective function along with terms for penalizing violation of equivalence conditions from the NP and relation phrase side information. Since the conditions from side information might be spurious, a factor ($\lambda_{\text{ent/rel},\theta}$) is multiplied with each term, which acts as a hyper-parameter and is tuned on a held out validation set. We also keep a constant (λ_{str}) with HolE objective function, to make selective use of structural information from KB for canonicalization. We choose HolE because it is one of the best performing KB embeddings techniques for tasks like link prediction in knowledge graphs. Since KBs store only true triples, we generate negative examples using local closed world heuristic [51]. To keep the rank of true triples higher than the non-existing ones, we use pairwise ranking loss function. The final objective function is described below.

$$
\begin{aligned}
\min_{\Theta} \; & \lambda_{str} \sum_{i \in D_+} \sum_{j \in D_-} \max(0, \gamma + \sigma(\eta_j) - \sigma(\eta_i)) \\
& + \sum_{\theta \in \mathscr{C}_{\text{ent}}} \frac{\lambda_{\text{ent},\theta}}{|\mathcal{Z}_{\text{ent},\theta}|} \sum_{v,v' \in \mathcal{Z}_{\text{ent},\theta}} \|e_v - e_{v'}\|^2 \\
& + \sum_{\phi \in \mathscr{C}_{\text{rel}}} \frac{\lambda_{\text{rel},\phi}}{|\mathcal{Z}_{\text{rel},\phi}|} \sum_{u,u' \in \mathcal{Z}_{\text{rel},\phi}} \|r_u - r_{u'}\|^2 \\
& + \lambda_{\text{reg}} \left(\sum_{v \in V} \|e_v\|^2 + \sum_{r \in R} \|e_r\|^2 \right).
\end{aligned}
$$

The objective function, consists of three main terms, along with one term for regularization. Optimization parameter, $\Theta = \{e_v\}_{v \in V} \cup \{r_u\}_{u \in R}$, is the set of all NP ($e_v$) and relation phrase ($r_u$) d-dimensional embeddings, where, V and R denote the set of all NPs and relation phrases in the input. In the first term, D_+, D_- specify the set of positive and negative examples and $\gamma > 0$ refers to the width of the margin [22]. Further, $\sigma(\cdot)$ denotes the logistic function and for a triple t_i (s, p, o), $\eta_i = r_p^T(e_s \star e_o)$, where $\star : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ is the circular correlation operator defined as follows.

$$
[a \star b]_k = \sum_{i=0}^{d-1} a_i b_{(k+i) \bmod d}.
$$

The first index of $(a \star b)$ measures the similarity between a and b, while other indices capture the interaction of features from a and b, in a particular order. Please refer to [142] for more details.

In the second and third terms, $\mathscr{C}_{\text{ent}}$ and $\mathscr{C}_{\text{rel}}$ are the collection of all types of NP and relation side information available from the previous step (Section 3.3.2), i.e., $\mathscr{C}_{\text{ent}} = \{\text{Entity Linking, PPDB, ..}\}$ and $\mathscr{C}_{\text{rel}} = \{\text{AMIE, KBP, ..}\}$. Further, $\lambda_{\text{ent},\theta}$ and $\lambda_{\text{rel},\phi}$ denote the constants associated with entity and relation side information. Their value is tuned using grid search on a held out validation set. The set of all equivalence conditions from a particular side information is denoted by $\mathcal{Z}_{\text{ent},\theta}$ and $\mathcal{Z}_{\text{rel},\phi}$. The rationale behind putting these terms is to allow inclusion of side information while learning embeddings, by enforcing two NPs or relations close together if they are equivalent as per the available side information. Since the side information is available for a fraction of NPs and relation phrases in the input, including these terms in the objective does not slow down the training of embeddings significantly.

The last term adds L2 regularization on the embeddings. All embeddings are initialized by averaging GloVe vectors [152]. We use mini-batch gradient descent for optimization.

3.3.4 Clustering Embeddings and Canonicalization

CESI clusters NPs and relation phrases by performing Hierarchical Agglomerative Clustering (HAC) using cosine similarity over the embeddings learned in the previous step (Section 3.3.3). HAC was preferred over other clustering methods because the number of clusters are not known beforehand. Complete linkage criterion is used for calculating the similarity between intermediate clusters as it gives smaller sized clusters, compared to single and average linkage criterion. This is more reasonable for canonicalization problem, where cluster sizes are expected to be small. The threshold value for HAC was chosen based on held out validation dataset.

The time complexity of HAC with complete linkage criterion is $O(n^2)$ [45]. For scaling up CESI to large knowledge graphs, one may go for modern variants of approximate Hierarchical clustering algorithms [89] at the cost of some loss in performance.

Finally, we decide a representative for each NP and relation phrase cluster. For each cluster, we compute a mean of all elements' embeddings weighted by the frequency of occurrence of each element in the input. NP or relation phrase which lies closest to the weighted cluster mean is chosen as the representative of the cluster.

3.4 Experiments

3.4.1 Experimental Setup

3.4.1.1 Datasets

Statistics of the three datasets used in the experiments of this chapter are summarized in Table 3.1. We present below brief summary of each dataset.

1. **Base and Ambiguous Datasets:** We obtained the Base and Ambiguous datasets from the authors of Galárraga et al. [61]. Base dataset was created by collecting triples containing 150 sampled Freebase entities that appear with at least two aliases in ReVerb

Datasets	# Gold Entities	#NPs	#Relations	#Triples
Base	150	290	3K	9K
Ambiguous	446	717	11K	37K
ReVerb45K	7.5K	15.5K	22K	45K

Table 3.1: Details of datasets used. ReVerb45K is the new dataset we propose in this chapter. Please see Section 3.4.1.1 for details.

Open KB. The same dataset was further enriched with mentions of homonym entities to create the Ambiguous dataset. Please see Galárraga et al. [61] for more details.

2. **ReVerb45K:** This is the new Open KB canonicalization dataset we propose in this work. ReVerb45K is a significantly extended version of the Ambiguous dataset, containing more than 20x NPs. ReVerb45K is constructed by intersecting information from the following three sources: ReVerb Open KB [54], Freebase entity linking information from [60], and Clueweb09 corpus [31]. Firstly, for every triple in ReVerb, we extracted the source text from Clueweb09 corpus from which the triple was generated. In this process, we rejected triples for which we could not find any source text. Then, based on the entity linking information from [60], we linked all subjects and objects of triples to their corresponding Freebase entities. If we could not find high confidence linking information for both subject and object in a triple, then it was rejected. Further, following the dataset construction procedure adopted by [61], we selected triples associated with all Freebase entities with at least two aliases occurring as subject in our dataset. Through these steps, we obtained 45K high-quality triples which we used for evaluation. We call this resulting dataset ReVerb45K.

   In contrast to Base and Ambiguous datasets, the number of entities, NPs and relation phrases in ReVerb45K are significantly larger. Please see Table 3.1 for a detailed comparison. This better mimics real-world KBs which tend to be sparse with very few edges per entity, as also observed by [22].

For getting test and validation set for each dataset, we randomly sampled 20% Freebase entities and called all the triples associated with them as validation set and rest was used as the test set.

3.4.1.2 Evaluation Metrics

Following [61], we use macro-, micro- and pairwise metrics for evaluating Open KB canonicalization methods. We briefly describe below these metrics for completeness. In all cases, C denotes the clusters produced by the algorithm to be evaluated, and E denotes the gold standard clusters. In all cases, F1 measure is given as the harmonic mean of precision and recall.

Macro: Macro precision (P_{macro}) is defined as the fraction of pure clusters in C, i.e., clusters in which all the NPs (or relations) are linked to the same gold entity (or relation). Macro recall (R_{macro}) is calculated like macro precision but with the roles of E and C interchanged.

$$\begin{aligned} P_{\text{macro}}(C, E) &= \frac{|\{c \in C : \exists e \in E : e \supseteq c\}|}{|C|} \\ R_{\text{macro}}(C, E) &= P_{\text{macro}}(E, C) \end{aligned}$$

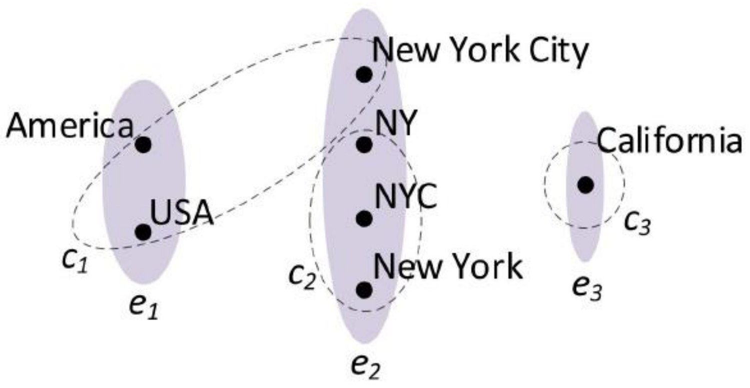
Micro: Micro precision (P_{micro}) is defined as the purity of C clusters [119] based on the assumption that the most frequent gold entity (or relation) in a cluster is correct. Micro recall (R_{micro}) is defined similarly as macro recall.

$$\begin{aligned} P_{\text{micro}}(C, E) &= \frac{1}{N} \sum_{c \in C} \max_{e \in E} |c \cap e| \\ R_{\text{micro}}(C, E) &= P_{\text{micro}}(E, C) \end{aligned}$$

Pairwise: Pairwise precision (P_{pair}) is measured as the ratio of the number of hits in C to the total possible pairs in C. Whereas, pairwise recall (R_{pair}) is the ratio of number of hits in C to all possible pairs in E. A pair of elements in a cluster in C produce a hit if they both refer to the same gold entity (or relation).

$$\begin{aligned} P_{\text{pair}}(C, E) &= \frac{\sum_{c \in C} |\{(v, v') \in e, \exists e \in E, \forall (v, v') \in c\}|}{\sum_{c \in C} {}^{|c|}C_2} \\ R_{\text{pair}}(C, E) &= \frac{\sum_{c \in C} |\{(v, v') \in e, \exists e \in E, \forall (v, v') \in c\}|}{\sum_{e \in E} {}^{|e|}C_2} \end{aligned}$$

Let us illustrate these metrics through a concrete NP canonicalization example shown in Figure 3.2. In this Figure, we can see that only c_2 and c_3 clusters in C are pure because they contain mentions of only one entity, and hence, $P_{\text{macro}} = \frac{2}{3}$. On the other hand, we have e_1 and e_3 as pure clusters if we interchange the roles of E and C. So, $R_{\text{macro}} = \frac{2}{3}$ in this case.



	Precision	Recall	F1
Macro	$\frac{2}{3}$	$\frac{2}{3}$	66.6
Micro	$\frac{6}{7}$	$\frac{6}{7}$	85.7
Pairwise	$\frac{4}{6}$	$\frac{4}{7}$	61.5

Figure 3.2: Top: Illustrative example for different evaluation metrics. e_i denotes actual clusters, whereas c_i denotes predicted clusters. Bottom: Metric results for the above example. Please see Section 3.4.1.2 for details.

For micro precision, we can see that *America*, *New York*, and *California* are the most frequent gold entities in C clusters. Hence, $P_{\text{micro}} = \frac{6}{7}$. Similarly, $R_{\text{micro}} = \frac{6}{7}$ in this case. For pairwise analysis, we need to first calculate the number of hits in C. In c_1 we have 3 possible pairs out of which only 1, (*America, USA*) is a hit as they belong to same gold cluster e_1. Similarly, we have 3 hits in c_2 and 0 hits in c_3. Hence, $P_{\text{pair}} = \frac{4}{6}$. To compute R_{pair}, we need total number of pairwise decisions in E, which is $1 + 6 + 0$, thus, $R_{\text{pair}} = \frac{4}{7}$. All the results are summarized in Table 3.2.

For evaluating NP canonicalization, we use Macro, Micro and Pairwise F1 score. However, in the case of relations, where gold labels are not available, we use macro, micro and pairwise precision values based on the scores given by human judges.

3.4.1.3 Methods Compared

Noun Phrase Canonicalization: For NP canonicalization, CESI has been compared against the following methods:

- **Morphological Normalization:** As used in [54], this involves applying simple normalization operations like removing tense, pluralization, capitalization etc. over NPs and relation phrases.

- **Paraphrase Database (PPDB):** Using PPDB 2.0 [150], we clustered two NPs together if they happened to share a common paraphrase. NPs which could not be found in PPDB are put into singleton clusters.

- **Entity Linking**: Since the problem of NP canonicalization is closely related to entity linking, we compare our method against Stanford CoreNLP Entity Linker [175]. Two NPs linked to the same entity are clustered together.

- **Galárraga-IDF** [**61**]: IDF Token Overlap was the best performing method proposed in [61] for NP canonicalization. In this method, IDF token similarity is defined between two NPs as in Section 3.3.2.1, and HAC is used to cluster the mentions.

- **Galárraga-StrSim** [**61**]: This method is similar to Galarraga-IDF, but with similarity metric being the Jaro-Winkler [204] string similarity measure.

- **Galárraga-Attr** [**61**]: Again, this method is similar to the Galarraga-IDF, except that Attribute Overlap is used as the similarity metric between two NPs in this case. Attribute for a NP n, is defined as the set of relation-NP pairs which co-occur with n in the input triples. Attribute overlap similarity between two NPs, is defined as the Jaccard coefficient of the set of attributes:
$$f_{\text{attr}}(n, n') = \frac{|A \cap A'|}{|A \cup A'|}$$
where, A and A' denote the set of attributes associated with n and n'.

  Since canonicalization methods using above similarity measures were found to be most effective in [61], even outperforming Machine Learning-based alternatives, we consider these three baselines as representatives of state-of-the-art in Open KB canonicalization.

- **GloVe**: In this scheme, each NP and relation phrase is represented by a 300 dimensional GloVe embedding [152] trained on Wikipedia 2014 and Gigaword 5 [148] datasets with 400k vocabulary size. Word vectors were averaged together to get embeddings for multi-word phrases. These GloVE embeddings were then clustered for final canonicalization.

- **HolE**: In this method, embeddings of NPs and relation phrases in an Open KB are obtained by applying HolE [142] over the Open KB. These embeddings are then clustered to obtain the final canonicalized groupings. Based on the initialization of embeddings, we differentiate between **HolE(Random)** and **HolE(GloVe)**.

- **CESI**: This is the method proposed in this chapter, please see Section 3.3 for more details.

Hyper-parameters: Following [61], we used Hierarchical Agglomerative Clustering (HAC) as the default clustering method across all methods (wherever necessary). For all methods, grid search over the hyperparameter space was performed, and results for the best performing setting are reported. This process was repeated for each dataset.

3.4.1.4 Relation Phrase Canonicalization

AMIE [62] was found to be effective for relation phrase canonicalization in [61]. We thus consider AMIE[1] as the state-of-the-art baseline for relation phrase canonicalization and compare against CESI. We note that AMIE requires NPs of the input Open KB to be already canonicalized. In all our evaluation datasets, we already have *gold* NP canonicalization available. We provide this gold NP canonicalization information as input to AMIE. Please note that CESI doesn't require such pre-canonicalized NP as input, as it performs *joint* NP and relation phrase canonicalization. Moreover, providing gold NP canonicalization information to AMIE puts CESI at a disadvantage. We decided to pursue this choice anyways in the interest of stricter evaluation. However, in spite of starting from this disadvantageous position, CESI significantly outperforms AMIE in relation phrase canonicalization, as we will see in Section 3.4.2.1.

For evaluating performance of both algorithms, we randomly sampled 25 non-singleton relation clusters for each of the three datasets and gave them to five different human evaluators[2] for assigning scores to each cluster. The setting was kept blind, i.e., identity of the algorithm producing a cluster was not known to the evaluators. Based on the average of evaluation scores, precision values were calculated. Only non-singleton clusters were sampled, as singleton clusters will always give a precision of one.

3.4.2 Results

In this section, we evaluate the following questions.

Q1. Is CESI effective in Open KB canonicalization? (Section 3.4.2.1)

Q2. What is the effect of side information in CESI's performance? (Section 3.4.2.2)

Q3. Does addition of entity linking side information degrade CESI's ability to canonicalize unlinked NPs (i.e., NPs missed by the entity linker)? (Section 3.4.2.3)

Finally, in Section 3.4.2.4, we present qualitative examples and discussions.

[1]We use support and confidence values of 2 and 0.2 for all the experiments in this chapter.
[2]Authors did not participate in this evaluation.

Method	Base Dataset			Ambiguous Dataset			ReVerb45K			
	Macro	Micro	Pair.	Macro	Micro	Pair.	Macro	Micro	Pair.	Row Average
Morph Norm	58.3	88.3	83.5	49.1	57.2	70.9	1.4	77.7	75.1	62.3
PPDB	42.4	46.9	32.2	37.3	60.2	69.3	46.0	45.4	64.2	49.3
EntLinker	54.9	65.1	75.2	49.7	83.2	68.8	62.8	81.8	80.4	69.1
Galárraga-StrSim	88.2	96.5	97.7	66.6	85.3	82.2	69.9	51.7	0.5	70.9
Galárraga-IDF	94.8	97.9	98.3	67.9	82.9	79.3	71.6	50.8	0.5	71.5
Galárraga-Attr	76.1	51.4	18.1	**82.9**	27.7	8.4	**75.1**	20.1	0.2	40.0
GloVe	95.7	97.2	91.1	65.9	89.9	90.1	56.5	82.9	75.3	82.7
HolE (Random)	69.5	91.3	86.6	53.3	85.0	75.1	5.4	74.6	50.9	65.7
HolE (GloVe)	75.2	93.6	89.3	53.9	85.4	76.7	33.5	75.8	51.0	70.4
CESI	**98.2**	**99.8**	**99.9**	66.2	**92.4**	**91.9**	62.7	**84.4**	**81.9**	**86.3**

Table 3.2: NP Canonicalization Results. CESI outperforms all other methods across datasets (Best in 7 out of 9 cases. Section 3.4.2.1)

3.4.2.1 Evaluating Effectiveness of CESI in Open KB Canonicalization

Noun Phrase Canonicalization: Results for NP canonicalization are summarized in Table 3.2. Overall, we find that CESI performs well consistently across the datasets. Morphological Normalization failed to give competitive performance in presence of homonymy. PPDB, in spite of being a vast reservoir of paraphrases, lacks information about real-world entities like people, places etc. Therefore, its performance remained weak throughout all datasets. Entity linking methods make use of contextual information from source text of each triple to link a NP to a KB entity. But their performance is limited because they are restricted by the entities in KB.

String similarity also gave decent performance in most cases but since they solely rely on surface form of NPs, they are bound to fail with NPs having dissimilar mentions.

Methods such as Galárraga-IDF, Galárraga-StrSim, and Galárraga-Attr performed poorly on ReVerb45K. Although, their performance is considerably better on the other two datasets. This is because of the fact that in contrast to Base and Ambiguous datasets, ReVerb45K has considerably large number of entities and comparatively fewer triples (Table 3.1). Galárraga-IDF token overlap is more likely to put two NPs together if they share an uncommon token, i.e., one with high IDF value. Hence, accuracy of the method relies heavily on the quality of document frequency estimates which may be quite misleading when we have smaller number of triples. Similar is the case with Galárraga-Attr which decides similarity of NPs based on the set of shared attributes. Since, attributes for a NP is defined as a set of relation-NP pairs occurring with it across all triples, sparse data also results in poor performance for this method.

GloVe captures semantics of NPs and unlike string similarity it doesn't rely on the surface form of NPs. Therefore, its performance has been substantial across all the datasets. HolE

	Macro Precision	Micro Precision	Pairwise Precision	Induced Relation Clusters
	Base Dataset			
AMIE	42.8	63.6	43.0	7
CESI	**88.0**	**93.1**	**88.1**	**210**
	Ambiguous Dataset			
AMIE	55.8	64.6	23.4	46
CESI	**76.0**	**91.9**	**80.9**	**952**
	ReVerb45K			
AMIE	69.3	84.2	66.2	51
CESI	**77.3**	**87.8**	**72.6**	**2116**

Table 3.3: Relation canonicalization results. Compared to AMIE, CESI canonicalizes more number of relation phrases at higher precision. Please see Section 3.4.2.1 for details.

captures structural information from the given triples and uses it for learning embeddings. Through our experiments, we can see that solely structural information from KB is quite effective for NP canonicalization. CESI performs the best across the datasets in 7 out of the 9 settings, as it incorporates the strength of all the listed methods. The superior performance of CESI compared to HolE clearly indicates that the side information is indeed helpful for canonicalization task. Results of GloVe, HolE and CESI suggest that embeddings based method are much more effective for Open KB canonicalization.

Relation Phrase Canonicalization Results for relation phrase canonicalization are presented in Table 3.3. For all experiments, in spite of using quite low values for minimum support and confidence, AMIE was unable to induce any reasonable number of non-singleton clusters (e.g., only 51 clusters out of the 22K relation phrases in the ReVerb45K dataset). For relation canonicalization experiments, AMIE was evaluated on gold NP canonicalized data as the algorithm requires NPs to be already canonicalized. CESI, on the other hand, was tested on all the datasets without making use of gold NP canonicalization information.

Based on the results in Table 3.3, it is quite evident that AMIE induces too few relation clusters to be of value in practical settings. On the other hand, CESI consistently performs well across all the datasets and induces significantly larger number of clusters.

	Macro F1	Micro F1	Pairwise F1
CESI	81.7	87.6	81.5
CESI w/o EL	81.3	87.3	80.7

Table 3.4: CESI's performance in canonicalizing unlinked NPs, with and without Entity Linking (EL) side information, in the ReVerb45K dataset. We observe that CESI does not overfit to EL side information, and thereby helps prevent performance degradation in unlinked NP canonicalization (in fact it even helps a little). Please see Section 3.4.2.3 for details.

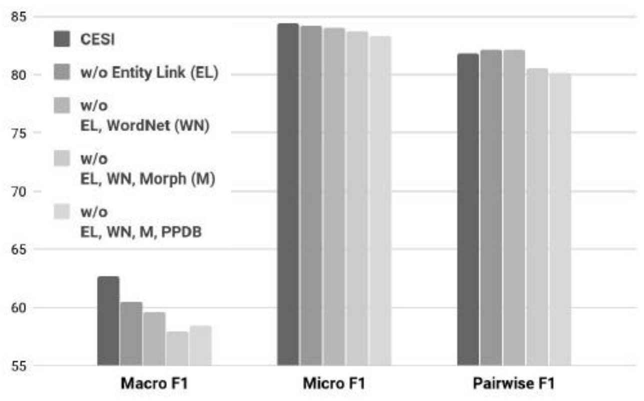
3.4.2.2 Effect of Side Information in CESI

In this section, we evaluate the effect of various side information in CESI's performance. For this, we evaluated the performances of various versions of CESI, each one of them obtained by ablating increasing amounts of side information from the full CESI model. Experimental results comparing these ablated versions on the ReVerb45K are presented in Figure 3.3a. From this figure, we observe that while macro performance benefits most from different forms of side information, micro and pairwise performance also show increased performance in the presence of various side information. This validates one of the central thesis of this chapter: side information, along with embeddings, can result in improved Open KB canonicalization.

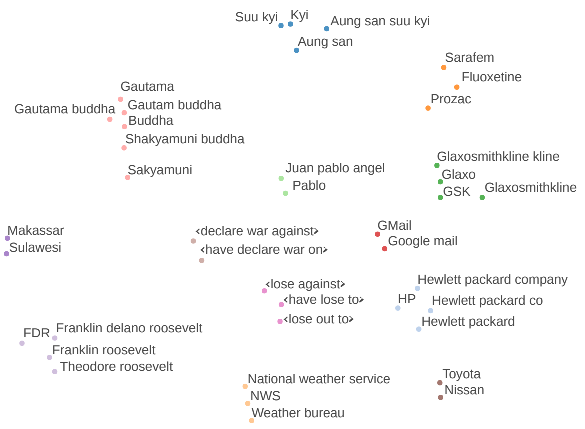
3.4.2.3 Effect of Entity Linking Side Information on Unlinked NP

From experiments in Section 3.4.2.2, we find that Entity Linking (EL) side information (see Section 3.3.2.1) is one of the most useful side information that CESI exploits. However, such side information is not available in case of unlinked NPs, i.e., NPs which were not linked by the entity linker. So, this naturally raises the following question: does CESI overfit to the EL side information and ignore the unlinked NPs, thereby resulting in poor canonicalization of such unlinked NPs?

In order to evaluate this question, we compared CESI's performance on unlinked NPs in the ReVerb45K dataset, with and without EL side information. We note that triples involving unlinked NPs constitute about 25% of the entire dataset. Results are presented in Table 3.4. From this table, we observe that CESI doesn't overfit to EL side information, and it selectively uses such information when appropriate (i.e., for linked NPs). Because of this robust nature, presence of EL side information in CESI doesn't have an adverse effect on the unlinked NPs, in fact there is a small gain in performance.



(a) Performance comparison of various side information-ablated versions of CESI for NP canonicalization in the ReVerb45K dataset. Overall, side information helps CESI improve performance. Please see Section 3.4.2.2 for details.



(b) t-SNE visualization of NP and relation phrase (marked in '$< \cdots >$') embeddings learned by CESI for ReVerb45K dataset. We observe that CESI is able to induce non-trivial canonical clusters. More details in Sec. 3.4.2.4.

3.4.2.4 Qualitative Evaluation

Figure 3.3b shows some of the NP and relation phrase clusters detected by CESI in ReVerb45K dataset. These results highlight the efficacy of algorithm in canonicalizing non-trivial NPs and relation phrases. The figure shows t-SNE [190] visualization of NP and relation phrase (marked in '$< \cdots >$') embeddings for a few examples. We can see that the learned embeddings are actually able to capture equivalence of NPs and relation phrases. The algorithm is able to correctly embed *Prozac*, *Sarafem* and *Fluoxetine* together (different names of the same drug), despite their having completely different surface forms.

Figure 3.3b also highlights the failures of CESI. For example, *Toyota* and *Nissan* have been embedded together although the two being different companies. Another case is with *Pablo* and *Juan Pablo Angel*, which refer to different entities. The latter case can be avoided by keeping track of the source domain type information of each NP for disambiguation. In this if we know that *Juan Pablo Angel* has come from *SPORTS* domain, whereas *Pablo* has come from a different domain then we can avoid putting them together. We tried using DMOZ [173] dataset, which provide mapping from URL domain to their categories, for handling such errors. But, because of poor coverage of URLs in DMOZ dataset, we couldn't get significant improvement in canonicalization results. We leave this as a future work.

3.5 Conclusion

Canonicalizing Open Knowledge Bases (KBs) is an important but underexplored problem. In this chapter, we proposed CESI, a novel method for canonicalizing Open KBs using learned embeddings and side information. CESI solves a joint objective to learn noun and relation phrase embeddings, while utilizing relevant side information in a principled manner. These learned embeddings are then clustered together to obtain canonicalized noun and relation phrase clusters. We also propose ReVerb45K, a new and larger dataset for Open KB canonicalization. Through extensive experiments on this and other real-world datasets, we demonstrate CESI's effectiveness over state-of-the-art baselines. CESI's source code and all data used in this work are publicly available.

Chapter 4

Improving Distantly-Supervised Relation Extraction using Graph Convolutional Networks and Side Information

4.1 Introduction

In this chapter, we present another solution to address sparsity problem in knowledge graph and also demonstrate how relevant side information can be effectively utilized for achieving state-of-the-art results. The construction of large-scale Knowledge Bases (KBs) like Freebase [21] and Wikidata [199] has proven to be useful in many natural language processing (NLP) tasks like question-answering, web search, etc. However, these KBs have limited coverage. Relation Extraction (RE) attempts to fill this gap by extracting semantic relationships between entity pairs from plain text. This task can be modeled as a simple classification problem after the entity pairs are specified. Formally, given an entity pair (e_1,e_2) from the KB and an entity annotated sentence (or instance), we aim to predict the relation r, from a predefined relation set, that exists between e_1 and e_2. If no relation exists, we simply label it NA.

Most supervised relation extraction methods require large labeled training data which is expensive to construct. Distant Supervision (DS) [128] helps with the construction of this dataset automatically, under the assumption that if two entities have a relationship in a KB, then all sentences mentioning those entities express the same relation. While this approach works well in generating large amounts of training instances, the DS assumption does not hold in all cases. Riedel et al. [161], Hoffmann et al. [75], Surdeanu et al. [180] propose multi-instance learning to relax this assumption. However, they use manually defined NLP tools based features which can be sub-optimal.

Recently, neural models have demonstrated promising performance on RE. Zeng et al. [219, 220] employ Convolutional Neural Networks (CNN) to learn representations of instances. For alleviating noise in distant supervised datasets, attention has been utilized by [109, 77]. Syntactic information from dependency parses has been used by [128, 71] for capturing long-range dependencies between tokens. Recently proposed Graph Convolution Networks (GCN) [46] have been effectively employed for encoding this information [121, 15]. However, all the above models rely only on the noisy instances from distant supervision for RE.

Relevant side information can be effective for improving RE. For instance, in the sentence, *Microsoft was started by Bill Gates.*, the type information of *Bill Gates (person)* and *Microsoft (organization)* can be helpful in predicting the correct relation *founderOfCompany*. This is because every relation constrains the type of its target entities. Similarly, relation phrase *"was started by"* extracted using Open Information Extraction (Open IE) methods can be useful, given that the aliases of relation *founderOfCompany*, e.g., *founded, co-founded, etc.*, are available. KBs used for DS readily provide such information which has not been completely exploited by current models.

In this chapter, we propose RESIDE, a novel distant supervised relation extraction method which utilizes additional supervision from KB through its neural network based architecture. RESIDE makes principled use of entity type and relation alias information from KBs, to impose soft constraints while predicting the relation. It uses encoded syntactic information obtained from Graph Convolution Networks (GCN), along with embedded side information, to improve neural relation extraction. Our contributions can be summarized as follows:

- We propose RESIDE, a novel neural method which utilizes additional supervision from KB in a principled manner for improving distant supervised RE.
- RESIDE uses Graph Convolution Networks (GCN) for modeling syntactic information and has been shown to perform competitively even with limited side information.
- Through extensive experiments on benchmark datasets, we demonstrate RESIDE's effectiveness over state-of-the-art baselines.

RESIDE's source code and datasets used in the chapter are available at `http://github.com/malllabiisc/RESIDE`.

4.2 Related Work

Distant supervision: Relation extraction is the task of identifying the relationship between two entity mentions in a sentence. In supervised paradigm, the task is considered as a multi-class classification problem but suffers from lack of large labeled training data. To address this limitation, [128] propose distant supervision (DS) assumption for creating large datasets, by heuristically aligning text to a given Knowledge Base (KB). As this assumption does not always hold true, some of the sentences might be wrongly labeled. To alleviate this shortcoming, Riedel et al. [161] relax distant supervision for multi-instance single-label learning. Subsequently, for handling overlapping relations between entities [75, 180] propose multi-instance multi-label learning paradigm.

Neural Relation Extraction: The performance of the above methods strongly rely on the quality of hand engineered features. Zeng et al. [219] propose an end-to-end CNN based method which could automatically capture relevant lexical and sentence level features. This method is further improved through piecewise max-pooling by [220]. Lin et al. [109], Nagarajan et al. [136] use attention [7] for learning from multiple valid sentences. We also make use of attention for learning sentence and bag representations.

Dependency tree based features have been found to be relevant for relation extraction [128]. He et al. [71] use them for getting promising results through a recursive tree-GRU based model. In RESIDE, we make use of recently proposed Graph Convolution Networks [46, 88], which have been found to be quite effective for modelling syntactic information [121, 139, 191].

Side Information in RE: Entity description from KB has been utilized for RE [78], but such information is not available for all entities. Type information of entities has been used by Ling and Weld [110], Liu et al. [112] as features in their model. Yaghoobzadeh et al. [209] also attempt to mitigate noise in DS through their joint entity typing and relation extraction model. However, KBs like Freebase provide reliable type information which could be directly utilized. In our work, we make principled use of entity type and relation alias information obtained from KB. We also use unsupervised Open Information Extraction (Open IE) methods [123, 4], which automatically discover possible relations without the need of any predefined ontology, which is used as a side information as defined in Section 4.3.3.

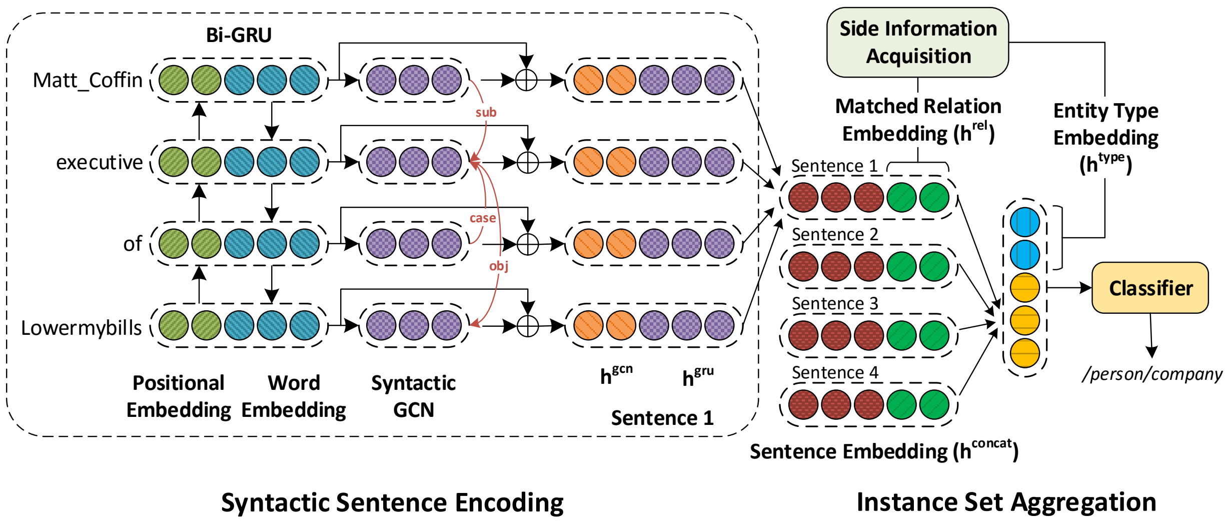


Figure 4.1: Overview of RESIDE. RESIDE first encodes each sentence in the bag by concatenating embeddings (denoted by $\oplus$) from Bi-GRU and Syntactic GCN for each token, followed by word attention (denoted by arrows). Then, sentence embedding is concatenated with relation alias information, which comes from the Side Information Acquisition Section (Figure 4.2), before computing attention over sentences. Finally, bag representation with entity type information is fed to a softmax classifier. Please see Section 4.3 for more details.

4.3 Proposed Method: RESIDE

4.3.1 Overview

In multi-instance learning paradigm, we are given a bag of sentences (or instances) $\{s_1, s_2, ...s_n\}$ for a given entity pair, the task is to predict the relation between them. RESIDE consists of three components for learning a representation of a given bag, which is fed to a softmax classifier. We briefly present the components of RESIDE below. Each component will be described in detail in the subsequent sections. The overall architecture of RESIDE is shown in Figure 4.1.

1. **Syntactic Sentence Encoding:** RESIDE uses a Bi-GRU over the concatenated positional and word embedding for encoding the local context of each token. For capturing long-range dependencies, GCN over dependency tree is employed and its encoding is appended to the representation of each token. Finally, attention over tokens is used to subdue irrelevant tokens and get an embedding for the entire sentence. More details in Section 4.3.2.
2. **Side Information Acquisition:** In this module, we use additional supervision from KBs and utilize Open IE methods for getting relevant side information. This information is later utilized by the model as described in Section 4.3.3.

3. **Instance Set Aggregation:** In this part, sentence representation from syntactic sentence encoder is concatenated with the *matched relation embedding* obtained from the previous step. Then, using attention over sentences, a representation for the entire bag is learned. This is then concatenated with *entity type embedding* before feeding into the softmax classifier for relation prediction. Please refer to Section 4.3.4 for more details.

Below, we provide the detailed description of the components of RESIDE.

4.3.2 Syntactic Sentence Encoding

For each sentence in the bag s_i with m tokens $\{w_1, w_2, ...w_m\}$, we first represent each token by k-dimensional GloVe embedding [152]. For incorporating relative position of tokens with respect to target entities, we use p-dimensional position embeddings, as used by [219]. The combined token embeddings are stacked together to get the sentence representation $\mathcal{H} \in \mathbb{R}^{m\times(k+2p)}$. Then, using Bi-GRU [38] over $\mathcal{H}$, we get the new sentence representation $\mathcal{H}^{gru} \in \mathbb{R}^{m\times d_{gru}}$, where d_{gru} is the hidden state dimension. Bi-GRUs have been found to be quite effective in encoding the context of tokens in several tasks [181, 67].

Although Bi-GRU is capable of capturing local context, it fails to capture long-range dependencies which can be captured through dependency edges. Prior works [128, 71] have exploited features from syntactic dependency trees for improving relation extraction. Motivated by their work, we employ Syntactic Graph Convolution Networks for encoding this information. For a given sentence, we generate its dependency tree using Stanford CoreNLP [120]. We then run GCN over the dependency graph and use Equation 2.9 for updating the embeddings, taking $\mathcal{H}^{gru}$ as the input. Since dependency graph has 55 different edge labels, incorporating all of them over-parameterizes the model significantly. Therefore, following [121, 139, 191] we use only three edge labels based on the direction of the edge $\{$*forward* $(\rightarrow)$, *backward* $(\leftarrow)$, *self-loop* $(\top)\}$. We define the new edge label L_{uv} for an edge (u, v, l_{uv}) as follows:

$$L_{uv} = \begin{cases} \rightarrow & \text{if edge exists in dependency parse} \\ \leftarrow & \text{if edge is an inverse edge} \\ \top & \text{if edge is a self-loop} \end{cases}$$

For each token w_i, GCN embedding $h^{gcn}_{i_{k+1}} \in \mathbb{R}^{d_{gcn}}$ after k^{th} layer is defined as:

$$h^{gcn}_{i_{k+1}} = f\left(\sum_{u\in\mathcal{N}(i)} g^k_{iu} \times \left(W^k_{L_{iu}} h^{gcn}_{u_k} + b^k_{L_{iu}}\right)\right).$$

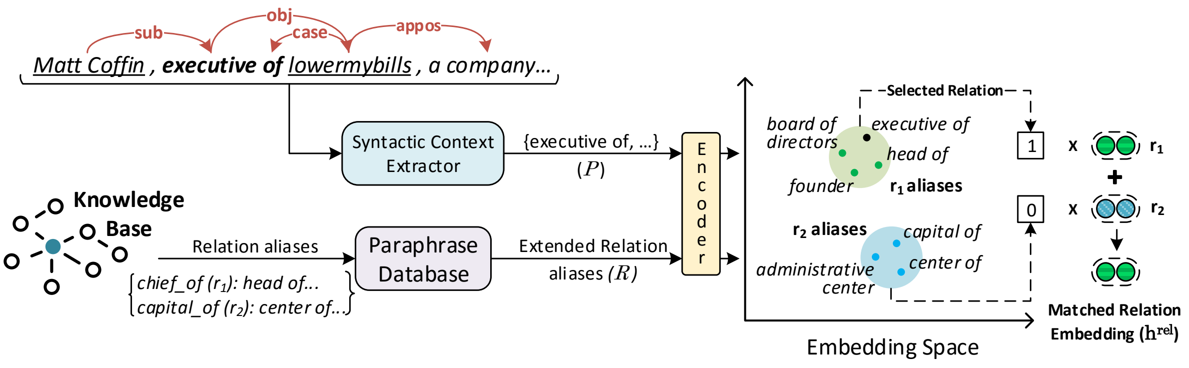


Figure 4.2: Relation alias side information extraction for a given sentence. First, Syntactic Context Extractor identifies relevant relation phrases $\mathcal{P}$ between target entities. They are then matched in the embedding space with the extended set of relation aliases $\mathcal{R}$ from KB. Finally, the relation embedding corresponding to the closest alias is taken as relation alias information. Please refer Section 4.3.3.

Here, g_{iu}^k denotes edgewise gating as defined in Equation 2.9 and L_{iu} refers to the edge label defined above. We use ReLU as activation function f, throughout our experiments. The syntactic graph encoding from GCN is appended to Bi-GRU output to get the final token representation, h_i^{concat} as $[h_i^{gru}; h_{i^{k+1}}^{gcn}]$. Since, not all tokens are equally relevant for RE task, we calculate the degree of relevance of each token using attention as used in [77]. For token w_i in the sentence, attention weight α_i is calculated as:

$$\alpha_i = \frac{\exp(u_i)}{\sum_{j=1}^{m} \exp(u_j)} \quad \text{where, } u_i = h_i^{concat} \cdot r.$$

where r is a random query vector and u_i is the relevance score assigned to each token. Attention values $\{\alpha_i\}$ are calculated by taking softmax over $\{u_i\}$. The representation of a sentence is given as a weighted sum of its tokens, $s = \sum_{j=1}^{m} \alpha_i h_i^{concat}$.

4.3.3 Side Information Acquisition

Relevant side information has been found to improve performance on several tasks [110, 192]. In distant supervision based relation extraction, since the entities are from a KB, knowledge about them can be utilized to improve relation extraction. Moreover, several unsupervised relation extraction methods (Open IE) [4, 123] allow extracting relation phrases between target entities without any predefined ontology and thus can be used to obtain relevant side information. In RESIDE, we employ Open IE methods and additional supervision from KB for improving neural relation extraction.

Relation Alias Side Information

RESIDE uses Stanford Open IE [4] for extracting relation phrases between target entities, which we denote by $\mathcal{P}$. As shown in Figure 4.2, for the sentence *Matt Coffin, executive of lowermybills, a company..*, Open IE methods extract *"executive of"* between *Matt Coffin* and *lowermybills*. Further, we extend $\mathcal{P}$ by including tokens at one hop distance in dependency path from target entities. Such features from dependency parse have been exploited in the past by [128, 71]. The degree of match between the extracted phrases in $\mathcal{P}$ and aliases of a relation can give important clues about the relevance of that relation for the sentence. Several KBs like Wikidata provide such relation aliases, which can be readily exploited. In RESIDE, we further expand the relation alias set using Paraphrase database (PPDB) [150]. We note that even for cases when aliases for relations are not available, providing only the names of relations give competitive performance. We shall explore this point further in Section 4.4.2.3.

For matching $\mathcal{P}$ with the PPDB expanded relation alias set $\mathcal{R}$, we project both in a d-dimensional space using GloVe embeddings [152]. Projecting phrases using word embeddings helps to further expand these sets, as semantically similar words are closer in embedding space [125, 152]. Then, for each phrase $p \in \mathcal{P}$, we calculate its cosine distance from all relation aliases in $\mathcal{R}$ and take the relation corresponding to the closest relation alias as a matched relation for the sentence. We use a threshold on cosine distance to remove noisy aliases. In RESIDE, we define a k_r-dimensional embedding for each relation which we call as *matched relation embedding* (h^{rel}). For a given sentence, h^{rel} is concatenated with its representation s, obtained from syntactic sentence encoder (Section 4.3.2) as shown in Figure 4.1. For sentences with $|\mathcal{P}| > 1$, we might get multiple matched relations. In such cases, we take the average of their embeddings. We hypothesize that this helps in improving the performance and find it to be true as shown in Section 4.4.2.

Entity Type Side Information

Type information of target entities has been shown to give promising results on relation extraction [110, 209]. Every relation puts some constraint on the type of entities which can be its subject and object. For example, the relation *person/place_of_birth* can only occur between a *person* and a *location*. Sentences in distance supervision are based on entities in KBs, where the type information is readily available.

In RESIDE, we use types defined by FIGER [110] for entities in Freebase. For each type, we define a k_t-dimensional embedding which we call as *entity type embedding* (h^{type}). For cases when an entity has multiple types in different contexts, for instance, *Paris* may have types *government* and *location*, we take the average over the embeddings of each type. We

Datasets	Split	# Sentences	# Entity-pairs
Riedel (# Relations: 53)	Train	455,771	233,064
	Valid	114,317	58,635
	Test	172,448	96,678
GDS (# Relations: 5)	Train	11,297	6,498
	Valid	1,864	1,082
	Test	5,663	3,247

Table 4.1: Details of datasets used. Please see Section 4.4.1.1 for more details.

concatenate the *entity type embedding* of target entities to the final bag representation before using it for relation classification. To avoid over-parameterization, instead of using all fine-grained 112 entity types, we use 38 coarse types which form the first hierarchy of FIGER types.

4.3.4 Instance Set Aggregation

For utilizing all valid sentences, following [109, 77], we use attention over sentences to obtain a representation for the entire bag. Instead of directly using the sentence representation s_i from Section 4.3.2, we concatenate the embedding of each sentence with *matched relation embedding* h_i^{rel} as obtained from Section 4.3.3. The attention score α_i for i^{th} sentence is formulated as:

$$\alpha_i = \frac{\exp(\hat{s}_i \cdot q)}{\sum_{j=1}^{n} \exp(\hat{s}_j \cdot q)} \quad \text{where, } \hat{s}_i = [s_i; h_i^{rel}].$$

here q denotes a random query vector. The bag representation $\mathcal{B}$, which is the weighted sum of its sentences, is then concatenated with the *entity type embeddings* of the subject (h_{sub}^{type}) and object (h_{obj}^{type}) from Section 4.3.3 to obtain $\hat{\mathcal{B}}$.

$$\hat{\mathcal{B}} = [\mathcal{B}; h_{sub}^{type}; h_{obj}^{type}] \quad \text{where, } \mathcal{B} = \sum_{i=1}^{n} \alpha_i \hat{s}_i.$$

Finally, $\hat{\mathcal{B}}$ is fed to a softmax classifier to get the probability distribution over the relations.

$$p(y) = \text{Softmax}(W \cdot \hat{\mathcal{B}} + b).$$

4.4 Experiments

4.4.1 Experimental Setup

4.4.1.1 Datasets

In our experiments, we evaluate the models on Riedel and Google Distant Supervision (GDS) dataset. Statistics of the datasets is summarized in Table 4.1. Below we described each in detail.

1. **Riedel:** The dataset is developed by [161] by aligning Freebase relations with New York Times (NYT) corpus, where sentences from the year 2005-2006 are used for creating the training set and from the year 2007 for the test set. The entity mentions are annotated using Stanford NER [57] and are linked to Freebase. The dataset has been widely used for RE by [75, 180] and more recently by [109, 56, 71].

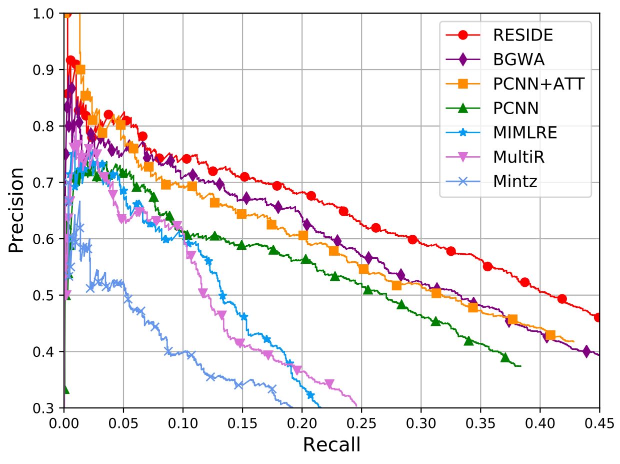
2. **GIDS:** Jat et al. [77] created Google IISc Distant Supervision (GIDS) dataset by extending the Google relation extraction corpus[1] with additional instances for each entity pair. The dataset assures that the at-least-one assumption of multi-instance learning, holds. This makes automatic evaluation more reliable and thus removes the need for manual verification.

4.4.1.2 Baselines

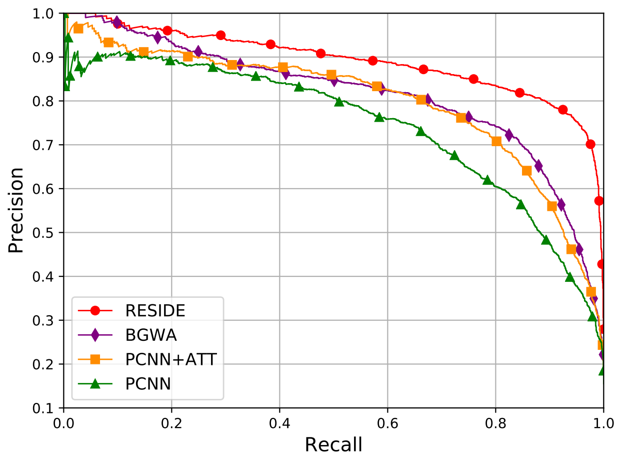
For evaluating RESIDE, we compare against the following baselines:

- **Mintz:** Multi-class logistic regression model proposed by [128] for distant supervision paradigm.
- **MultiR:** Probabilistic graphical model for multi instance learning by [75]
- **MIMLRE:** A graphical model which jointly models multiple instances and multiple labels. More details in [180].
- **PCNN:** A CNN based relation extraction model by [220] which uses piecewise max-pooling for sentence representation.
- **PCNN+ATT:** A piecewise max-pooling over CNN based model which is used by [109] to get sentence representation followed by attention over sentences.
- **BGWA:** Bi-GRU based relation extraction model with word and sentence level attention [77].
- **RESIDE:** The method proposed in this chapter, please refer Section 4.3 for more details.

[1] https://research.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html



(a) Riedel dataset



(b) GIDS dataset

Figure 4.3: Comparison of Precision-recall curve. RESIDE achieves higher precision over the entire range of recall than all the baselines on both datasets. Please refer Section 4.4.2.1 for more details.

4.4.1.3 Evaluation Criteria

Following the prior works [109, 56], we evaluate the models using held-out evaluation scheme. This is done by comparing the relations discovered from test articles with those in Freebase. We evaluate the performance of models with Precision-Recall curve and top-N precision (P@N) metric in our experiments.

4.4.2 Results

In this section we attempt to answer the following questions:

Q1. Is RESIDE more effective than existing approaches for distant supervised RE? (4.4.2.1)

Q2. What is the effect of ablating different components on RESIDE's performance? (4.4.2.2)

Q3. How is the performance affected in the absence of relation alias information? (4.4.2.3)

4.4.2.1 Performance Comparison

For evaluating the effectiveness of our proposed method, RESIDE, we compare it against the baselines stated in Section 4.4.1.2. We use only the neural baselines on GDS dataset. The Precision-Recall curves on Riedel and GDS are presented in Figure 4.3. Overall, we find that RESIDE achieves higher precision over the entire recall range on both the datasets. All the non-neural baselines could not perform well as the features used by them are mostly derived from NLP tools which can be erroneous. RESIDE outperforms PCNN+ATT and BGWA which

	One			Two			All		
	P@100	P@200	P@300	P@100	P@200	P@300	P@100	P@200	P@300
PCNN	73.3	64.8	56.8	70.3	67.2	63.1	72.3	69.7	64.1
PCNN+ATT	73.3	69.2	60.8	77.2	71.6	66.1	76.2	73.1	67.4
BGWA	78.0	71.0	63.3	81.0	73.0	64.0	82.0	75.0	72.0
RESIDE	**80.0**	**75.5**	**69.3**	**83.0**	**73.5**	**70.6**	**84.0**	**78.5**	**75.6**

Table 4.2: P@N for relation extraction using variable number of sentences in bags (with more than one sentence) in Riedel dataset. Here, One, Two and All represents the number of sentences randomly selected from a bag. RESIDE attains improved precision in all settings. More details in Section 4.4.2.1

indicates that incorporating side information helps in improving the performance of the model. The higher performance of BGWA and PCNN+ATT over PCNN shows that attention helps in distant supervised RE. Following [109, 111], we also evaluate our method with different number of sentences. Results summarized in Table 4.2, show the improved precision of RESIDE in all test settings, as compared to the neural baselines, which demonstrates the efficacy of our model.

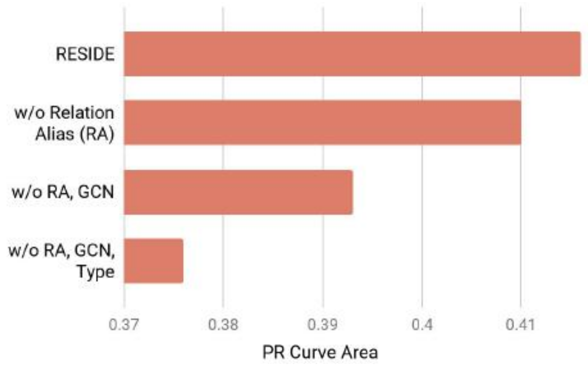
4.4.2.2 Ablation Results

In this section, we analyze the effect of various components of RESIDE on its performance. For this, we evaluate various versions of our model with cumulatively removed components. The experimental results are presented in Figure 4.4a. We observe that on removing different components from RESIDE, the performance of the model degrades drastically. The results validate that GCNs are effective at encoding syntactic information. Further, the improvement from side information shows that it is complementary to the features extracted from text, thus validating the central thesis of this chapter, that inducing side information leads to improved relation extraction.

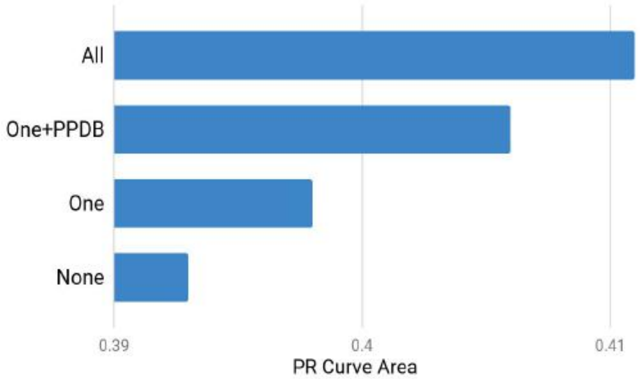
4.4.2.3 Effect of Relation Alias Side Information

In this section, we test the performance of the model in setting where relation alias information is not readily available. For this, we evaluate the performance of the model on four different settings:

- **None:** Relation aliases are not available.
- **One:** The name of relation is used as its alias.
- **One+PPDB:** Relation name extended using Paraphrase Database (PPDB).



(a) Performance comparison of different ablated version of RESIDE on Riedel dataset. Overall, GCN and side information helps RESIDE improve performance. Refer Section 4.4.2.2.



(b) Performance on settings defined in Section 4.4.2.3 with respect to the presence of relation alias side information on Riedel dataset. RESIDE performs comparably in the absence of relations from KB.

- **All:** Relation aliases from Knowledge Base[1]

The overall results are summarized in Figure 4.4b. We find that the model performs best when aliases are provided by the KB itself. Overall, we find that RESIDE gives competitive performance even when very limited amount of relation alias information is available. We observe that performance improves further with the availability of more alias information.

4.5 Conclusion

In this chapter, we propose RESIDE, a novel neural network based model which makes principled use of relevant side information, such as entity type and relation alias, from Knowledge Base, for improving distant supervised relation extraction. RESIDE employs Graph Convolution Networks for encoding syntactic information of sentences and is robust to limited side information. Through extensive experiments on benchmark datasets, we demonstrate RESIDE's effectiveness over state-of-the-art baselines. We have made RESIDE's source code publicly available to promote reproducible research.

[1]Each relation in Riedel dataset is manually mapped to corresponding Wikidata property for getting relation aliases. Few examples are presented in supplementary material.

Part II

Exploiting Graph Convolutional Networks in NLP

Chapter 5

Documents Timestamping using Graph Convolutional Networks

5.1 Introduction

In this chapter, we present our first application of Graph Convolutional Networks for solving document timestamping problem. Date of a document, also referred to as the Document Creation Time (DCT), is at the core of many important tasks, such as, information retrieval [145, 103, 42], temporal reasoning [118, 113], text summarization [200], event detection [1], and analysis of historical text [44], among others. In all such tasks, the document date is assumed to be available and also accurate – a strong assumption, especially for arbitrary documents from the Web. Thus, there is a need to automatically predict the date of a document based on its content. This problem is referred to as *Document Dating*.

Initial attempts on automatic document dating started with generative models by [44]. This model is later improved by [83] who incorporate additional features such as POS tags, collocations, etc. Chambers [32] shows significant improvement over these prior efforts through their discriminative models using handcrafted temporal features. Kotsakos et al. [91] propose a statistical approach for document dating exploiting term burstiness [95].

Document dating is a challenging problem which requires extensive reasoning over the temporal structure of the document. Let us motivate this through an example shown in Figure 5.1. In the document, *four years after* plays a crucial role in identifying the creation time of the document. The existing approaches give higher confidence for timestamp immediate to the year mention *1995*. NeuralDater exploits the syntactic and temporal structure of the document to predict the right timestamp (1999) for the document. With the exception of [32], all prior works on the document dating problem ignore such informative temporal structure within the

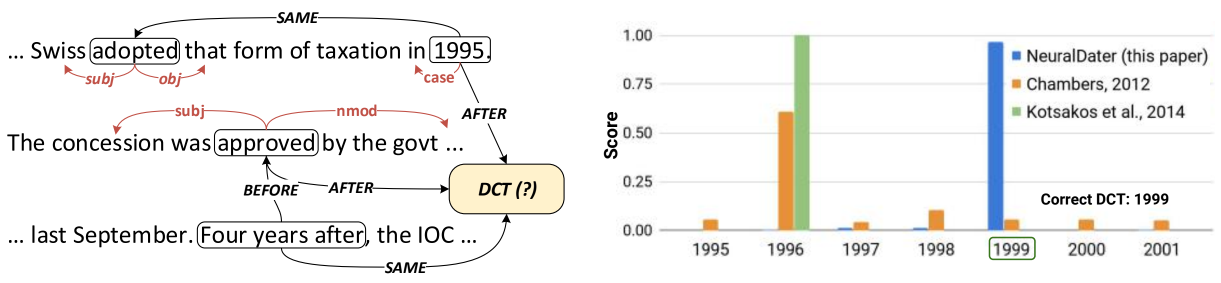


Figure 5.1: **Top:** An example document annotated with syntactic and temporal dependencies. In order to predict the right value of 1999 for the Document Creation Time (DCT), inference over these document structures is necessary. **Bottom:** Document date prediction by two state-of-the-art-baselines and NeuralDater, the method proposed in this chapter. While the two previous methods are getting misled by the temporal expression (*1995*) in the document, NeuralDater is able to use the syntactic and temporal structure of the document to predict the right value (*1999*).

document.

Research in document event extraction and ordering have made it possible to extract such temporal structures involving events, temporal expressions, and the (unknown) document date in a document [130, 35]. While methods to perform reasoning over such structures exist [196, 197, 189, 114, 157], none of them have exploited advances in deep learning [93, 73, 66]. In particular, recently proposed Graph Convolution Networks (GCN) [47, 88] have emerged as a way to learn graph representation while encoding structural information and constraints represented by the graph. We adapt GCNs for the document dating problem and make the following contributions:

- We propose NeuralDater, a Graph Convolution Network (GCN)-based approach for document dating. To the best of our knowledge, this is the first application of GCNs, and more broadly deep neural network-based methods, for the document dating problem.
- NeuralDater is the first document dating approach which exploits syntactic as well temporal structure of the document, all within a principled joint model.
- Through extensive experiments on multiple real-world datasets, we demonstrate NeuralDater's effectiveness over state-of-the-art baselines.

NeuralDater's source code and datasets used in the chapter are available at `http://github.com/malllabiisc/NeuralDater`.

5.2 Related Work

Automatic Document Dating: de Jong et al. [44] propose the first approach for automating document dating through a statistical language model. Kanhabua and Nørvåg [83] further extend this work by incorporating semantic-based preprocessing and temporal entropy [83] based term-weighting. Chambers [32] proposes a MaxEnt based discriminative model trained on hand-crafted temporal features. He also proposes a model to learn probabilistic constraints between year mentions and the actual creation time of the document. We draw inspiration from his work for exploiting temporal reasoning for document dating. Kotsakos et al. [91] propose a purely statistical method which considers lexical similarity alongside burstiness [95] of terms for dating documents. To the best of our knowledge, NeuralDater, our proposed method, is the first method to utilize deep learning techniques for the document dating problem.

Event Ordering Systems: Temporal ordering of events is a vast research topic in NLP. The problem is posed as a temporal relation classification between two given temporal entities. Machine Learned classifiers and well crafted linguistic features for this task are used in [34, 129]. D'Souza and Ng [53] use a hybrid approach by adding 437 hand-crafted rules. Chambers and Jurafsky [33], Yoshikawa et al. [217] try to classify with many more temporal constraints, while utilizing integer linear programming and Markov logic.

CAEVO, a CAscading EVent Ordering architecture [35] use sieve-based architecture [97] for temporal event ordering for the first time. They mix multiple learners according to their precision based ranks and use transitive closure for maintaining consistency of temporal graph. Mirza and Tonelli [130] recently propose CATENA (CAusal and TEmporal relation extraction from NAtural language texts), the first integrated system for the temporal and causal relations extraction between pre-annotated events and time expressions. They also incorporate sieve-based architecture which outperforms existing methods in temporal relation classification domain. We make use of CATENA for temporal graph construction in our work.

Graph Convolutional Networks (GCN): GCNs generalize Convolutional Neural Network (CNN) over graphs. GCN is introduced by [29], and later extended by [47] with efficient localized filter approximation in spectral domain. Kipf and Welling [88] propose a first-order approximation of localized filters through layer-wise propagation rule. GCNs over syntactic dependency trees have been recently exploited in the field of semantic-role labeling [121], neural machine translation [15], event detection [139]. In our work, we successfully use GCNs for document dating.

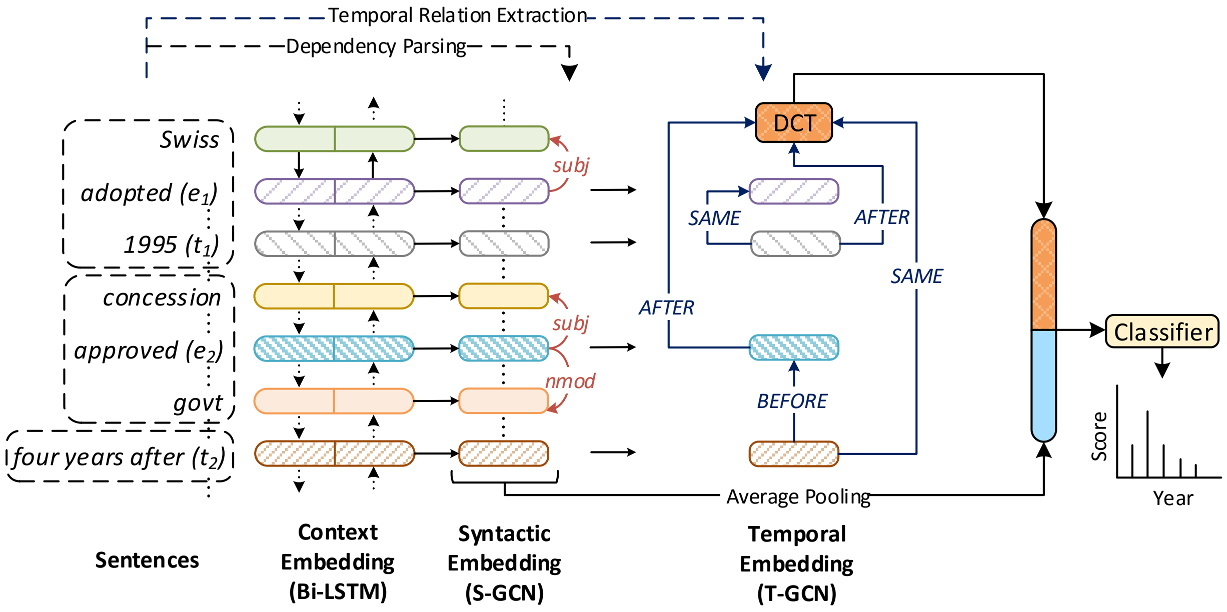


Figure 5.2: Overview of NeuralDater. NeuralDater exploits syntactic and temporal structure in a document to learn effective representation, which in turn are used to predict the document time. NeuralDater uses a Bi-directional LSTM (Bi-LSTM), two Graph Convolution Networks (GCN) – one over the dependency tree and the other over the document's temporal graph – along with a softmax classifier, all trained end-to-end jointly. Please see Section 5.3 for more details.

5.3 Proposed Approach: NeuralDater

5.3.1 Overview

The Documents Dating problem may be cast as a multi-class classification problem [91, 32]. In this section, we present an overview of NeuralDater, the document dating system proposed in this chapter. Architectural overview of NeuralDater is shown in Figure 5.2.

NeuralDater is a deep learning-based multi-class classification system. It takes in a document as input and returns its predicted date as output by exploiting the syntactic and temporal structure of document.

NeuralDater network consists of three layers which learn an embedding for the Document Creation Time (DCT) node corresponding to the document. This embedding is then fed to a softmax classifier which produces a distribution over timestamps. Following prior research [32, 91], we work with year granularity for the experiments in this chapter. We, however, note that NeuralDater can be trained for finer granularity with appropriate training data. The NeuralDater network is trained end-to-end using training data. We briefly present NeuralDater's various components below. Each component is described in greater detail in subsequent sections.

- **Context Embedding**: In this layer, NeuralDater uses a Bi-directional LSTM (Bi-LSTM) to learn embedding for each token in the document. Bi-LSTMs have been shown to be quite effective in capturing local context inside token embeddings [181].
- **Syntactic Embedding**: In this step, NeuralDater revises token embeddings from the previous step by running a GCN over the dependency parses of sentences in the document. We refer to this GCN as **Syntactic GCN** or **S-GCN**. While the Bi-LSTM captures immediate local context in token embeddings, S-GCN augments them by capturing syntactic context.
- **Temporal Embedding**: In this step, NeuralDater further refines embeddings learned by S-GCN to incorporate cues from temporal structure of event and times in the document. NeuralDater uses state-of-the-art causal and temporal relation extraction algorithm [130] for extracting temporal graph for each document. A GCN is then run over this temporal graph to refine the embeddings from the previous layer. We refer to this GCN as **Temporal GCN** or **T-GCN**. In this step, a special DCT node is introduced whose embedding is also learned by the T-GCN.
- **Classifier**: Embedding of the DCT node along with average pooled embeddings learned by S-GCN are fed to a fully connected softmax classifier which makes the final prediction about the date of the document.

Even though the previous discussion is presented in a sequential manner, the whole network is trained in a joint end-to-end manner using backpropagation. Below, we present detailed description of various components of NeuralDater.

5.3.2 Context Embedding (Bi-LSTM)

Let us consider a document D with n tokens $w_1, w_2, ..., w_n$. We first represent each token by a k-dimensional word embedding. For the experiments in this chapter, we use GloVe [152] embeddings. These token embeddings are stacked together to get the document representation $\mathcal{X} \in \mathbb{R}^{n \times k}$. We then employ a Bi-directional LSTM (Bi-LSTM) [74] on the input matrix $\mathcal{X}$ to obtain contextual embedding for each token. After stacking contextual embedding of all these tokens, we get the new document representation matrix $\mathcal{H}^{cntx} \in \mathbb{R}^{n \times r_{cntx}}$. In this new representation, each token is represented in a r_{cntx}-dimensional space. Our choice of LSTMs for learning contextual embeddings for tokens is motivated by the previous success of LSTMs in this task [181].

5.3.3 Syntactic Embedding (S-GCN)

While the Bi-LSTM is effective at capturing immediate local context of a token, it may not be as effective in capturing longer range dependencies among words in a sentence. For example, in Figure 5.1, we would like the embedding of token *approved* to be directly affected by *govt*, even though they are not immediate neighbors. A dependency parse may be used to capture such longer-range connections. In fact, similar features were exploited by [32] for the document dating problem. NeuralDater captures such longer-range information by using another GCN run over the syntactic structure of the document. We describe this in detail below.

The context embedding, $\mathcal{H}^{cntx} \in \mathbb{R}^{n \times r_{cntx}}$ learned in the previous step is used as input to this layer. For a given document, we first extract its syntactic dependency structure by applying the Stanford CoreNLP's dependency parser [120] on each sentence in the document individually. We now employ the Graph Convolution Network (GCN) over this dependency graph using the GCN formulation presented in Section 2.5. We call this GCN the Syntactic GCN or S-GCN, as mentioned in Section 5.3.

Since S-GCN operates over the dependency graph and uses Equation 2.9 for updating embeddings, the number of parameters in S-GCN is directly proportional to the number of dependency edge types. Stanford CoreNLP's dependency parser returns 55 different dependency edge types. This large number of edge types is going to significantly over-parameterize S-GCN, thereby increasing the possibility of overfitting. In order to address this, we use only three edge types in S-GCN. For each edge connecting nodes w_i and w_j in $\mathcal{E}'$ (see Equation 2.5), we determine its new type $L(w_i, w_j)$ as follows:

- $L(w_i, w_j) = \rightarrow$ if $(w_i, w_j, l(w_i, w_j)) \in \mathcal{E}'$, i.e., if edge is an original dependency parse edge
- $L(w_i, w_j) = \leftarrow$ if $(w_i, w_j, l(w_i, w_j)^{-1}) \in \mathcal{E}'$, i.e., if the edges is an inverse edge
- $L(w_i, w_j) = \top$ if $(w_i, w_j, \top) \in \mathcal{E}'$, i.e., if the edge is a self-loop with $w_i = w_j$

S-GCN now estimates embedding $h^{syn}_{w_i} \in \mathbb{R}^{r_{syn}}$ for each token w_i in the document using the formulation shown below.

$$h^{syn}_{w_i} = f\left(\sum_{w_j \in \mathcal{N}(w_i)} \left(W_{L(w_i,w_j)} h^{cntx}_{w_j} + b_{L(w_i,w_j)} \right) \right)$$

Please note S-GCN's use of the new edge types $L(w_i, w_j)$ above, instead of the $l(w_i, w_j)$ types used in Equation 2.9. By stacking embeddings for all the tokens together, we get the new embedding matrix $\mathcal{H}^{syn} \in \mathbb{R}^{n \times r_{syn}}$ representing the document.

AveragePooling: We obtain an embedding h_D^{avg} for the whole document by average pooling of every token representation.

$$h_D^{avg} = \frac{1}{n} \sum_{i=1}^{n} h_{w_i}^{syn}. \tag{5.1}$$

5.3.4 Temporal Embedding (T-GCN)

In this layer, NeuralDater exploits temporal structure of the document to learn an embedding for the Document Creation Time (DCT) node of the document. First, we describe the construction of temporal graph, followed by GCN-based embedding learning over this graph.

Temporal Graph Construction: NeuralDater uses Stanford's SUTime tagger [36] for date normalization and the event extraction classifier of [35] for event detection. The annotated document is then passed to CATENA [130], current state-of-the-art temporal and causal relation extraction algorithm, to obtain a temporal graph for each document. Since our task is to predict the creation time of a given document, we supply DCT as unknown to CATENA. We hypothesize that the temporal relations extracted in absence of DCT are helpful for document dating and we indeed find this to be true, as shown in Section 5.4.2. Temporal graph is a directed graph, where nodes correspond to events, time mentions, and the Document Creation Time (DCT). Edges in this graph represent causal and temporal relationships between them. Each edge is attributed with a label representing the type of the temporal relation. CATENA outputs 9 different types of temporal relations, out of which we selected five types, viz., *AFTER*, *BEFORE*, *SAME*, *INCLUDES*, and *IS_INCLUDED*. The remaining four types were ignored as they were substantially infrequent.

Please note that the temporal graph may involve only a small number of tokens in the document. For example, in the temporal graph in Figure 5.2, there are a total of 5 nodes: two temporal expression nodes (*1995* and *four years after*), two event nodes (*adopted* and *approved*), and a special DCT node. This graph also consists of temporal relation edges such as (*four years after*, *approved*, *BEFORE*).

Temporal Graph Convolution: NeuralDater employs a GCN over the temporal graph constructed above. We refer to this GCN as the Temporal GCN or T-GCN, as mentioned in Section 5.3. T-GCN is based on the GCN formulation presented in Section 2.5. Unlike S-GCN, here we consider label and direction specific parameters as the temporal graph consists of only five types of edges.

Datasets	# Docs	Start Year	End Year
APW	675k	1995	2010
NYT	647k	1987	1996

Table 5.1: Details of datasets used. Please see Section 5.4.1 for details.

Let n_T be the number of nodes in the temporal graph. Starting with $\mathcal{H}^{syn}$ (Section 5.3.3), T-GCN learns a r_{temp}-dimensional embedding for each node in the temporal graph. Stacking all these embeddings together, we get the embedding matrix $\mathcal{H}^{temp} \in \mathbb{R}^{n_T \times r_{temp}}$. T-GCN embeds the temporal constraints induced by the temporal graph in $h^{temp}_{DCT} \in \mathbb{R}^{r_{temp}}$, embedding of the DCT node of the document.

5.3.5 Classifier

Finally, the DCT embedding h^{temp}_{DCT} and average-pooled syntactic representation h^{avg}_D (see Equation 5.1) of document D are concatenated and fed to a fully connected feed forward network followed by a softmax. This allows the NeuralDater to exploit context, syntactic, and temporal structure of the document to predict the final document date y.

$$\begin{aligned} h^{avg+temp}_D &= [h^{temp}_{DCT} \ ; \ h^{avg}_D] \\ p(y|D) &= \text{Softmax}(W \cdot h^{avg+temp}_D + b). \end{aligned}$$

5.4 Experiments

5.4.1 Experimental Setup

Datasets: We experiment on Associated Press Worldstream (APW) and New York Times (NYT) sections of Gigaword corpus [147]. The original dataset contains around 3 million documents of APW and 2 million documents of NYT from span of multiple years. From both sections, we randomly sample around 650k documents while maintaining balance among years. Documents belonging to years with substantially fewer documents are omitted. Details of the dataset can be found in Table 5.1. For train, test and validation splits, the dataset was randomly divided in 80:10:10 ratio.

Evaluation Criteria: Given a document, the model needs to predict the year in which the document was published. We measure performance in terms of overall accuracy of the model.

Baselines: For evaluating NeuralDater, we compared against the following methods:

Method	APW	NYT
BurstySimDater	45.9	38.5
MaxEnt-Time+NER	52.5	42.3
MaxEnt-Joint	52.5	42.5
MaxEnt-Uni-Time	57.5	50.5
CNN	56.3	50.4
NeuralDater	**64.1**	**58.9**

Table 5.2: Accuracies of different methods on APW and NYT datasets for the document dating problem (higher is better). NeuralDater significantly outperforms all other competitive baselines. This is our main result. Please see Section 5.4.2.1 for more details.

- **BurstySimDater** Kotsakos et al. [91]: This is a purely statistical method which uses lexical similarity and term burstiness [95] for dating documents in arbitrary length time frame. For our experiments, we took the time frame length as 1 year. Please refer to [91] for more details.

- **MaxEnt-Time-NER**: Maximum Entropy (MaxEnt) based classifier trained on hand-crafted temporal and Named Entity Recognizer (NER) based features. More details in [32].

- **MaxEnt-Joint**: Refers to MaxEnt-Time-NER combined with year mention classifier as described in [32].

- **MaxEnt-Uni-Time:** MaxEnt based discriminative model which takes bag-of-words representation of input document with normalized time expression as its features.

- **CNN:** A Convolution Neural Network (CNN) [96] based text classification model proposed by [85], which attained state-of-the-art results in several domains.

- **NeuralDater**: Our proposed method, refer Section 5.3.

Hyperparameters: By default, edge gating (Section 2.5) is used in all GCNs. The parameter K represents the number of layers in T-GCN (Section 5.3.4). We use 300-dimensional GloVe embeddings and 128-dimensional hidden state for both GCNs and BiLSTM with 0.8 dropout. We used Adam [86] with 0.001 learning rate for training.

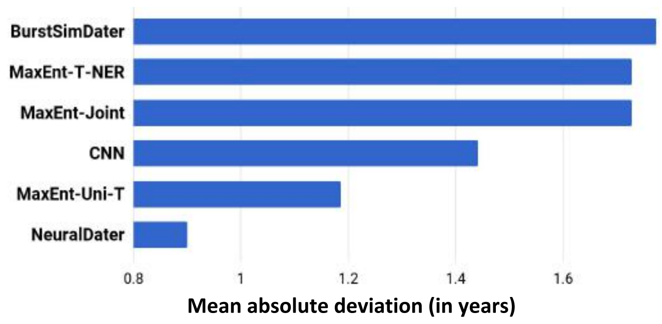
Method	Accuracy
T-GCN	57.3
S-GCN + T-GCN ($K = 1$)	57.8
S-GCN + T-GCN ($K = 2$)	58.8
S-GCN + T-GCN ($K = 3$)	**59.1**
Bi-LSTM	58.6
Bi-LSTM + CNN	59.0
Bi-LSTM + T-GCN	60.5
Bi-LSTM + S-GCN + T-GCN (no gate)	62.7
Bi-LSTM + S-GCN + T-GCN ($K = 1$)	**64.1**
Bi-LSTM + S-GCN + T-GCN ($K = 2$)	63.8
Bi-LSTM + S-GCN + T-GCN ($K = 3$)	63.3

Table 5.3: Accuracies of different ablated methods on the APW dataset. Overall, we observe that incorporation of context (Bi-LSTM), syntactic structure (S-GCN) and temporal structure (T-GCN) in NeuralDater achieves the best performance. Please see Section 5.4.2.1 for details.

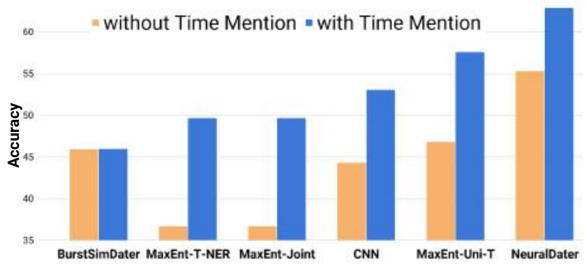
5.4.2 Results

5.4.2.1 Performance Comparison

In order to evaluate the effectiveness of NeuralDater, our proposed method, we compare it against existing document dating systems and text classification models. The final results are summarized in Table 5.2. Overall, we find that NeuralDater outperforms all other methods with a significant margin on both datasets. Compared to the previous state-of-the-art in document dating, BurstySimDater [91], we get 19% average absolute improvement in accuracy across both datasets. We observe only a slight gain in the performance of MaxEnt-based model (MaxEnt-Time+NER) of [32] on combining with temporal constraint reasoner (MaxEnt-Joint). This may be attributed to the fact that the model utilizes only year mentions in the document, thus ignoring other relevant signals which might be relevant to the task. BurstySimDater performs considerably better in terms of precision compared to the other baselines, although it significantly underperforms in accuracy. We note that NeuralDater outperforms all these prior models both in terms of precision and accuracy. We find that even generic deep-learning based text classification models, such as CNN [85], are quite effective for the problem. However, since such a model doesn't give specific attention to temporal features in the document, its performance remains limited. From Figure 5.3a, we observe that NeuralDater's top prediction achieves on average the lowest deviation from the true year.



(a) Mean absolute deviation (in years; lower is better) between a model's top prediction and the true year in the APW dataset. We find that NeuralDater, the proposed method, achieves the least deviation. Please see Section 5.4.2.1 for details



(b) Evaluating performance of different methods on dating documents with and without time mentions. Please see Section 5.4.2.3 for details.

5.4.2.2 Ablation Comparisons

For demonstrating the efficacy of GCNs and BiLSTM for the problem, we evaluate different ablated variants of NeuralDater on the APW dataset. Specifically, we validate the importance of using syntactic and temporal GCNs and the effect of eliminating BiLSTM from the model. Overall results are summarized in Table 5.3. The first block of rows in the table corresponds to the case when BiLSTM layer is excluded from NeuralDater, while the second block denotes the case when BiLSTM is included. We also experiment with multiple stacked layers of T-GCN (denoted by K) to observe its effect on the performance of the model.

We observe that embeddings from Syntactic GCN (S-GCN) are much better than plain GloVe embeddings for T-GCN as S-GCN encodes the syntactic neighborhood information in event and time embeddings which makes them more relevant for document dating task.

Overall, we observe that including BiLSTM in the model improves performance significantly. Single BiLSTM model outperforms all the models listed in the first block of Table 5.3. Also, some gain in performance is observed on increasing the number of T-GCN layers (K) in absence of BiLSTM, although the same does not follow when BiLSTM is included in the model. This observation is consistent with [121], as multiple GCN layers become redundant in the presence of BiLSTM. We also find that eliminating edge gating from our best model deteriorates its overall performance.

In summary, these results validate our thesis that joint incorporation of syntactic and temporal structure of a document in NeuralDater results in improved performance.

5.4.2.3 Discussion and Error Analysis

In this section, we list some of our observations while trying to identify pros and cons of NeuralDater, our proposed method. We divided the development split of the APW dataset into two sets – those with and without any mention of time expressions (year). We apply NeuralDater and other methods to these two sets of documents and report accuracies in Figure 5.3b. We find that overall, NeuralDater performs better in comparison to the existing baselines in both scenarios. Even though the performance of NeuralDater degrades in the absence of time mentions, its performance is still the best relatively. Based on other analysis, we find that NeuralDater fails to identify timestamp of documents reporting local infrequent incidents without explicit time mention. NeuralDater becomes confused in the presence of multiple misleading time mentions; it also loses out on documents discussing events which are outside the time range of the text on which the model was trained. In future, we plan to eliminate these pitfalls by incorporating additional signals from Knowledge Graphs about entities mentioned in the document. We also plan to utilize free text temporal expression [94] in documents for improving performance on this problem.

5.5 Conclusion

We propose NeuralDater, a Graph Convolutional Network (GCN) based method for document dating which exploits syntactic and temporal structures in the document in a principled way. To the best of our knowledge, this is the first application of deep learning techniques for the problem of document dating. Through extensive experiments on real-world datasets, we demonstrate the effectiveness of NeuralDater over existing state-of-the-art approaches. We are hopeful that the representation learning techniques explored in this chapter will inspire further development and adoption of such techniques in the temporal information processing research community.

Chapter 6

Incorporating Syntactic and Semantic Information in Word Embeddings using Graph Convolutional Networks

6.1 Introduction

As we saw in the last chapter, Graph Convolutional Networks prove to be very effective for exploiting different graph structure in NLP. Here, we utilize them for learning word representation. Representing words as real-valued vectors is an effective and widely adopted technique in NLP. Such representations capture properties of words based on their usage and allow them to generalize across tasks. Meaningful word embeddings have been shown to improve performance on several relevant tasks, such as named entity recognition (NER) [18], parsing [171], and part-of-speech (POS) tagging [116]. Using word embeddings for initializing Deep Neural Networks has also been found to be quite useful [41, 81, 176].

Most popular methods for learning word embeddings are based on the distributional hypothesis, which utilizes the co-occurrence statistics from *sequential* context of words for learning word representations [125, 152]. More recently, this approach has been extended to include syntactic contexts [100] derived from dependency parse of text. Higher order dependencies have also been exploited by Komninos and Manandhar [90], Li et al. [101]. Syntax-based embeddings encode functional similarity (in-place substitutable words) rather than topical similarity (topically related words) which provides an advantage on specific tasks like question classification [90]. However, current approaches incorporate syntactic context by concatenating words with their dependency relations. For instance, in Figure 6.1 *scientists_subj*, *water_obj*, and *mars_nmod* needs to be included as a part of vocabulary for utilizing the dependency context of *discover*. This severely expands the vocabulary, thus limiting the scalability of models on large corpora. For instance, in Levy and Goldberg [100] and Komninos and Manandhar [90], the context vocabulary explodes to around 1.3 million for learning embeddings of 220k words.

Incorporating relevant signals from semantic knowledge sources such as WordNet [127], FrameNet [8], and Paraphrase Database (PPDB) [151] has been shown to improve the quality of word embeddings. Recent works utilize these by incorporating them in a neural language modeling objective function [218, 3], or as a post-processing step [55, 135]. Although existing approaches improve the quality of word embeddings, they require explicit modification for handling different types of semantic information.

Recently proposed Graph Convolutional Networks (GCN) [46, 87] have been found to be useful for encoding structural information in graphs. Even though GCNs have been successfully employed for several NLP tasks such as machine translation [15], semantic role labeling [121], document dating [191] and text classification [213], they have so far not been used for learning word embeddings, especially leveraging cues such as syntactic and semantic information. GCNs provide flexibility to represent diverse syntactic and semantic relationships between words all within one framework, without requiring relation-specific special handling as in previous methods. Recognizing these benefits, we make the following contributions in this chapter.

1. We propose SynGCN, a Graph Convolution based method for learning word embeddings. Unlike previous methods, SynGCN utilizes syntactic context for learning word representations without increasing vocabulary size.
2. We also present SemGCN, a framework for incorporating diverse semantic knowledge (e.g., synonymy, antonymy, hyponymy, etc.) in learned word embeddings, without requiring relation-specific special handling as in previous methods.
3. Through experiments on multiple intrinsic and extrinsic tasks, we demonstrate that our proposed methods obtain substantial improvement over state-of-the-art approaches, and also yield an advantage when used in conjunction with methods such as ELMo [154].

6.2 Related Work

Word Embeddings: Recently, there has been much interest in learning meaningful word representations such as neural language modeling [19] based continuous-bag-of-words (CBOW) and skip-gram (SG) models [125]. This is further extended by Pennington et al. [152] which learns embeddings by factorizing word co-occurrence matrix to leverage global statistical information. Other formulations for learning word embeddings include multi-task learning [41] and ranking frameworks [79].

Syntax-based Embeddings: Dependency parse context based word embeddings is first introduced by Levy and Goldberg [100]. They allow encoding syntactic relationships between words and show improvements on tasks where functional similarity is more relevant than topical similarity. The inclusion of syntactic context is further enhanced through second-order [90] and multi-order [101] dependencies. However, in all these existing approaches, the word vocabulary is severely expanded for incorporating syntactic relationships.

Incorporating Semantic Knowledge Sources: Semantic relationships such as *synonymy*, *antonymy*, *hypernymy*, etc. from several semantic sources have been utilized for improving the quality of word representations. Existing methods either exploit them jointly [205, 84, 3] or as a post-processing step [55, 135]. SynGCN falls under the latter category and is more effective at incorporating semantic constraints (Section 6.4.2.2 and 6.4.2.3).

Graph Convolutional Networks: In this chapter, we use the first-order formulation of GCNs via a layer-wise propagation rule as proposed by [87]. Recently, some variants of GCNs have also been proposed [208, 194]. A detailed description of GCNs and their applications can be found in Bronstein et al. [27]. In NLP, GCNs have been utilized for semantic role labeling [121], machine translation [15], and relation extraction [193]. Recently, Yao et al. [213] use GCNs for text classification by jointly embedding words and documents. However, their learned embeddings are task specific whereas in our work we aim to learn task agnostic word representations.

6.3 Proposed Methods: SynGCN and SemGCN

6.3.1 Overview

The task of learning word representations in an unsupervised setting can be formulated as follows: Given a text corpus, the aim is to learn a d-dimensional embedding for each word in the vocabulary. Most of the distributional hypothesis based approaches only utilize sequential context for each word in the corpus. However, this becomes suboptimal when the relevant context words lie beyond the window size. For instance in Figure 6.1, a relevant context word *discover* for *Mars* is missed if the chosen window size is less than 3. On the contrary, a large window size might allow irrelevant words to influence word embeddings negatively.

Using dependency based context helps to alleviate this problem. However, all existing syntactic context based methods [100, 90, 101] severely expand vocabulary size (as discussed in Section 6.1) which limits their scalability to a large corpus. To eliminate this drawback, we propose SynGCN which employs Graph Convolution Networks to better encode syntactic information in embeddings. We prefer GCNs over other graph encoding architectures such as

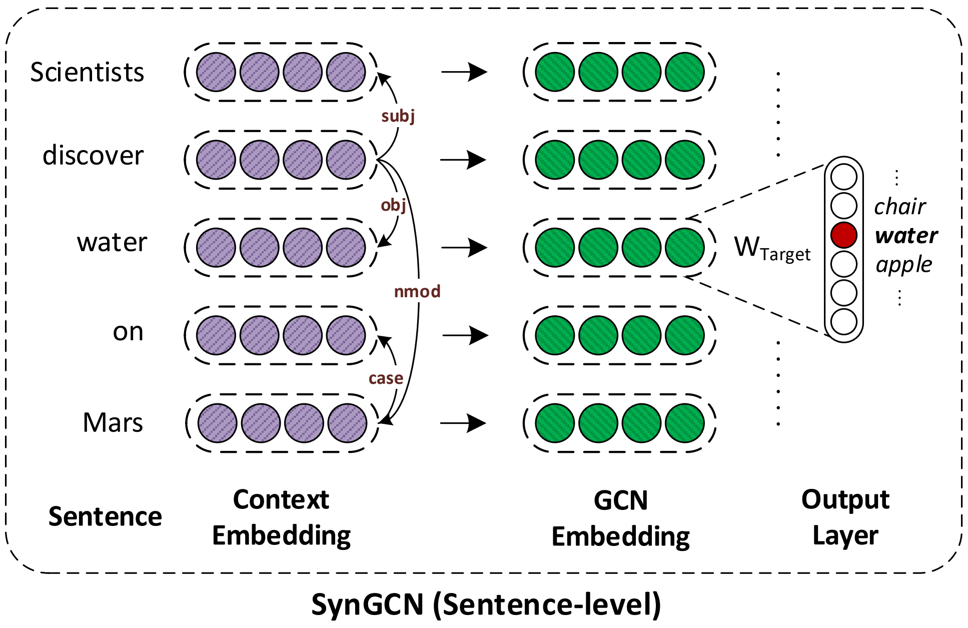


Figure 6.1: Overview of SynGCN: SynGCN employs Graph Convolution Network for utilizing dependency context for learning word embeddings. For each word in vocabulary, the model learns its representation by aiming to predict each word based on its dependency context encoded using GCNs. Please refer Section 6.3.2 for more details.

Tree LSTM [182] as GCNs do not restrict graphs to be trees and have been found to be more effective at capturing global information [223]. Moreover, they give substantial speedup as they do not involve recursive operations which are difficult to parallelize. The overall architecture is shown in Figure 6.1, for more details refer to Section 6.3.2.

Enriching word embeddings with semantic knowledge helps to improve their quality for several NLP tasks. Existing approaches are either incapable of utilizing these diverse relations or need to be explicitly modeled for exploiting them. In this chapter, we propose SemGCN which automatically learns to utilize multiple semantic constraints by modeling them as different edge types. It can be used as a post-processing method similar to Faruqui et al. [55], Mrkšić et al. [135]. We describe it in more detail in Section 6.3.3.

6.3.2 SynGCN

In this section, we provide a detailed description of our proposed method, SynGCN. Following Mikolov et al. [126], Levy and Goldberg [100], Komninos and Manandhar [90], we separately define target and context embeddings for each word in the vocabulary as parameters in the model. For a given sentence $s = (w_1, w_2, \ldots, w_n)$, we first extract its dependency parse graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$ using Stanford CoreNLP parser [120]. Here, $\mathcal{V}_s = \{w_1, w_2, \ldots, w_n\}$ and $\mathcal{E}_s$ denotes the labeled directed dependency edges of the form (w_i, w_j, l_{ij}), where l_{ij} is the dependency relation of w_i to w_j.

Similar to Mikolov et al. [126]'s continuous-bag-of-words (CBOW) model, which defines the context of a word w_i as $\mathcal{C}_{w_i} = \{w_{i+j} : -c \leq j \leq c, j \neq 0\}$ for a window of size c, we define the context as its neighbors in $\mathcal{G}_s$, i.e., $\mathcal{C}_{w_i} = \mathcal{N}(w_i)$. Now, unlike CBOW which takes the sum of the context embedding of words in $\mathcal{C}_{w_i}$ to predict w_i, we apply directed Graph Convolution Network (as defined in Section 2.5) on $\mathcal{G}_s$ with context embeddings of words in s as input features. Thus, for each word w_i in s, we obtain a representation h_i^{k+1} after k-layers of GCN using Equation 2.9 which we reproduce below for ease of readability (with one exception as described below).

$$h_i^{k+1} = f\left(\sum_{j \in \mathcal{N}(i)} g_{l_{ij}}^k \times \left(W_{l_{ij}}^k h_j^k + b_{l_{ij}}^k\right)\right)$$

Please note that unlike in Equation 2.9, we use $\mathcal{N}(i)$ instead of $\mathcal{N}_+(i)$ in SynGCN, i.e., we do not include self-loops in $\mathcal{G}_s$. This helps to avoid overfitting to the initial embeddings, which is undesirable in the case of SynGCN as it uses random initialization. We note that similar strategy has been followed by Mikolov et al. [126]. Furthermore, to handle erroneous edges in automatically constructed dependency parse graph, we perform edge-wise gating (Section 2.5) to give importance to relevant edges and suppress the noisy ones. The embeddings obtained are then used to calculate the loss as described in Section 6.3.4.

SynGCN utilizes *syntactic context* to learn more meaningful word representations. We validate this in Section 6.4.2.1. Note that, the word vocabulary remains unchanged during the entire learning process, this makes SynGCN more scalable compared to the existing approaches.

Note that, SynGCN is a generalization of CBOW model, as shown below.

Theorem 6.1 *SynGCN is a generalization of Continuous-bag-of-words (CBOW) model.*

Proof: The reduction can be obtained as follows. For a given sentence s, take the neighborhood of each word w_i in $\mathcal{G}_s$ as it sequential context, i.e., $\mathcal{N}(w_i) = \{w_{i+j} : -c \leq j \leq c, j \neq 0\}\ \forall w_i \in s$. Now, if the number of GCN layers are restricted to 1 and the activation function is taken as identity ($f(x) = x$), then Equation 2.9 reduces to

$$h_i = \sum_{-c \leq j \leq c, j \neq 0} \left(g_{l_{ij}} \times \left(W_{l_{ij}} h_j + b_{l_{ij}}^k\right)\right).$$

Finally, $W_{l_{ij}}^k$ and $b_{l_{ij}}^k$ can be fixed to an identity matrix ($\mathbf{I}$) and a zero vector ($\mathbf{0}$), respectively,

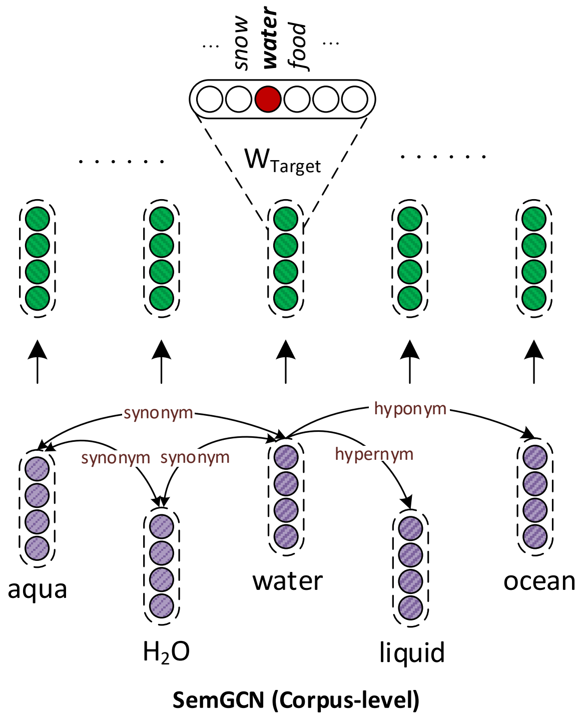


Figure 6.2: Overview of SemGCN, our proposed Graph Convolution based framework for incorporating diverse semantic information in learned embeddings. Double-headed edges denote two edges in both directions. Please refer to Section 6.3.3 for more details.

and edge-wise gating ($g_{l_{ij}}$) can be set to 1. This gives

$$h_i = \sum_{-c \le j \le c, j \ne 0} (\mathbf{I} \cdot h_j + \mathbf{0}) = \sum_{-c \le j \le c, j \ne 0} h_j,$$

which is the hidden layer equation of CBOW model. □

6.3.3 SemGCN

In this section, we propose another Graph Convolution based framework, SemGCN, for incorporating semantic knowledge in pre-trained word embeddings. Most of the existing approaches like Faruqui et al. [55], Mrkšić et al. [135] are restricted to handling symmetric relations like *synonymy* and *antonymy*. On the other hand, although recently proposed [3] is capable of handling asymmetric information, it still requires manually defined relation strength function which can be labor intensive and suboptimal.

SemGCN is capable of incorporating both symmetric as well as asymmetric information jointly. Unlike SynGCN, SemGCN operates on a corpus-level directed labeled graph with words as nodes and edges representing semantic relationship among them from different sources.

For instance, in Figure 6.2, semantic relations such as *hyponymy*, *hypernymy* and *synonymy* are represented together in a single graph. Symmetric information is handled by including a directed edge in both directions. Given the corpus level graph $\mathcal{G}$, the training procedure is similar to that of SynGCN, i.e., predict the word w based on its neighbors in $\mathcal{G}$. Inspired by Faruqui et al. [55], we preserve the semantics encoded in pre-trained embeddings by initializing both target and context embeddings with given word representations and keeping target embeddings fixed during training. SemGCN uses Equation 2.9 to update node embeddings. Please note that in this case $\mathcal{N}_{+}(v)$ is used as the neighborhood definition to preserve the initial learned representation of the words.

6.3.4 Training Details

Given the GCN representation (h_t) of a word (w_t), the training objective of SynGCN and SemGCN is to predict the target word given its neighbors in the graph. Formally, for each method we maximize the following objective[1].

$$E = \sum_{t=1}^{|V|} \log P(w_t | w_1^t, w_2^t \dots w_{N_t}^t)$$

where, w_t is the target word and $w_1^t, w_2^t \dots w_{N_t}^t$ are its neighbors in the graph. The probability $P(w_t | w_1^t, w_2^t \dots w_{N_t}^t)$ is calculated using the softmax function, defined as

$$P(w_t | w_1^t, w_2^t \dots w_{N_t}^t) = \frac{\exp(v_{w_t}^T h_t)}{\sum_{i=1}^{|V|} \exp(v_{w_i}^T h_t)}.$$

Hence, E reduces to

$$E = \sum_{t=1}^{|V|} \left(v_{w_t}^T h_t - \log \sum_{i=1}^{|V|} \exp(v_{w_i}^T h_t) \right), \tag{6.1}$$

where, h_t is the GCN representation of the target word w_t and v_{w_t} is its target embedding.

The second term in Equation 6.1 is computationally expensive as the summation needs to be taken over the entire vocabulary. This can be overcome using several approximations like noise-contrastive estimation [69] and hierarchical softmax [134]. In our methods, we use negative sampling as used by Mikolov et al. [126].

[1]We also experimented with joint SynGCN and SemGCN model but our preliminary experiments gave suboptimal performance as compared to the sequential model. This can be attributed to the fact that syntactic information is orders of magnitude greater than the semantic information available. Hence, the semantic constraints are not effectively utilized. We leave the analysis of the joint model as a future work.

6.4 Experiments

6.4.1 Experimental Setup

6.4.1.1 Dataset and Training

In our experiments, we use Wikipedia[1] corpus for training the models. After discarding too long and too short sentences, we get an average sentence length of nearly 20 words. The corpus consists of 57 million sentences with 1.1 billion tokens and 1 billion syntactic dependencies.

6.4.1.2 Baselines

For evaluating SynGCN (Section 6.3.2), we compare against the following baselines:

- **Word2vec** is continuous-bag-of-words model originally proposed by Mikolov et al. [126].
- **GloVe** [152], a log-bilinear regression model which leverages global co-occurrence statistics of corpus.
- **Deps** [100] is a modification of skip-gram model which uses dependency context in place of sequential context.
- **EXT** [90] is an extension of Deps which utilizes second-order dependency context features.

SemGCN (Section 6.3.3) model is evaluated against the following methods:

- **Retro-fit** [55] is a post-processing procedure which uses similarity constraints from semantic knowledge sources.
- **Counter-fit** [135], a method for injecting both antonym and synonym constraints into word embeddings.
- **JointReps** [3], a joint word representation learning method which simultaneously utilizes the corpus and KB.

6.4.1.3 Evaluation method

To evaluate the effectiveness of our proposed methods, we compare them against the baselines on the following intrinsic and extrinsic tasks[2]:

- **Intrinsic Tasks:**

  Word Similarity is the task of evaluating closeness between semantically similar words. Following Komninos and Manandhar [90], Pennington et al. [152], we evaluate on Simlex-999 [72], WS353 [58], and RW [115] datasets.

  Concept Categorization involves grouping nominal concepts into natural categories.

[1]https://dumps.wikimedia.org/enwiki/20180301/

[2]Details of hyperparameters are in supplementary.

	Word Similarity				Concept Categorization				Word Analogy	
Method	WS353S	WS353R	SimLex999	RW	AP	Battig	BLESS	ESSLI	SemEval2012	MSR
Word2vec	71.4	52.6	38.0	30.0	63.2	43.3	77.8	63.0	18.9	44.0
GloVe	69.2	**53.4**	36.7	29.6	58.0	41.3	80.0	59.3	18.7	45.8
Deps	65.7	36.2	39.6	33.0	61.8	41.7	65.9	55.6	22.9	40.3
EXT	69.6	44.9	43.2	18.6	52.6	35.0	65.2	66.7	21.8	18.8
SynGCN	**73.2**	45.7	**45.5**	**33.7**	**69.3**	**45.2**	**85.2**	**70.4**	**23.4**	**52.8**

Table 6.1: **SynGCN Intrinsic Evaluation:** Performance on word similarity (Spearman correlation), concept categorization (cluster purity), and word analogy (Spearman correlation). Overall, SynGCN outperforms other existing approaches in 9 out of 10 settings. Please refer to Section 6.4.2.1 for more details.

For instance, *tiger* and *elephant* should belong to *mammal* class. In our experiments, we evalute on AP [2], Battig [12], BLESS [13], ESSLI [14] datasets.

Word Analogy task is to predict word b_2, given three words a_1, a_2, and b_1, such that the relation $b_1 : b_2$ is same as the relation $a_1 : a_2$. We compare methods on MSR [126] and SemEval-2012 [82].

- **Extrinsic Tasks:**

  Named Entity Recognition (NER) is the task of locating and classifying entity mentions into categories like *person*, *organization* etc. We use Lee et al. [98]'s model on CoNLL-2003 dataset [185] for evaluation.

  Question Answering in Stanford Question Answering Dataset (**SQuAD**) [158] involves identifying answer to a question as a segment of text from a given passage. Following Peters et al. [154], we evaluate using Clark and Gardner [40]'s model.

  Part-of-speech (POS) tagging aims at associating with each word, a unique tag describing its syntactic role. For evaluating word embeddings, we use Lee et al. [98]'s model on Penn Treebank POS dataset [122].

  Co-reference Resolution (Coref) involves identifying all expressions that refer to the same entity in the text. To inspect the effect of embeddings, we use Lee et al. [98]'s model on CoNLL-2012 shared task dataset [155].

6.4.2 Results

In this section, we attempt to answer the following questions.

Q1. Does SynGCN learn better word embeddings than existing approaches? (Section 6.4.2.1)

Q2. Does SemGCN effectively handle diverse semantic information as compared to other methods? (Section 6.4.2.2)

Method	POS	SQuAD	NER	Coref
Word2vec	95.0±0.1	78.5±0.3	89.0±0.2	65.1±0.3
GloVe	94.6±0.1	78.2±0.2	89.1±0.1	64.9±0.2
Deps	95.0±0.1	77.8±0.3	88.6±0.3	64.8±0.1
EXT	94.9±0.2	**79.6±0.1**	88.0±0.1	64.8±0.1
SynGCN	**95.4±0.1**	**79.6±0.2**	**89.5±0.1**	**65.8±0.1**

Table 6.2: **SynGCN Extrinsic Evaluation:** Comparison on parts-of-speech tagging (POS), question answering (SQuAD), named entity recognition (NER), and co-reference resolution (Coref). SynGCN performs comparable or outperforms all existing approaches on all tasks. Refer Section 6.4.2.1 for details.

Q3. How does SemGCN perform compared to other methods when provided with the same semantic constraints? (Section 6.4.2.3)

Q4. Does dependency context based embedding encode complementary information compared to ELMo? (Section 6.4.2.4)

6.4.2.1 SynGCN Evaluation

The evaluation results on intrinsic tasks – word similarity, concept categorization, and analogy – are summarized in Table 6.1. We report Spearman correlation for word similarity and analogy tasks and cluster purity for concept categorization task. Overall, we find that SynGCN, our proposed method, outperforms all the existing word embedding approaches in 9 out of 10 settings. The inferior performance of SynGCN and other dependency context based methods on WS353R dataset compared to sequential context based methods is consistent with the observation reported in Levy and Goldberg [100], Komninos and Manandhar [90]. This is because the syntactic context based embeddings capture functional similarity rather than topical similarity (as discussed in Section 6.1). On average, we obtain around 1.5%, 5.7% and 7.5% absolute increase in performance on word similarity, concept categorization and analogy tasks compared to the best performing baseline. The results demonstrate that the learned embeddings from SynGCN more effectively capture semantic and syntactic properties of words.

We also evaluate the performance of different word embedding approaches on the downstream tasks as defined in Section 6.4.1.3. The experimental results are summarized in Table 6.2. Overall, we find that SynGCN either outperforms or performs comparably to other methods on all four tasks. Compared to the sequential context based methods, dependency based methods perform superior at question answering task as they effectively encode syntactic information. This is consistent with the observation of Peters et al. [154].

Init Embeddings (=X)	Word2vec			GloVe			Deps			EXT			SynGCN		
Datasets	WS353	AP	MSR	WS353	AP	MSR	WS353	AP	MSR	WS353	AP	MSR	WS353	AP	MSR
Performance of X	63.0	63.2	44.0	58.0	60.4	45.8	55.6	64.2	40.3	59.3	53.5	18.8	61.7	69.3	52.8
Retro-fit (X,1)	63.4	**67.8**	**46.7**	58.5	61.1	**47.2**	54.8	64.7	41.0	61.6	55.1	40.5	61.2	67.1	51.4
Counter-fit (X,2)	60.3	62.9	31.4	53.7	62.5	29.6	46.9	60.4	33.4	52.0	54.4	35.8	55.2	66.4	31.7
JointReps (X,4)	60.9	61.1	28.5	59.2	55.5	37.6	54.8	58.7	38.0	58.8	54.8	20.6	60.9	68.2	24.9
SemGCN (X,4)	**64.8**	**67.8**	36.8	**63.3**	**63.2**	44.1	**62.3**	$\boxed{\textbf{69.3}}$	**41.1**	**62.9**	**67.1**	**52.1**	$\boxed{\textbf{65.3}}$	$\boxed{\textbf{69.3}}$	$\boxed{\textbf{54.4}}$

Table 6.3: **SemGCN Intrinsic Evaluation:** Evaluation of different methods for incorporating diverse semantic constraints initialized using various pre-trained embeddings (X). M(X, R) denotes the fine-tuned embeddings using method M taking X as initialization embeddings. R denotes the type of semantic relations used as defined in Section 6.4.2.2. SemGCN outperforms other methods in 13 our of 15 settings. SemGCN with SynGCN gives the best performance across all tasks (highlighted using $\boxed{\cdot}$). Please refer Section 6.4.2.2 for details.

Method	POS	SQuAD	NER	Coref
X = SynGCN	95.4±0.1	79.6±0.2	**89.5±0.1**	65.8±0.1
Retro-fit (X,1)	94.8±0.1	79.6±0.1	88.8±0.1	66.0±0.2
Counter-fit (X,2)	94.7±0.1	79.8±0.1	88.3±0.3	65.7±0.3
JointReps (X,4)	95.4±0.1	79.4±0.3	89.1±0.3	65.6±0.1
SemGCN (X,4)	**95.5±0.1**	**80.4±0.1**	**89.5±0.1**	**66.1±0.1**

Table 6.4: **SemGCN Extrinsic Evaluation:** Comparison of different methods for incorporating diverse semantic constraints in SynGCN embeddings on all extrinsic tasks. Refer Section 6.4.2.2 for details.

6.4.2.2 Evaluation with Diverse Semantic Information

In this section, we compare SemGCN against the methods listed in Section 6.4.1.2 for incorporating diverse semantic information in pre-trained embeddings. We use *hypernym*, *hyponym*, and *antonym* relations from WordNet, and *synonym* relations from PPDB as semantic information. For each method, we provide the semantic information that it can utilize, e.g., Retro-fit can only make use of *synonym* relation[1]. In our results, **M(X, R)** denotes the fine-tuned embeddings obtained using method M while taking X as initialization embeddings. R denotes the types of semantic information used as defined below.

- **R=1:** Only synonym information.
- **R=2:** Synonym and antonym information.
- **R=4:** Synonym, antonym, hypernym and hyponym information.

For instance, Counter-fit (GloVe, 2) represents GloVe embeddings fine-tuned by Counter-fit using synonym and antonym information.

[1]Experimental results controlling for semantic information are provided in Section 6.4.2.3.

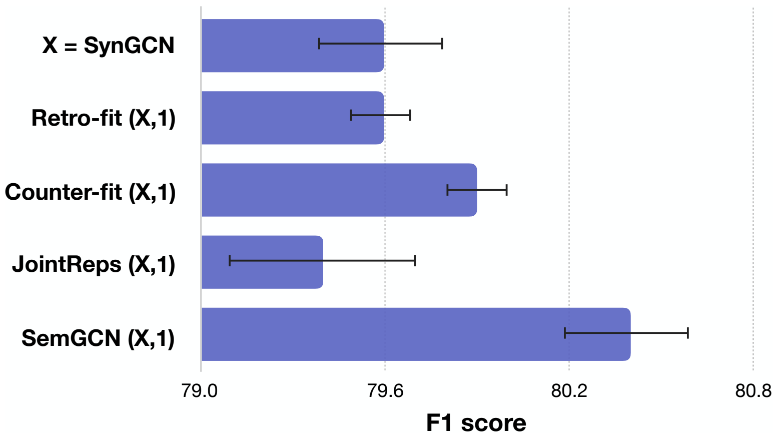


Figure 6.3: Comparison of different methods when provided with the same semantic information (synonym) for fine tuning SynGCN embeddings. Results denote the F1-score on SQuAD dataset. SemGCN gives considerable improvement in performance. Please refer Section 6.4.2.3 for details.

Similar to Section 6.4.2.1, the evaluation is performed on the three intrinsic tasks. Due to space limitations, we report results on one representative dataset per task. The results are summarized in Table 6.3. We find that in 13 out of 15 settings, SemGCN outperforms other methods. Overall, we observe that SemGCN, when initialized with SynGCN, gives the best performance on all the tasks (highlighted by $\boxed{\cdot}$ in Table 6.3).

For comparing performance on the extrinsic tasks, we first fine-tune SynGCN embeddings using different methods for incorporating semantic information. The embeddings obtained by this process are then evaluated on extrinsic tasks, as in Section 6.4.2.1. The results are shown in Table 6.4. We observe that while the other methods do not always consistently give improvement over the baseline SynGCN, SemGCN is able to improve upon SynGCN in all settings (better or comparable). Overall, we observe that SynGCN along with SemGCN is the most suitable method for incorporating both syntactic and semantic information.

6.4.2.3 Evaluation with Same Semantic Information

In this section, we compare SemGCN against other baselines when provided with the same semantic information: synonyms from PPDB. Similar to Section 6.4.2.2, we compare both on intrinsic and extrinsic tasks with different initializations. The evaluation results of fine-tuned SynGCN embeddings by different methods on SQuAD are shown in the Figure 6.3. The remaining results are included in the supplementary (Table S1 and S2). We observe that compared to other methods, SemGCN is most effective at incorporating semantic constraints across all the initializations and outperforms others at both intrinsic and extrinsic tasks.

Method	POS	SQuAD	NER	Coref
ELMo (E)	96.1±0.1	81.8±0.2	90.3±0.3	67.8±0.1
E+SemGCN(SynGCN, 4)	**96.2±0.1**	**82.4±0.1**	**90.9±0.1**	**68.3±0.1**

Table 6.5: Comparison of ELMo with SynGCN and SemGCN embeddings on multiple extrinsic tasks. For each task, models use a linear combination of the provided embeddings whose weights are learned. Results show that our proposed methods encode complementary information which is not captured by ELMo. Please refer Section 6.4.2.4 for more details.

6.4.2.4 Comparison with ELMo

Recently, ELMo [154] has been proposed which fine-tunes word embedding based on sentential context. In this section, we evaluate SynGCN and SemGCN when given along with pre-trained ELMo embeddings. The results are reported in Table 6.5. The results show that dependency context based embeddings encode complementary information which is not captured by ELMo as it only relies on sequential context. Hence, our proposed methods serves as an effective combination with ELMo.

6.5 Conclusion

In this chapter, we have proposed SynGCN, a graph convolution based approach which utilizes syntactic context for learning word representations. SynGCN overcomes the problem of vocabulary explosion and outperforms state-of-the-art word embedding approaches on several intrinsic and extrinsic tasks. We also propose SemGCN, a framework for jointly incorporating diverse semantic information in pre-trained word embeddings. The combination of SynGCN and SemGCN gives the best overall performance. We make the source code of both models available to encourage reproducible research.

Part III

GNN Enhancements

Chapter 7

Improving Semi-Supervised Learning through Confidence-based Graph Convolutional Networks

7.1 Introduction

In this chapter, we present our first enhancement of Graph Neural Networks for making them more effective at embedding heterogeneous real world graphs. Graphs are all around us, ranging from citation and social networks to knowledge graphs. Predicting properties of nodes in such graphs is often desirable. For example, given a citation network, we may want to predict the research area of an author. Making such predictions, especially in the semi-supervised setting, has been the focus of graph-based semi-supervised learning (SSL) [177]. In graph-based SSL, a small set of nodes are initially labeled. Starting with such supervision and while utilizing the rest of the graph structure, the initially unlabeled nodes are labeled. Conventionally, the graph structure has been incorporated as an explicit regularizer which enforces a smoothness constraint on the labels estimated on nodes [225, 17, 203]. Recently proposed Graph Convolutional Networks (GCN) [46, 87] provide a framework to apply deep neural networks to graph-structured data. GCNs have been employed successfully for improving performance on tasks such as semantic role labeling [121], machine translation [15], relation extraction [193, 223], document dating [191], shape segmentation [216], and action recognition [76]. GCN formulations for graph-based SSL have also attained state-of-the-art performance [87, 106, 195]. In this chapter, we also focus on the task of graph-based SSL using GCNs.

GCN iteratively estimates embedding of nodes in the graph by aggregating embeddings of neighborhood nodes, while backpropagating errors from a target loss function. Finally, the

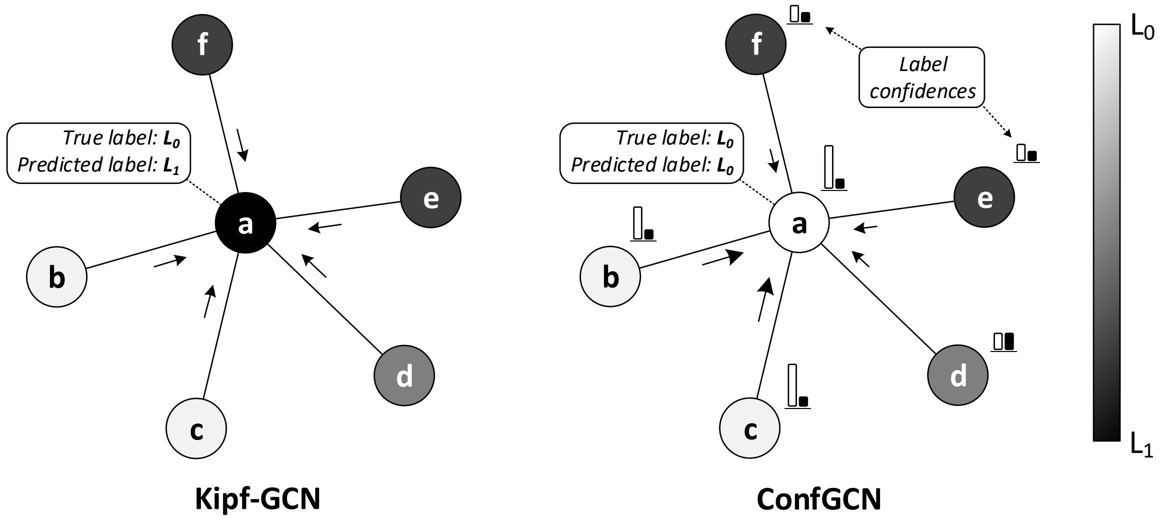


Figure 7.1: Label prediction on node a by Kipf-GCN and ConfGCN (this chapter). L_0 is a's true label. Shade intensity of a node reflects the estimated score of label L_1 assigned to that node. Since Kipf-GCN is not capable of estimating influence of one node on another, it is misled by the dominant label L_1 in node a's neighborhood and thereby making the wrong assignment. ConfGCN, on the other hand, estimates confidences (shown by bars) over the label scores, and uses them to increase influence of nodes b and c to estimate the right label on a. Please see Section 7.1 for details.

learned node embeddings are used to estimate label scores on the nodes. In addition to the label scores, it is desirable to also have confidence estimates associated with them. Such confidence scores may be used to determine how much to trust the label scores estimated on a given node. While methods to estimate label score confidence in non-deep graph-based SSL has been previously proposed [146], confidence-based GCN is still unexplored.

In order to fill this important gap, we propose ConfGCN, a GCN framework for graph-based SSL. ConfGCN jointly estimates label scores on nodes, along with confidences over them. One of the added benefits of confidence over node's label scores is that they may be used to subdue irrelevant nodes in a node's neighborhood, thereby controlling the number of effective neighbors for each node. In other words, this enables *anisotropic* behavior in GCNs. Let us explain this through the example shown in Figure 7.1. In this figure, while node a has true label L_0 (white), it is incorrectly classified as L_1 (black) by Kipf-GCN [87][1]. This is because Kipf-GCN suffers from limitations of its neighborhood aggregation scheme [206]. For example, Kipf-GCN has no constraints on the number of nodes that can influence the representation of a given target node. In a k-layer Kipf-GCN model, each node is influenced by all the nodes in its k-hop neighborhood.

[1]In this chapter, unless otherwise stated, we refer to Kipf-GCN whenever we mention GCN.

However, in real world graphs, nodes are often present in *heterogeneous* neighborhoods, i.e., a node is often surrounded by nodes of other labels. For example, in Figure 7.1, node a is surrounded by three nodes (d, e, and f) which are predominantly labeled L_1, while two nodes (b and c) are labeled L_0. Please note that all of these are estimated label scores during GCN learning. In this case, it is desirable that node a is more influenced by nodes b and c than the other three nodes. However, since Kipf-GCN doesn't discriminate among the neighboring nodes, it is swayed by the majority and thereby estimating the wrong label L_1 for node a.

ConfGCN is able to overcome this problem by estimating confidences on each node's label scores. In Figure 7.1, such estimated confidences are shown by bars, with white and black bars denoting confidences in scores of labels L_0 and L_1, respectively. ConfGCN uses these label confidences to subdue nodes d, e, f since they have low confidence for their label L_1 (shorter black bars), whereas nodes b and c are highly confident about their labels being L_0 (taller white bars). This leads to higher influence of b and c during aggregation, and thereby ConfGCN correctly predicting the true label of node a as L_0 with high confidence. This clearly demonstrates the benefit of label confidences and their utility in estimating node influences. Graph Attention Networks (GAT) [195], a recently proposed method also provides a mechanism to estimate influences by allowing nodes to attend to their neighborhood. However, as we shall see in Section 7.5, ConfGCN, through its use of label confidences, is considerably more effective.

Our contributions in this chapter are as follows.

- We propose ConfGCN, a Graph Convolutional Network (GCN) framework for semi-supervised learning which models label distribution and their confidences for each node in the graph. To the best of our knowledge, this is the first confidence-enabled formulation of GCNs.

- ConfGCN utilize label confidences to estimate influence of one node on another in a label-specific manner during neighborhood aggregation of GCN learning.

- Through extensive evaluation on multiple real-world datasets, we demonstrate ConfGCN effectiveness over state-of-the-art baselines.

ConfGCN's source code and datasets used in the chapter are available at `http://github.com/malllabiisc/ConfGCN`.

7.2 Related Work

Semi-Supervised learning (SSL) on graphs: SSL on graphs is the problem of classifying nodes in a graph, where labels are available only for a small fraction of nodes. Conventionally, the graph structure is imposed by adding an explicit graph-based regularization term in the loss function [225, 203, 17]. Recently, implicit graph regularization via learned node representation has proven to be more effective. This can be done either sequentially or in an end to end fashion. Methods like DeepWalk [153], node2vec [68], and LINE [184] first learn graph representations via sampled random walk on the graph or breadth first search traversal and then use the learned representation for node classification. On the contrary, Planetoid [212] learns node embedding by jointly predicting the class labels and the neighborhood context in the graph. Recently, Kipf and Welling [87] employs Graph Convolutional Networks (GCNs) to learn node representations.

Graph Convolutional Networks (GCNs): The generalization of Convolutional Neural Networks to non-euclidean domains is proposed by Bruna et al. [28] which formulates the spectral and spatial construction of GCNs. This is later improved through an efficient localized filter approximation [46]. Kipf and Welling [87] provide a first-order formulation of GCNs and show its effectiveness for SSL on graphs. Marcheggiani and Titov [121] propose GCNs for directed graphs and provide a mechanism for edge-wise gating to discard noisy edges during aggregation. This is further improved by Veličković et al. [195] which allows nodes to attend to their neighboring nodes, implicitly providing different weights to different nodes. Liao et al. [106] propose Graph Partition Neural Network (GPNN), an extension of GCNs to learn node representations on large graphs. GPNN first partitions the graph into subgraphs and then alternates between locally and globally propagating information across subgraphs. Recently, Lovasz Convolutional Networks Yadav et al. [208] is proposed for incorporating global graph properties in GCNs. An extensive survey of GCNs and their applications can be found in Bronstein et al. [27].

Confidence Based Methods: The natural idea of incorporating confidence in predictions has been explored by Li and Sethi [102] for the task of active learning. Lei [99] proposes a confidence based framework for classification problems, where the classifier consists of two regions in the predictor space, one for confident classifications and other for ambiguous ones. In representation learning, uncertainty (inverse of confidence) is first utilized for word embeddings by Vilnis and McCallum [198]. Athiwaratkun and Wilson [5] further extend this idea to learn hierarchical word representation through encapsulation of probability distributions. Orbach and Crammer [146] propose TACO (Transduction Algorithm with COnfidence), the first graph based method which learns label distribution along with its uncertainty for semi-supervised node classification. Bojchevski and Gnnemann [20] embeds graph nodes as Gaussian distribution using ranking based framework which allows to capture uncertainty of representation. They update node embeddings to maintain neighborhood ordering, i.e. 1-hop neighbors are more similar to 2-hop neighbors and so on. Gaussian embeddings have been used for collaborative filtering [52] and topic modelling [43] as well.

7.3 Notation & Problem Statement

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ be an undirected graph, where $\mathcal{V} = \mathcal{V}_l \cup \mathcal{V}_u$ is the union of labeled ($\mathcal{V}_l$) and unlabeled ($\mathcal{V}_u$) nodes in the graph with cardinalities n_l and n_u, $\mathcal{E}$ is the set of edges and $\mathcal{X} \in \mathbb{R}^{(n_l+n_u)\times d}$ is the input node features. The actual label of a node v is denoted by a one-hot vector $Y_v \in \mathbb{R}^m$, where m is the number of classes. Given $\mathcal{G}$ and seed labels $Y \in \mathbb{R}^{n_l \times m}$, the goal is to predict the labels of the unlabeled nodes. To incorporate confidence, we additionally estimate label distribution $\boldsymbol{\mu}_v \in \mathbb{R}^m$ and a diagonal co-variance matrix $\boldsymbol{\Sigma}_v \in \mathbb{R}^{m\times m}$, $\forall v \in \mathcal{V}$. Here, $\boldsymbol{\mu}_{v,i}$ denotes the score of label i on node v, while $(\boldsymbol{\Sigma}_v)_{ii}$ denotes the variance in the estimation of $\boldsymbol{\mu}_{v,i}$. In other words, $(\boldsymbol{\Sigma}_v^{-1})_{ii}$ is ConfGCN's confidence in $\boldsymbol{\mu}_{v,i}$.

7.4 Proposed Method: Confidence Based Graph Convolutional Networks (ConfGCN)

Following [146], ConfGCN uses co-variance matrix based symmetric Mahalanobis distance for defining distance between two nodes in the graph. Formally, for any two given nodes u and v, with label distributions $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$ and co-variance matrices $\boldsymbol{\Sigma}_u$ and $\boldsymbol{\Sigma}_v$, distance between them is defined as follows.

$$d_M(u, v) = (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v).$$

Characteristic of the above distance metric is that if either of $\boldsymbol{\Sigma}_u$ or $\boldsymbol{\Sigma}_v$ has large eigenvalues, then the distance will be low irrespective of the closeness of $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$. On the other hand, if $\boldsymbol{\Sigma}_u$ and $\boldsymbol{\Sigma}_v$ both have low eigenvalues, then it requires $\boldsymbol{\mu}_u$ and $\boldsymbol{\mu}_v$ to be close for their distance to be low. Given the above properties, we define r_{uv}, the influence score of node u on its neighboring node v during GCN aggregation, as follows.

$$r_{uv} = \frac{1}{d_M(u, v)}.$$

This influence score gives more relevance to neighboring nodes with highly confident similar label, while reducing importance of nodes with low confident label scores. This results in ConfGCN acquiring anisotropic capability during neighborhood aggregation. For a node v,

Dataset	Nodes	Edges	Classes	Features	Label Mismatch	$\frac{\lvert\mathcal{V}_l\rvert}{\lvert\mathcal{V}\rvert}$
Cora	2,708	5,429	7	1,433	0.002	0.052
Cora-ML	2,995	8,416	7	2,879	0.018	0.166
Citeseer	3,327	4,372	6	3,703	0.003	0.036
Pubmed	19,717	44,338	3	500	0.0	0.003

Table 7.1: Details of the datasets used in the chapter. Please refer Section 7.5.1.1 for more details.

ConfGCN's equation for updating embedding at the k-th layer is thus defined as follows.

$$\boldsymbol{h}_v^{k+1} = f\left(\sum_{u\in\mathcal{N}(v)} r_{uv} \times \left(\boldsymbol{W}^k\boldsymbol{h}_u^k + \boldsymbol{b}^k\right)\right), \forall v \in \mathcal{V}. \tag{7.1}$$

The final node representation obtained from ConfGCN is used for predicting labels of the nodes in the graph as follows.

$$\hat{\boldsymbol{Y}}_v = \text{softmax}(\boldsymbol{W}^K\boldsymbol{h}_v^K + \boldsymbol{b}^K),\ \forall v \in \mathcal{V}$$

where, K denotes the number of ConfGCN's layers. Finally, in order to learn label scores $\{\boldsymbol{\mu}_v\}$ and co-variance matrices $\{\boldsymbol{\Sigma}_v\}$ jointly with other parameters $\{\boldsymbol{W}^k, \boldsymbol{b}^k\}$, following Orbach and Crammer [146], we include the following two terms in ConfGCN's objective function.

For enforcing neighboring nodes to be close to each other, we include L_{smooth} defined as

$$L_{\text{smooth}} = \sum_{(u,v)\in\mathcal{E}} (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T(\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v).$$

To impose the desirable property that the label distribution of nodes in $\mathcal{V}_l$ should be close to their input label distribution, we incorporate L_{label} defined as

$$L_{\text{label}} = \sum_{v\in\mathcal{V}_l} (\boldsymbol{\mu}_v - \boldsymbol{Y}_v)^T(\boldsymbol{\Sigma}_v^{-1} + \frac{1}{\gamma}\boldsymbol{I})(\boldsymbol{\mu}_v - \boldsymbol{Y}_v).$$

Here, for input labels, we assume a fixed uncertainty of $\frac{1}{\gamma}\boldsymbol{I} \in \mathbb{R}^{L\times L}$, where $\gamma > 0$. We also include the following regularization term, L_{reg} to constraint the co-variance matrix to be finite and positive.

$$L_{\text{reg}} = \sum_{v\in\mathcal{V}} \text{Tr}\ \max(-\boldsymbol{\Sigma}_v, 0),$$

This regularization term enforces soft positivity constraint on co-variance matrix. Additionally in ConfGCN, we include the L_{const} in the objective, to push the label distribution ($\boldsymbol{\mu}$) close to the final model prediction ($\hat{\boldsymbol{Y}}$).

$$L_{\text{const}} = \sum_{v \in \mathcal{V}} (\boldsymbol{\mu}_v - \hat{\boldsymbol{Y}}_v)^T (\boldsymbol{\mu}_v - \hat{\boldsymbol{Y}}_v).$$

Finally, we include the standard cross-entropy loss for semi-supervised multi-class classification over all the labeled nodes ($\mathcal{V}_l$).

$$L_{\text{cross}} = - \sum_{v \in \mathcal{V}_l} \sum_{j=1}^{m} \boldsymbol{Y}_{vj} \log(\hat{\boldsymbol{Y}}_{vj}).$$

The final objective for optimization is the linear combination of the above defined terms.

$$\begin{aligned} L(\theta) = &- \sum_{i \in \mathcal{V}_l} \sum_{j=1}^{L} \boldsymbol{Y}_{ij} \log(\hat{\boldsymbol{Y}}_{ij}) \\ &+ \lambda_1 \sum_{(u,v) \in \mathcal{E}} (\boldsymbol{\mu}_u - \boldsymbol{\mu}_v)^T (\boldsymbol{\Sigma}_u^{-1} + \boldsymbol{\Sigma}_v^{-1})(\boldsymbol{\mu}_u - \boldsymbol{\mu}_v) \\ &+ \lambda_2 \sum_{u \in \mathcal{V}_l} (\boldsymbol{\mu}_u - \boldsymbol{Y}_u)^T (\boldsymbol{\Sigma}_u^{-1} + \frac{1}{\gamma}\boldsymbol{I})(\boldsymbol{\mu}_u - \boldsymbol{Y}_u) \\ &+ \lambda_3 \sum_{v \in \mathcal{V}} (\boldsymbol{\mu}_u - \hat{\boldsymbol{Y}}_u)^T (\boldsymbol{\mu}_u - \hat{\boldsymbol{Y}}_u) \\ &+ \lambda_4 \sum_{v \in \mathcal{V}} \text{Tr} \max(-\boldsymbol{\Sigma}_v, 0) \end{aligned}$$

where, $\theta = \{\boldsymbol{W}^k, \boldsymbol{b}^k, \boldsymbol{\mu}_v, \boldsymbol{\Sigma}_v\}$ and $\lambda_i \in \mathbb{R}$, are the weights of the terms in the objective. We optimize $L(\theta)$ using stochastic gradient descent. We hypothesize that all the terms help in improving ConfGCN's performance and we validate this in Section 7.5.2.4.

7.5 Experiments

7.5.1 Experimental Setup

7.5.1.1 Datasets

For evaluating the effectiveness of ConfGCN, we evaluate on several semi-supervised classification benchmarks. Following the experimental setup of [87, 106], we evaluate on Cora, Citeseer, and Pubmed datasets [166]. The dataset statistics is summarized in Table 7.1. Label mismatch denotes the fraction of edges between nodes with different labels in the training data. The benchmark datasets commonly used for semi-supervised classification task have substantially

Method	Citeseer	Cora	Pubmed	Cora ML
LP [225]	45.3	68.0	63.0	-
ManiReg [17]	60.1	59.5	70.7	-
SemiEmb [203]	59.6	59.0	71.1	-
Feat [212]	57.2	57.4	69.8	-
DeepWalk [153]	43.2	67.2	65.3	-
GGNN [104]	68.1	77.9	77.2	-
Planetoid [212]	64.9	75.7	75.7	-
Kipf-GCN [87]	69.4 ± 0.4	80.9 ± 0.4	76.8 ± 0.2	85.7 ± 0.3
G-GCN [121]	69.6 ± 0.5	81.2 ± 0.4	77.0 ± 0.3	86.0 ± 0.2
GPNN [106]	68.1 ± 1.8	79.0 ± 1.7	73.6 ± 0.5	69.4 ± 2.3
GAT [195]	72.5 ± 0.7	**83.0** ± **0.7**	79.0 ± 0.3	83.0 ± 0.8
ConfGCN (this work)	**72.7** ± **0.8**	82.0 ± 0.3	**79.5** ± **0.5**	**86.5** ± **0.3**

Table 7.2: Performance comparison of several methods for semi-supervised node classification on multiple benchmark datasets. ConfGCN performs consistently better across all the datasets. Baseline method performances on Citeseer, Cora and Pubmed datasets are taken from Liao et al. [106], Veličković et al. [195]. We consider only the top performing baseline methods on these datasets for evaluation on the Cora-ML dataset. Please refer Section 7.5.2.1 for details.

low label mismatch rate. In order to examine models on datasets with more heterogeneous neighborhoods, we also evaluate on Cora-ML dataset [20].

All the four datasets are citation networks, where each document is represented using bag-of-words features in the graph with undirected citation links between documents. The goal is to classify documents into one of the predefined classes. We use the data splits used by [212] and follow similar setup for Cora-ML dataset. Following [87], additional 500 labeled nodes are used for hyperparameter tuning.

Hyperparameters: We use the same data splits as described in [212], with a test set of 1000 labeled nodes for testing the prediction accuracy of ConfGCN and a validation set of 500 labeled nodes for optimizing the hyperparameters. The ranges of hyperparameters were adapted from previous literature [146, 87]. The model is trained using Adam [86] with a learning rate of 0.01. The weight matrices along with μ are initialized using Xavier initialization [65] and $\mathbf{\Sigma}$ matrix is initialized with identity. To avoid numerical instability we model $\mathbf{\Sigma^{-1}}$ directly and compute $\mathbf{\Sigma}$ wherever required. Following Kipf and Welling [87], we use two layers of GCN (K) for all the experiments in this chapter.

7.5.1.2 Baselines

For evaluating ConfGCN, we compare against the following baselines:

- **Feat** [212] takes only node features as input and ignores the graph structure.
- **ManiReg** [17] is a framework for providing data-dependent geometric regularization.
- **SemiEmb** [203] augments deep architectures with semi-supervised regularizers to improve their training.
- **LP** [225] is an iterative iterative label propagation algorithm which propagates a nodes labels to its neighboring unlabeled nodes according to their proximity.
- **DeepWalk** [153] learns node features by treating random walks in a graph as the equivalent of sentences.
- **Planetoid** [212] provides a transductive and inductive framework for jointly predicting class label and neighborhood context of a node in the graph.
- **GCN** [87] is a variant of convolutional neural networks used for semi-supervised learning on graph-structured data.
- **G-GCN** [121] is a variant of GCN with edge-wise gating to discard noisy edges during aggregation.
- **GGNN** [104] is a generalization of RNN framework which can be used for graph-structured data.
- **GPNN** [106] is a graph partition based algorithm which propagates information after partitioning large graphs into smaller subgraphs.
- **GAT** [195] is a graph attention based method which provides different weights to different nodes by allowing nodes to attend to their neighborhood.

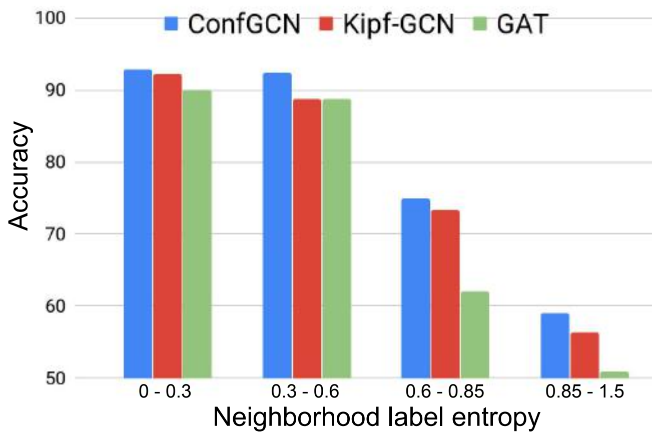
7.5.2 Results

In this section, we attempt to answer the following questions:

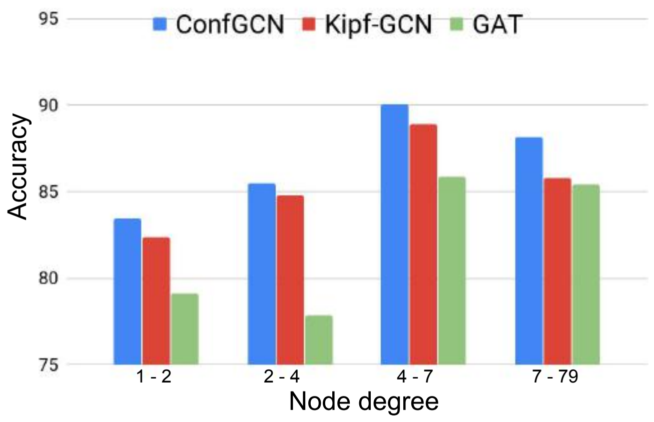
Q1. How does ConfGCN compare against existing methods for the semi-supervised node classification task? (Section 7.5.2.1)

Q2. How do the performance of methods vary with increasing node degree and neighborhood label mismatch? (Section 7.5.2.2)

Q3. How does increasing the number of layers effect ConfGCN's performance? (Section 7.5.2.3)



(a)



(b)

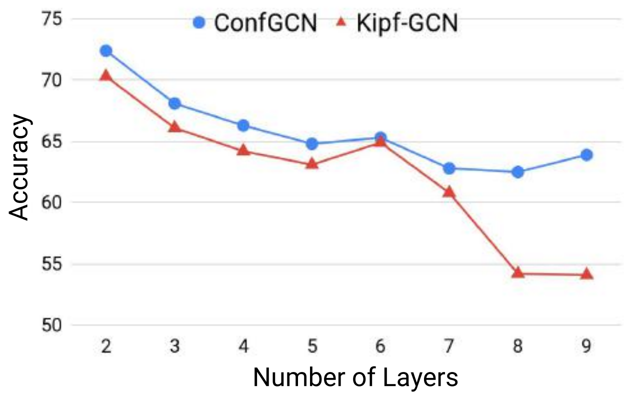
Figure 7.2: Plots of node classification accuracy vs. (a) neighborhood label entropy and (b) node degree. On x-axis, we plot quartiles of (a) neighborhood label entropy and (b) degree, i.e., each bin has 25% of the samples in sorted order. Overall, we observe that ConfGCN performs better than Kipf-GCN and GAT at all levels of node entropy and degree. Please see Section 7.5.2.2 for details.

Q4. What is the effect of ablating different terms in ConfGCN's loss function? (Section 7.5.2.4)

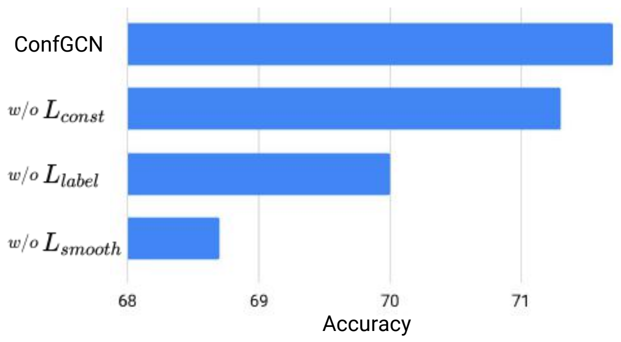
7.5.2.1 Node Classification

The evaluation results for semi-supervised node classification are summarized in Table 7.2. Results of all other baseline methods on Cora, Citeseer and Pubmed datasets are taken from [106, 195] directly. For evaluation on the Cora-ML dataset, only top performing baselines from the other three datasets are considered. Overall, we find that ConfGCN outperforms all existing approaches consistently across all the datasets.

This may be attributed to ConfGCN's ability to model nodes' label distribution along with the confidence scores which subdues the effect of noisy nodes during neighborhood aggregation. The lower performance of GAT [195] compared to Kipf-GCN on Cora-ML shows that computing attention based on the hidden representation of nodes is not much helpful in suppressing noisy neighborhood nodes. We also observe that the performance of GPNN [106] suffers on the Cora-ML dataset. This is due to the fact that while propagating information between small subgraphs, the high label mismatch rate in Cora-ML (please see Table 7.1) leads to wrong information propagation. Hence, during the global propagation step, this error is further magnified.



(a) Evaluation of Kipf-GCN and ConfGCN on the citeseer dataset with increasing number of GCN layers. Overall, ConfGCN outperforms Kipf-GCN, and while both methods' performance degrade with increasing layers, ConfGCN's degradation is more gradual than Kipf-GCN's abrupt drop. Please see Section 7.5.2.3 for details.



(b) Performance comparison of different ablated version of ConfGCN on the citeseer dataset. These results justify inclusion of the different terms in ConfGCN's objective function. Please see Section 7.5.2.4 for details.

7.5.2.2 Effect of Node Entropy and Degree on Performance

In this section, we provide an analysis of the performance of Kipf-GCN, GAT and ConfGCN for node classification on the Cora-ML dataset which has higher label mismatch rate. We use neighborhood label entropy to quantify label mismatch, which for a node u is defined as follows.

$$\text{NeighborLabelEntropy}(u) = -\sum_{l=1}^{L} p_{ul} \log p_{ul}$$

$$\text{where,}\ \ p_{ul} = \frac{|\{v \in \mathcal{N}(u) \mid \text{label}(v) = l\}|}{|\mathcal{N}(u)|}.$$

Here, label(v) is the true label of node v. The results for neighborhood label entropy and node degree are summarized in Figures 7.2a and 7.2b, respectively. On the x-axis of these figures, we plot quartiles of label entropy and degree, i.e., each bin has 25% of the instances in sorted order. With increasing neighborhood label entropy, the node classification task is expected to become more challenging. We indeed see this trend in Figure 7.2a where performances of all the methods degrade with increasing neighborhood label entropy. However, ConfGCN performs comparatively better than the existing state-of-art approaches at all levels of node entropy.

In case of node degree also (Figure 7.2b), we find that ConfGCN performs better than Kipf-GCN and GAT at all quartiles of node degrees. Classifying sparsely connected nodes (first and second bins) is challenging as very little information is present in the node neighborhood. Performance improves with availability of moderate number of neighboring nodes (third bin), but further increase in degree (fourth bin) results in introduction of many potentially noisy neighbors, thereby affecting performance of all the methods. For higher degree nodes, ConfGCN gives an improvement of around 3% over GAT and Kipf-GCN. This shows that ConfGCN, through its use of label confidences, is able to give higher influence score to relevant nodes in the neighborhood during aggregation while reducing importance of the noisy ones.

7.5.2.3 Effect of Increasing Convolutional Layers

Recently, Xu et al. [206] highlighted an unusual behavior of Kipf-GCN where its performance degrades significantly with increasing number of layers. This is because of increase in the number of influencing nodes with increasing layers, resulting in "averaging out" of information during aggregation. For comparison, we evaluate the performance of Kipf-GCN and ConfGCN on citeseer dataset with increasing number of convolutional layers. The results are summarized in Figure 7.3a. We observe that Kipf-GCN's performance degrades drastically with increasing number of layers, whereas ConfGCN's decrease in performance is more gradual. This shows that confidence based GCN helps in alleviating this problem. We also note that ConfGCN outperforms Kipf-GCN at all layer levels.

7.5.2.4 Ablation Results

In this section, we evaluate the different ablated version of ConfGCN by cumulatively eliminating terms from its objective function as defined in Section 7.4. The results on citeseer dataset are summarized in Figure 7.3b. Overall, we find that each term ConfGCN's loss function (Equation 7.4) helps in improving its performance and the method performs best when all the terms are included.

7.6 Conclusion

In this chapter, we present ConfGCN, a confidence based Graph Convolutional Network which estimates label scores along with their confidences jointly in a GCN-based setting. In ConfGCN, the influence of one node on another during aggregation is determined using the estimated confidences and label scores, thus inducing anisotropic behavior to GCN. We demonstrate the effectiveness of ConfGCN against state-of-the-art methods for the semi-supervised node classification task and analyze its performance in different settings. We make ConfGCN's source code available.

Chapter 8

Composition-based Multi-Relational Graph Convolutional Networks for Relational Graphs

8.1 Introduction

In this chapter, we address another important limitation of Graph Convolutional models. Most of the existing research on GCNs [87, 70, 195] have focused on learning representations of nodes in simple undirected graphs. A more general and pervasive class of graphs are multi-relational graphs[1]. A notable example of such graphs is knowledge graphs. Most of the existing GCN based approaches for handling relational graphs [121, 165] suffer from over-parameterization and are limited to learning only node representations. Hence, such methods are not directly applicable for tasks that require relation embedding vectors such as link prediction. Initial attempts at learning representations for relations in graphs [133, 16] have shown some performance gains on tasks like node classification and neural machine translation.

There has been extensive research on embedding Knowledge Graphs (KG) [140, 201] where representations of both nodes and relations are jointly learned. These methods are restricted to learning embeddings using link prediction objective. Even though GCNs can learn from task-specific objectives such as classification, their application has been largely restricted to non-relational graph setting. Thus, there is a need for a framework which can utilize KG embedding techniques for learning task-specific node and relation embeddings. In this chapter, we propose CompGCN, a novel GCN framework for multi-relational graphs which systematically

[1]In this chapter, multi-relational graphs refer to graphs with edges that have labels and directions.

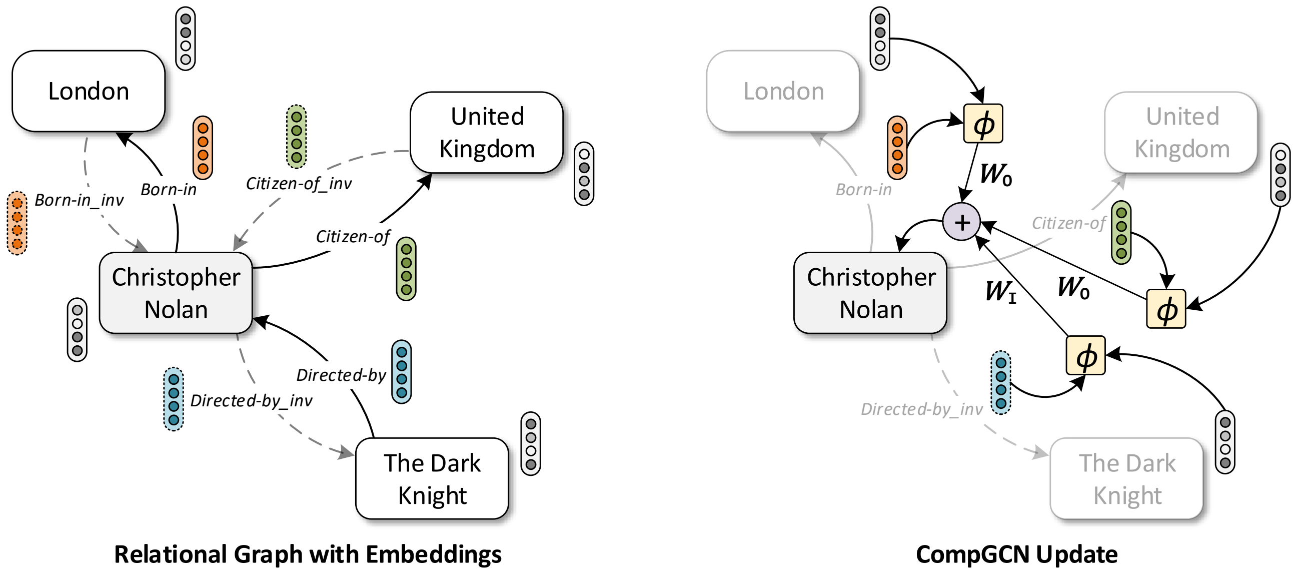


Figure 8.1: Overview of CompGCN. Given node and relation embeddings, CompGCN performs a composition operation $\phi(\cdot)$ over each edge in the neighborhood of a central node (e.g. *Christopher Nolan* above). The composed embeddings are then convolved with specific filters $\boldsymbol{W}_O$ and $\boldsymbol{W}_I$ for original and inverse relations respectively. We omit self-loop in the diagram for clarity. The message from all the neighbors are then aggregated to get an updated embedding of the central node. Also, the relation embeddings are transformed using a separate weight matrix. Please refer to Section 8.3 for details.

leverages entity-relation composition operations from knowledge graph embedding techniques. CompGCN addresses the shortcomings of previously proposed GCN models by jointly learning vector representations for both nodes and relations in the graph. An overview of CompGCN is presented in Figure 8.1. The contributions of our work can be summarized as follows:

1. We propose CompGCN, a novel framework for incorporating multi-relational information in Graph Convolutional Networks which leverages a variety of composition operations from knowledge graph embedding techniques to jointly embed both nodes and relations in a graph.
2. We demonstrate that CompGCN framework generalizes several existing multi-relational GCN methods (Proposition 8.1) and also scales with the increase in number of relations in the graph (Section 8.4.2.3).
3. Through extensive experiments on tasks such as node classification, link prediction, and graph classification, we demonstrate the effectiveness of our proposed method.

8.2 Related Work

Graph Convolutional Networks: GCNs generalize Convolutional Neural Networks (CNNs) to non-Euclidean data. GCNs were first introduced by [28] and later made scalable through efficient localized filters in the spectral domain [46]. A first-order approximation of GCNs using Chebyshev polynomials has been proposed by [87]. Recently, several of its extensions have also been formulated [70, 195, 207]. Most of the existing GCN methods follow *Message Passing Neural Networks* (MPNN) framework [64] for node aggregation. Our proposed method can be seen as an instantiation of the MPNN framework. However, it is specialized for relational graphs.

GCNs for Multi-Relational Graph: An extension of GCNs for relational graphs is proposed by [121]. However, they only consider direction-specific filters and ignore relations due to over-parameterization. [165] address this shortcoming by proposing basis and block-diagonal decomposition of relation specific filters. *Weighted Graph Convolutional Network* [167] utilizes learnable relational specific scalar weights during GCN aggregation. While these methods show performance gains on node classification and link prediction, they are limited to embedding only the nodes of the graph. Contemporary to our work, [215] have also proposed an extension of GCNs for embedding both nodes and relations in multi-relational graphs. However, our proposed method is a more generic framework which can leverage any KG composition operator. We compare against their method in Section 8.4.2.1.

Knowledge Graph Embedding: Knowledge graph (KG) embedding is a widely studied field [140, 201] with application in tasks like link prediction and question answering [23]. Most of KG embedding approaches define a score function and train node and relation embeddings such that valid triples are assigned a higher score than the invalid ones. Based on the type of score function, KG embedding method are classified as translational [22, 202], semantic matching based [211, 142] and neural network based [172, 49]. In our work, we evaluate the performance of CompGCN on link prediction with methods of all three types.

8.3 CompGCN Details

In this section, we provide a detailed description of our proposed method, CompGCN. The overall architecture is shown in Figure 8.1. We represent a multi-relational graph by $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Z}})$ as defined in Section 2.5 where $\boldsymbol{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{R}| \times d_0}$ denotes the initial relation features. Our model is motivated by the first-order approximation of GCNs using Chebyshev polynomials [87]. Following Marcheggiani and Titov [121], we also allow the information in a directed edge to flow along both directions. Hence, we extend $\mathcal{E}$ and $\mathcal{R}$ with corresponding inverse edges and relations, i.e.,

$$\mathcal{E}' = \mathcal{E} \cup \{(v, u, r^{-1}) \mid (u, v, r) \in \mathcal{E}\} \cup \{(u, u, \top) \mid u \in \mathcal{V})\},$$

and $\mathcal{R}' = \mathcal{R} \cup \mathcal{R}_{inv} \cup \{\top\}$, where $\mathcal{R}_{inv} = \{r^{-1} \,|\, r \in \mathcal{R}\}$ denotes the inverse relations and $\top$ indicates the self loop.

8.3.1 Relation-based Composition

Unlike most of the existing methods which embed only nodes in the graph, CompGCN learns a d-dimensional representation $\boldsymbol{h}_r \in \mathbb{R}^d, \forall r \in \mathcal{R}$ along with node embeddings $\boldsymbol{h}_v \in \mathbb{R}^d, \forall v \in \mathcal{V}$. Representing relations as vectors alleviates the problem of over-parameterization while applying GCNs on relational graphs. Further, it allows CompGCN to exploit any available relation features ($\boldsymbol{\mathcal{Z}}$) as initial representations. To incorporate relation embeddings into the GCN formulation, we leverage the entity-relation composition operations used in Knowledge Graph embedding approaches [22, 140], which are of the form

$$\boldsymbol{e}_o = \phi(\boldsymbol{e}_s, \boldsymbol{e}_r).$$

Here, $\phi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ is a composition operator, s, r, and o denote subject, relation and object in the knowledge graph and $\boldsymbol{e}_{(\cdot)} \in \mathbb{R}^d$ denotes their corresponding embeddings. In this chapter, we restrict ourselves to non-parameterized operations like subtraction [22], multiplication [211] and circular-correlation [142]. However, CompGCN can be extended to parameterized operations like Neural Tensor Networks (NTN) [172] and ConvE [49]. We defer their analysis as future work.

As we show in Section 8.4.2, the choice of composition operation is important in deciding the quality of the learned embeddings. Hence, superior composition operations for Knowledge Graphs developed in future can be adopted to improve CompGCN's performance further.

8.3.2 CompGCN Update Equation

The GCN update equation (Eq. 2.5)) defined in Section 2.4 can be re-written as

$$\boldsymbol{h}_v = f\left(\sum_{(u,r) \in \mathcal{N}(v)} \boldsymbol{W}_r \boldsymbol{h}_u \right),$$

where $\mathcal{N}(v)$ is a set of immediate neighbors of v for its outgoing edges. Since this formulation suffers from over-parameterization, in CompGCN we perform composition (ϕ) of a neighboring node u with respect to its relation r as defined above. This allows our model to be relation aware while being linear ($\mathcal{O}(|\mathcal{R}|d)$) in the number of feature dimensions. Moreover, for treating original, inverse, and self edges differently, we define separate filters for each of them. The update equation of CompGCN is given as:

$$\boldsymbol{h}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}_{\lambda(r)} \phi(\boldsymbol{x}_u, \boldsymbol{z}_r) \right), \tag{8.1}$$

where $\boldsymbol{x}_u, \boldsymbol{z}_r$ denotes initial features for node u and relation r respectively, $\boldsymbol{h}_v$ denotes the updated representation of node v, and $\boldsymbol{W}_{\lambda(r)} \in \mathbb{R}^{d_1\times d_0}$ is a relation-type specific parameter. In CompGCN, we use direction specific weights, i.e., $\lambda(r) = \text{dir}(r)$, given as:

$$\boldsymbol{W}_{\text{dir}(r)} = \begin{cases} \boldsymbol{W}_O, & r \in \mathcal{R} \\ \boldsymbol{W}_I, & r \in \mathcal{R}_{inv} \\ \boldsymbol{W}_S, & r = \top \ \textit{(self-loop)} \end{cases} \tag{8.2}$$

Further, in CompGCN, after the node embedding update defined in Eq. 8.1, the relation embeddings are also transformed as follows:

$$\boldsymbol{h}_r = \boldsymbol{W}_{\text{rel}} \boldsymbol{z}_r, \tag{8.3}$$

where $\boldsymbol{W}_{\text{rel}} \in \mathbb{R}^{d_1\times d_0}$ is a learnable transformation matrix which projects the relations to the same embedding space as nodes and allows them to be utilized in the next CompGCN layer.

To ensure that CompGCN scales with the increasing number of relations, we use a variant of the basis formulations proposed in Schlichtkrull et al. [165]. Instead of independently defining an embedding for each relation, they are expressed as a linear combination of a set of basis vectors. Formally, let $\{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{\mathcal{B}}\}$ be a set of learnable basis vectors. Then, initial relation representation is given as:

$$\boldsymbol{z}_r = \sum_{b=1}^{\mathcal{B}} \alpha_{b_r} \boldsymbol{v}_b.$$

Here, $\alpha_{b_r} \in \mathbb{R}$ is relation and basis specific learnable scalar weight. Note that this is different from the formulation in Schlichtkrull et al. [165], where a separate set of basis matrices is defined for each GCN layer. In CompGCN, basis vectors are defined only for the first layer, and the later layers share the relations through transformations according to Equation 8.3.

We can extend the formulation of Equation 8.1 to the case where we have k-stacked CompGCN layers. Let $\boldsymbol{h}_v^{k+1}$ denote the representation of a node v obtained after k layers

Methods	$\boldsymbol{W}^k_{\lambda(r)}$	$\phi(\boldsymbol{h}^k_u, \boldsymbol{h}^k_r)$
Kipf-GCN [87]	$\boldsymbol{W}^k$	$\boldsymbol{h}^k_u$
Relational-GCN [165]	$\boldsymbol{W}^k_r$	$\boldsymbol{h}^k_u$
Directed-GCN [121]	$\boldsymbol{W}^k_{\text{dir}(r)}$	$\boldsymbol{h}^k_u$
Weighted-GCN [167]	$\boldsymbol{W}^k$	$\alpha^k_r \boldsymbol{h}^k_u$

Table 8.1: Reduction of CompGCN to several existing Graph Convolutional methods. Here, α^k_r is a relation specific scalar, $\boldsymbol{W}^k_r$ denotes a separate weight for each relation, and $\boldsymbol{W}^k_{\text{dir}(r)}$ is as defined in Equation 8.2. Please refer to Proposition 8.1 for more details.

which is defined as

$$\boldsymbol{h}^{k+1}_v = f\left(\sum_{(u,r)\in\mathcal{N}(v)} \boldsymbol{W}^k_{\lambda(r)} \phi(\boldsymbol{h}^k_u, \boldsymbol{h}^k_r) \right). \tag{8.4}$$

Similarly, let $\boldsymbol{h}^{k+1}_r$ denote the representation of a relation r after k layers. Then,

$$\boldsymbol{h}^{k+1}_r = \boldsymbol{W}^k_{\text{rel}}\ \boldsymbol{h}^k_r.$$

Here, $\boldsymbol{h}^0_v$ and $\boldsymbol{h}^0_r$ are the initial node ($\boldsymbol{x}_v$) and relation ($\boldsymbol{z}_r$) features respectively.

Proposition 8.1 CompGCN *generalizes the following Graph Convolutional based methods:* ***Kipf-GCN*** *[87],* ***Relational GCN*** *[165],* ***Directed GCN*** *[121], and* ***Weighted GCN*** *[167].*

Proof: For Kipf-GCN, this can be trivially obtained by making weights ($\boldsymbol{W}_{\lambda(r)}$) and composition function (ϕ) relation agnostic in Equation 8.4, i.e., $\boldsymbol{W}_{\lambda(r)} = \boldsymbol{W}$ and $\phi(\boldsymbol{h}_u, \boldsymbol{h}_r) = \boldsymbol{h}_u$. Similar reductions can be obtained for other methods as shown in Table 8.1. □

8.4 Experiments

8.4.1 Experimental Setup

8.4.1.1 Evaluation tasks

In our experiments, we evaluate CompGCN on the below-mentioned tasks.

- **Link Prediction** is the task of inferring missing facts based on the known facts in Knowledge Graphs. In our experiments, we utilize FB15k-237 [186] and WN18RR [49] datasets for evaluation. Following [22], we use filtered setting for evaluation and report Mean Reciprocal Rank (MRR), Mean Rank (MR) and Hits@N.

	Link Prediction		Node Classification		Graph Classification	
	FB15k-237	WN18RR	MUTAG (Node)	AM	MUTAG (Graph)	PTC
Graphs	1	1	1	1	188	344
Entities	14,541	40,943	23,644	1,666,764	17.9 (Avg)	25.5 (Avg)
Edges	310,116	93,003	74,227	5,988,321	39.6 (Avg)	29.5 (Avg)
Relations	237	11	23	133	4	4
Classes	-	-	2	11	2	2

Table 8.2: The details of the datasets used for node classification, link prediction, and graph classification tasks. Please refer to Section 8.4.1.1 for more details.

- **Node Classification** is the task of predicting the labels of nodes in a graph-based on node features and their connections. Similar to [165], we evaluate CompGCN on MUTAG (Node) and AM [162] datasets.
- **Graph Classification**, where, given a set of graphs and their corresponding labels, the goal is to learn a representation for each graph which is fed to a classifier for prediction. We evaluate on 2 bioinformatics dataset: MUTAG (Graph) and PTC [210].

A summary statistics of the datasets used is provided in Table 8.2.

8.4.1.2 Baselines

Across all tasks, we compare against the following GCN methods for relational graphs: (1) Relational-GCN (**R-GCN**) [165] which uses relation-specific weight matrices that are defined as a linear combinations of a set of basis matrices. (2) Directed-GCN (**D-GCN**) [121] has separate weight matrices for incoming edges, outgoing edges, and self-loops. It also has relation-specific biases. (3) Weighted-GCN (**W-GCN**) [167] assigns a learnable scalar weight to each relation and multiplies an incoming "message" by this weight. Apart from this, we also compare with several task-specific baselines mentioned below.

Link prediction: For evaluating CompGCN, we compare against several non-neural and neural baselines: TransE [22], DistMult [211], ComplEx [188], R-GCN [165], KBGAN [30], ConvE [49], ConvKB [138], SACN [167], HypER [9], RotatE [179], ConvR [80], and VR-GCN [215].

Node and Graph Classification: For node classification, following [165], we compare with Feat [149], WL [168], and RDF2Vec [162]. Finally, for graph classification, we evaluate against PachySAN [143], Deep Graph CNN (DGCNN) [222], and Graph Isomorphism Network (GIN) [207].

	FB15k-237					WN18RR				
	MRR	MR	H@10	H@3	H@1	MRR	MR	H@10	H@3	H@1
TransE [22]	.294	357	.465	-	-	.226	3384	.501	-	-
DistMult [211]	.241	254	.419	.263	.155	.43	5110	.49	.44	.39
ComplEx [188]	.247	339	.428	.275	.158	.44	5261	.51	.46	.41
R-GCN [165]	.248	-	.417		.151	-	-	-		-
KBGAN [30]	.278	-	.458		-	.214	-	.472	-	-
ConvE [49]	.325	244	.501	.356	.237	.43	4187	.52	.44	.40
ConvKB [138]	.243	311	.421	.371	.155	.249	3324	.524	.417	.057
SACN [167]	.35	-	.54	.39	.26	.47	-	.54	.48	.43
HypER [9]	.341	250	.520	.376	.252	.465	5798	.522	.477	.436
RotatE [179]	.338	**177**	.533	.375	.241	.476	**3340**	**.571**	.492	.428
ConvR [80]	.350	-	.528	.385	.261	.475	-	.537	.489	**.443**
VR-GCN [215]	.248	-	.432	.272	.159	-	-	-	-	-
CompGCN (Proposed Method)	**.355**	197	**.535**	**.390**	**.264**	**.479**	3533	.546	**.494**	**.443**

Table 8.3: Link prediction performance of CompGCN and several recent models on FB15k-237 and WN18RR datasets. The results of all the baseline methods are taken directly from the previous papers. We find that CompGCN outperforms all the existing methods on 4 out of 5 metrics on FB15k-237 and 3 out of 5 metrics on WN18RR. Please refer to Section 8.4.2.1 for more details.

8.4.2 Results

In this section, we attempt to answer the following questions.

Q1. How does CompGCN perform on link prediction compared to existing methods? (8.4.2.1)

Q2. What is the effect of using different GCN encoders and choice of the compositional operator in CompGCN on link prediction performance? (8.4.2.1)

Q3. Does CompGCN scale with the number of relations in the graph? (8.4.2.3)

Q4. How does CompGCN perform on node and graph classification tasks? (8.4.2.4)

8.4.2.1 Performance Comparison on Link Prediction

In this section, we evaluate the performance of CompGCN and the baseline methods listed in Section 8.4.1.2 on link prediction task. The results on FB15k-237 and WN18RR datasets are presented in Table 8.3. The scores of baseline methods are taken directly from the previous papers [179, 30, 167, 9, 80, 215]. However, for ConvKB, we generate the results using the corrected evaluation code[1]. Overall, we find that CompGCN outperforms all the existing methods in 4 out of 5 metrics on FB15k-237 and in 3 out of 5 metrics on WN18RR dataset. We note that the best performing baseline RotatE uses rotation operation in complex domain. The same operation can be utilized in a complex variant of our proposed method to improve

[1]https://github.com/KnowledgeBaseCompleter/eval-ConvKB

Scoring Function (=X) →	TransE			DistMult			ConvE		
Methods ↓	MRR	MR	H@10	MRR	MR	H@10	MRR	MR	H@10
X	0.294	357	0.465	0.241	354	0.419	0.325	244	0.501
X + D-GCN	0.299	351	0.469	0.321	225	0.497	0.344	200	0.524
X + R-GCN	0.281	325	0.443	0.324	230	0.499	0.342	197	0.524
X + W-GCN	0.267	1520	0.444	0.324	229	0.504	0.344	201	0.525
X + CompGCN (Sub)	0.335	**194**	0.514	0.336	231	0.513	0.352	199	0.530
X + CompGCN (Mult)	**0.337**	233	0.515	**0.338**	**200**	**0.518**	0.353	216	0.532
X + CompGCN (Corr)	0.336	214	**0.518**	0.335	227	0.514	$\boxed{\mathbf{0.355}}$	197	$\boxed{\mathbf{0.535}}$
X + CompGCN ($\mathcal{B} = 50$)	0.330	203	0.502	0.333	210	0.512	0.350	$\boxed{\mathbf{193}}$	0.530

Table 8.4: Performance on link prediction task evaluated on FB15k-237 dataset. X + M (Y) denotes that method M is used for obtaining entity (and relation) embeddings with X as the scoring function. In the case of CompGCN, Y denotes the composition operator used. $\mathcal{B}$ indicates the number of relational basis vectors used. Overall, we find that CompGCN outperforms all the existing methods across different scoring functions. ConvE + CompGCN (Corr) gives the best performance across all settings (highlighted using $\boxed{\cdot}$). Please refer to Section 8.4.2.1 for more details.

its performance further. We defer this as future work.

8.4.2.2 Comparison of Different GCN Encoders on Link Prediction Performance

Next, we evaluate the effect of using different GCN methods as an encoder along with a representative score function (shown in Figure 8.2) from each category: TransE (translational), DistMult (semantic-based), and ConvE (neural network-based). In our results, **X + M (Y)** denotes that method **M** is used for obtaining entity embeddings (and relation embeddings in the case of CompGCN) with **X** as the score function as depicted in Figure 8.2. **Y** denotes the composition operator in the case of CompGCN. We evaluate CompGCN on three non-parametric composition operators inspired from TransE [22], DistMult [211], and HolE [142] defined as

- **Subtraction (Sub):** $\phi(\boldsymbol{e}_s, \boldsymbol{e}_r) = \boldsymbol{e}_s - \boldsymbol{e}_r$.
- **Multiplication (Mult):** $\phi(\boldsymbol{e}_s, \boldsymbol{e}_r) = \boldsymbol{e}_s * \boldsymbol{e}_r$.
- **Circular-correlation (Corr):** $\phi(\boldsymbol{e}_s, \boldsymbol{e}_r) = \boldsymbol{e}_s \star \boldsymbol{e}_r$

The overall results are summarized in Table 8.4. Similar to Schlichtkrull et al. [165], we find that utilizing Graph Convolutional based method as encoder gives a substantial improvement in performance for most types of score functions. We observe that although all the baseline GCN methods lead to some degradation with TransE score function, no such behavior is observed for CompGCN. On average, CompGCN obtains around 6%, 4% and 3% relative increase in MRR with TransE, DistMult, and ConvE objective respectively compared to the best performing

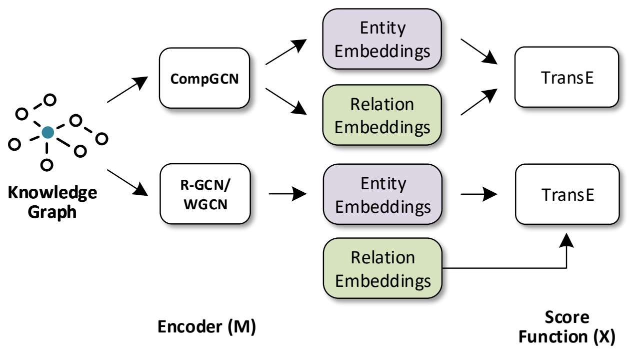


Figure 8.2: Knowledge Graph link prediction with CompGCN and other methods. CompGCN generates both entity and relation embedding as opposed to just entity embeddings for other models. For more details, please refer to Section 8.4.2.2

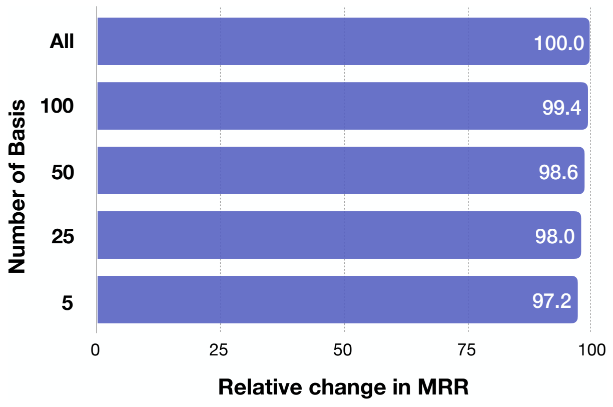


Figure 8.3: Performance of CompGCN with different number of relation basis vectors on link prediction task. We report the relative change in MRR on FB15k-237 dataset. Overall, CompGCN gives comparable performance even with limited parameters. Refer to Section 8.4.2.3 for details.

baseline. The superior performance of CompGCN can be attributed to the fact that it learns both entity and relation embeddings jointly thus providing more expressive power in learned representations. Overall, we find that CompGCN with ConvE (highlighted using $\boxed{\cdot}$) is the best performing method for link prediction.

Effect of composition Operator: The results on link prediction with different composition operators are presented in Table 8.4. We find that with DistMult score function, multiplication operator (Mult) gives the best performance while with ConvE, circular-correlation surpasses all other operators. Overall, we observe that more complex operators like circular-correlation outperform or perform comparably to simpler operators such as subtraction.

8.4.2.3 Parameter Efficiency of CompGCN

In this section, we analyze the performance of CompGCN on changing the number of relation basis vectors ($\mathcal{B}$) as defined in Section 8.3. For this, we evaluate the best performing model for link prediction (ConvE + CompGCN (Corr)) with a variable number of basis vectors. The results are summarized in Figure 8.3. We find that our model performance improves with the increasing number of basis vectors. We note that with $\mathcal{B} = 100$, the performance of the model becomes comparable to the case where all relations have their individual embeddings. In Table 8.4, we report the results for the best performing model across all score function with $\mathcal{B}$ set to 50. We note that the parameter-efficient variant also gives a comparable performance and outperforms the baselines in all settings. This demonstrates that CompGCN is scalable with

	MUTAG (Node)	**AM**
Feat*	77.9	66.7
WL*	80.9	87.4
RDF2Vec*	67.2	88.3
R-GCN*	73.2	89.3
SynGCN	74.8 ± 5.5	86.2 ± 1.9
WGCN	77.9 ± 3.2	90.2 ± 0.9
CompGCN	**85.3 ± 1.2**	**90.6 ± 0.2**

	MUTAG (Graph)	**PCT**
PachySAN†	**92.6 ± 4.2**	60.0 ± 4.8
DGCNN†	85.8	58.6
GIN†	89.4 ± 4.7	64.6 ± 7.0
R-GCN	82.3 ± 9.2	67.8 ± 13.2
SynGCN	79.3 ± 10.3	69.4 ± 11.5
WGCN	78.9 ± 12.0	67.3 ± 12.0
CompGCN	89.0 ± 11.1	**71.6 ± 12.0**

Table 8.5: Performance comparison on node classification (**Left**) and graph classification (**Right**) tasks. $*$ and † indicate that results are directly taken from [165] and [207] respectively. Overall, we find that CompGCN either outperforms or performs comparably compared to the existing methods. Please refer to Section 8.4.2.4 for more details.

the increasing number of relations and thus can be utilized for larger graphs effectively.

8.4.2.4 Evaluation on Node and Graph Classification

In this section, we evaluate CompGCN on node and graph classification tasks on datasets as described in Section 8.4.1.1. The experimental results are presented in Table 8.5. For node classification task, we report accuracy on test split provided by [163], whereas for graph classification, following [210] and [207], we report the average and standard deviation of validation accuracies across the 10 folds cross-validation. Overall, we find that CompGCN outperforms all the baseline methods on node classification and gives a comparable performance on graph classification task. This demonstrates the effectiveness of incorporating relations using CompGCN over the existing GCN based models. On node classification, compared to the best performing baseline, we obtain an average improvement of 3% across both datasets while on graph classification, we obtain an improvement of 3% on PCT dataset.

8.5 Conclusion

In this chapter, we proposed CompGCN, a novel Graph Convolutional based framework for multi-relational graphs which leverages a variety of composition operators from Knowledge Graph embedding techniques to jointly embed nodes and relations in a graph. Our method generalizes several existing multi-relational GCN methods. Moreover, our method alleviates the problem of over-parameterization by sharing relation embeddings across layers and using basis decomposition. Through extensive experiments on knowledge graph link prediction, node classification, and graph classification tasks, we showed the effectiveness of CompGCN over existing GCN methods and demonstrated its scalability with increasing number of relations.

Chapter 9

Conclusion and Future Work

The first part of the thesis explored two different ways of addressing the sparsity problem in Knowledge Graphs (KG). We begin with alleviating it through canonicalization, which involves identifying and merging identical nodes in a given KG. For this, we proposed CESI (Canonicalization using Embeddings and Side Information), a novel method for canonicalizing Open KBs using learned embeddings and side information. CESI solves a joint objective to learn noun and relation phrase embeddings while utilizing relevant side information in a principled manner. These learned embeddings are then clustered together to obtain canonicalized noun and relation phrase clusters. The second approach is Relation Extraction which involves using unstructured text for extracting facts to densify KGs. We propose RESIDE, a novel neural network-based model which makes principled use of relevant side information, such as entity type and relation alias, from Knowledge Base, for improving distantly supervised relation extraction. RESIDE employs Graph Convolution Networks for encoding syntactic information of sentences and is robust to limited side information. Through experimental results, we demonstrated the effectiveness of all the proposed methods on several benchmark datasets.

In the second part of the thesis, we showed the effectiveness of utilizing recently proposed Graph Convolutional Networks (GCNs) for exploiting several graph structures in NLP. We demonstrated their effectiveness on two important problems: Document Timestamping and learning word embeddings. For the first problem, we proposed NeuralDater, a GCN based method for document dating which exploits syntactic and temporal structures in the document in a principled way. To the best of our knowledge, this is the first application of deep learning techniques for the problem of document dating. For the second problem, we proposed SynGCN, a graph convolution-based approach which utilizes syntactic context for learning word representations. SynGCN overcomes the problem of vocabulary explosion and outperforms state-of-the-art word embedding approaches on several intrinsic and extrinsic tasks. We also propose SemGCN, a framework for jointly incorporating diverse semantic information in pre-trained word embeddings. The combination of SynGCN and SemGCN gives the best overall performance.

Finally, in the third part of the thesis, we addressed two significant limitations of existing Graph Convolutional Network-based methods. First, we address the issue of noisy representation of hub nodes in GCNs because of neighborhood aggregation scheme which puts no constraint on the influence neighborhood of a node. For this, we present ConfGCN, confidence based Graph Convolutional Network, which estimates label scores along with their confidences jointly in a GCN based setting. In ConfGCN, the influence of one node on another during aggregation is determined using the estimated confidences and label scores, thus inducing anisotropic behavior to GCN. Apart from this also extend existing GCN models for multi-relational graphs, which are a more pervasive class of graphs for modeling data. We propose CompGCN, a novel Graph Convolutional based framework for multi-relational graphs which leverages a variety of composition operators from Knowledge Graph embedding techniques to embed nodes and relations in a graph jointly. Our method generalizes several existing multi-relational GCN methods. Moreover, our method alleviates the problem of over-parameterization by sharing relation embeddings across layers and using basis decomposition. Through extensive experiments on several tasks, we demonstrated the effectiveness of our proposed solutions.

Future Works: An exciting future direction for addressing sparsity in Knowledge Graphs is to utilize contextualized embedding methods such as ELMo [154] and BERT [50] instead of GloVe for obtaining the representation of noun and relation phrases in Open KG canonicalization. Contextualized embedding approaches have been shown to give superior performance than standard word2vec based embeddings for a variety of tasks. However, utilizing them for canonicalization has not been explored so far. Another future work includes extending our proposed model RESIDE to utilize more types of side information. KGs are a vast storehouse of facts, among which a lot of them can be utilized for further improving RE. For further enhancing performance on document timestamping problem, one can explore utilizing knowledge graphs which contain information about world events. Exploiting world knowledge for the task is also more close to how a human would approach the problem. In Chapter 6, we explored extending word2vec using Graph Convolutional Networks. However, recently, contextualized embedding methods have been shown to be much more effective. Thus, one can use similar ideas to extend models such as BERT using GCNs for utilizing syntactic information for learning better representation. Finally, existing GCN methods suffer from several other limitations, which have been highlighted by Xu et al. [206] in his work.

Bibliography

[1] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 37–45, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.290954. URL http://doi.acm.org/10.1145/290941.290954. 47

[2] Abdulrahman Almuhareb. Attributes in lexical acquisition. 2006. 67

[3] Mohammed Alsuhaibani, Danushka Bollegala, Takanori Maehara, and Ken-ichi Kawarabayashi. Jointly learning word embeddings using a corpus and a knowledge base. *PLOS ONE*, 13(3):1–26, 03 2018. doi: 10.1371/journal.pone.0193094. URL https://doi.org/10.1371/journal.pone.0193094. 60, 61, 64, 66

[4] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1034. URL https://www.aclweb.org/anthology/P15-1034. 36, 39, 40

[5] Ben Athiwaratkun and Andrew Gordon Wilson. On modeling hierarchical data via probabilistic order embeddings. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJCXZQbAZ. 76

[6] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, pages 722–735, Berlin, Heidelberg, 2007. Springer-Verlag.

ISBN 3-540-76297-3, 978-3-540-76297-3. URL http://dl.acm.org/citation.cfm?id=1785162.1785216. 15

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv e-prints*, abs/1409.0473, September 2014. URL https://arxiv.org/abs/1409.0473. 36

[8] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, ACL '98, pages 86–90, 1998. doi: 10.3115/980845.980860. URL https://doi.org/10.3115/980845.980860. 60

[9] Ivana Balažević, Carl Allen, and Timothy M Hospedales. Hypernetwork knowledge graph embeddings. In *International Conference on Artificial Neural Networks*, 2019. 91, 92

[10] Satanjeev Banerjee and Ted Pedersen. *An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet*, pages 136–145. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002. ISBN 978-3-540-45715-2. doi: 10.1007/3-540-45715-1_11. URL https://doi.org/10.1007/3-540-45715-1_11. 20

[11] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2670–2676, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1625275.1625705. 15, 18

[12] Marco Baroni and Alessandro Lenci. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721, December 2010. ISSN 0891-2017. doi: 10.1162/coli_a_00016. URL http://dx.doi.org/10.1162/coli_a_00016. 67

[13] Marco Baroni and Alessandro Lenci. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, pages 1–10, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-937284-16-9. URL http://dl.acm.org/citation.cfm?id=2140490.2140491. 67

[14] Marco Baroni, Stefan Evert, and Alessandro Lenci. Esslli 2008 workshop on distributional lexical semantics. Association for Logic, Language and Information, 2008. 67

[15] Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D17-1209. 1, 3, 7, 35, 49, 60, 61, 73

[16] Daniel Beck, Gholamreza Haffari, and Trevor Cohn. Graph-to-sequence learning using gated graph neural networks. In Iryna Gurevych and Yusuke Miyao, editors, *ACL 2018 - The 56th Annual Meeting of the Association for Computational Linguistics*, pages 273–283. Association for Computational Linguistics (ACL), 2018. ISBN 9781948087322. 85

[17] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7: 2399–2434, December 2006. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1248547.1248632. 73, 76, 80, 81

[18] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8): 1798–1828, Aug 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2013.50. 59

[19] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944966. 60

[20] Aleksandar Bojchevski and Stephan Gnnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=r1ZdKJ-0W. 76, 80

[21] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-102-6. doi: 10.1145/1376616.1376746. URL http://doi.acm.org/10.1145/1376616.1376746. 1, 15, 34

[22] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In

C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf. 2, 16, 17, 22, 24, 87, 88, 90, 91, 92, 93

[23] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1067. URL https://www.aclweb.org/anthology/D14-1067. 87

[24] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1067. URL http://aclweb.org/anthology/D14-1067. 1

[25] Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. In Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 165–180, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. 1

[26] Matthias Bröcheler, Lilyana Mihalkova, and Lise Getoor. Probabilistic similarity logic. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, UAI'10, pages 73–82, Arlington, Virginia, United States, 2010. AUAI Press. ISBN 978-0-9749039-6-5. URL http://dl.acm.org/citation.cfm?id=3023549.3023558. 17

[27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, July 2017. ISSN 1053-5888. doi: 10.1109/MSP.2017.2693418. 61, 76

[28] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013. URL http://arxiv.org/abs/1312.6203. 76, 87

[29] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014. 49

[30] Liwei Cai and William Yang Wang. KBGAN: Adversarial learning for knowledge graph embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1470–1480, 2018. URL https://www.aclweb.org/anthology/N18-1133. 91, 92

[31] Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. Clueweb09 data set, 2009. 24

[32] Nathanael Chambers. Labeling documents with timestamps: Learning from their time expressions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 98–106, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390524.2390539. 47, 49, 50, 52, 55, 56

[33] Nathanael Chambers and Dan Jurafsky. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 698–706, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1613715.1613803. 49

[34] Nathanael Chambers, Shan Wang, and Dan Jurafsky. Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 173–176, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557769.1557820. 49

[35] Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. Dense event ordering with a multi-pass architecture. *Transactions of the Association of Computational Linguistics*, 2:273–284, 2014. URL http://www.aclweb.org/anthology/Q14-1022. 48, 49, 53

[36] Angel X. Chang and Christopher Manning. Sutime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*. European Language Resources Association (ELRA), 2012. URL http://www.aclweb.org/anthology/L12-1122. 53

[37] Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rytstxWAW. 7

[38] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179. URL http://www.aclweb.org/anthology/D14-1179. 38

[39] Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the Sixth International Conference on Knowledge Capture*, K-CAP '11, pages 113–120, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0396-5. doi: 10.1145/1999676.1999697. URL http://doi.acm.org/10.1145/1999676.1999697. 2, 15

[40] Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-1078. 67

[41] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2078186. 59, 60

[42] Wisam Dakka, Luis Gravano, and Panagiotis G. Ipeirotis. Answering general time sensitive queries. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 1437–1438, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-991-3. doi: 10.1145/1458082.1458320. URL http://doi.acm.org/10.1145/1458082.1458320. 47

[43] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804. Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-1077. URL http://www.aclweb.org/anthology/P15-1077. 77

[44] Franciska M.G. de Jong, H. Rode, and Djoerd Hiemstra. *Temporal Language Models for*

the Disclosure of Historical Text, pages 161–168. KNAW, 9 2005. ISBN 90-6984-456-7. Imported from EWI/DB PMS [db-utwente:inpr:0000003683]. 3, 47, 49

[45] Daniel Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977. 23

[46] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *CoRR*, abs/1606.09375, 2016. URL http://arxiv.org/abs/1606.09375. 1, 3, 7, 35, 36, 60, 73, 76, 87

[47] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3844–3852, USA, 2016. Curran Associates Inc. ISBN 978-1-5108-3881-9. URL http://dl.acm.org/citation.cfm?id=3157382.3157527. 48, 49

[48] Claudio Delli Bovi, Luis Espinosa Anke, and Roberto Navigli. Knowledge base unification via sense embeddings and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 726–736, Lisbon, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/D15-1084. 18

[49] Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818, February 2018. URL https://arxiv.org/abs/1707.01476. 87, 88, 90, 91, 92

[50] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 97

[51] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 601–610, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623623. URL http://doi.acm.org/10.1145/2623330.2623623. 2, 22

[52] Ludovic Dos Santos, Benjamin Piwowarski, and Patrick Gallinari. Gaussian embeddings for collaborative filtering. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 1065–1068, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5022-8. doi: 10.1145/3077136.3080722. URL http://doi.acm.org/10.1145/3077136.3080722. 77

[53] Jennifer D'Souza and Vincent Ng. Classifying temporal relations with rich linguistic knowledge. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 918–927. Association for Computational Linguistics, 2013. URL http://www.aclweb.org/anthology/N13-1112. 49

[54] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP '11)*, Edinburgh, Scotland, UK, July 27-31 2011. 15, 20, 24, 26

[55] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard H. Hovy, and Noah A. Smith. Retrofitting word vectors to semantic lexicons. *CoRR*, abs/1411.4166, 2014. URL http://arxiv.org/abs/1411.4166. 60, 61, 62, 64, 65, 66

[56] Xiaocheng Feng, Jiang Guo, Bing Qin, Ting Liu, and Yongjie Liu. Effective deep memory networks for distant supervised relation extraction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4002–4008, 2017. doi: 10.24963/ijcai.2017/559. URL https://doi.org/10.24963/ijcai.2017/559. 42, 43

[57] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics, Association for Computational Linguistics. doi: 10.3115/1219840.1219885. URL https://doi.org/10.3115/1219840.1219885. 42

[58] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 406–414,

New York, NY, USA, 2001. ACM. ISBN 1-58113-348-0. doi: 10.1145/371920.372094. URL http://doi.acm.org/10.1145/371920.372094. 66

[59] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6533–6542, USA, 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4. URL http://dl.acm.org/citation.cfm?id=3295222.3295399. 7

[60] Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. Facc1: Freebase annotation of clueweb corpora, version 1. *Release date*, pages 06–26, 2013. 24

[61] Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 1679–1688, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2598-1. doi: 10.1145/2661829.2662073. URL http://doi.acm.org/10.1145/2661829.2662073. 16, 18, 19, 20, 21, 23, 24, 25, 27, 28

[62] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. Amie: Association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 413–422, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2035-1. doi: 10.1145/2488388.2488425. URL http://doi.acm.org/10.1145/2488388.2488425. 18, 21, 28

[63] W. M. Gentleman and G. Sande. Fast fourier transforms: For fun and profit. In *Proceedings of the November 7-10, 1966, Fall Joint Computer Conference*, AFIPS '66 (Fall), pages 563–578, New York, NY, USA, 1966. ACM. doi: 10.1145/1464291.1464352. URL http://doi.acm.org/10.1145/1464291.1464352. 10

[64] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1263–1272. JMLR.org, 2017. URL http://dl.acm.org/citation.cfm?id=3305381.3305512. 4, 87

[65] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of

Proceedings of Machine Learning Research, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL http://proceedings.mlr.press/v9/glorot10a.html. 80

[66] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 48

[67] A. Graves, A. r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, May 2013. doi: 10.1109/ICASSP.2013.6638947. 38

[68] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. 76

[69] Michael Gutmann and Aapo Hyvrinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL http://proceedings.mlr.press/v9/gutmann10a.html. 65

[70] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017. 85, 87

[71] Zhengqiu He, Wenliang Chen, Zhenghua Li, Meishan Zhang, Wei Zhang, and Min Zhang. See: Syntax-aware entity embedding for neural relation extraction, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16362. 35, 36, 38, 40, 42

[72] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with genuine similarity estimation. *Comput. Linguist.*, 41(4):665–695, December 2015. ISSN 0891-2017. doi: 10.1162/COLI_a_00237. URL http://dx.doi.org/10.1162/COLI_a_00237. 66

[73] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012. ISSN 1053-5888. doi: 10.1109/MSP.2012.2205597. 48

[74] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9 (8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `http://dx.doi.org/10.1162/neco.1997.9.8.1735`. 3, 51

[75] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics, 2011. 34, 36, 42

[76] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1243–1252. IEEE computer Society, 2017. 3, 73

[77] S. Jat, S. Khandelwal, and P. Talukdar. Improving Distantly Supervised Relation Extraction using Word and Entity Based Attention. *ArXiv e-prints*, April 2018. 35, 39, 41, 42

[78] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *AAAI*, 2017. 36

[79] Shihao Ji, Hyokun Yun, Pinar Yanardag, Shin Matsushima, and S. V. N. Vishwanathan. Wordrank: Learning word embeddings via robust ranking. *CoRR*, abs/1506.02761, 2015. URL `http://arxiv.org/abs/1506.02761`. 60

[80] Xiaotian Jiang, Quan Wang, and Bin Wang. Adaptive convolution for multi-relational learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019. URL `https://www.aclweb.org/anthology/N19-1103"`. 91, 92

[81] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5: 339–351, 2017. URL `http://aclweb.org/anthology/Q17-1024`. 59

[82] David A. Jurgens, Peter D. Turney, Saif M. Mohammad, and Keith J. Holyoak. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint*

Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, SemEval '12, pages 356–364, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2387636.2387693. 67

[83] Nattiya Kanhabua and Kjetil Nørvåg. Improving temporal language models for determining time of non-timestamped documents. In *International Conference on Theory and Practice of Digital Libraries*, pages 358–370. Springer, 2008. 47, 49

[84] Douwe Kiela, Felix Hill, and Stephen Clark. Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2044–2048. Association for Computational Linguistics, 2015. doi: 10.18653/v1/D15-1242. URL http://aclweb.org/anthology/D15-1242. 61

[85] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1181. URL http://www.aclweb.org/anthology/D14-1181. 55, 56

[86] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 12 2014. 55, 80

[87] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016. URL http://arxiv.org/abs/1609.02907. 1, 3, 7, 11, 60, 61, 73, 74, 76, 79, 80, 81, 85, 87, 90

[88] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. 36, 48, 49

[89] Ari Kobren, Nicholas Monath, Akshay Krishnamurthy, and Andrew McCallum. A hierarchical algorithm for extreme clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 255–264, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098079. URL http://doi.acm.org/10.1145/3097983.3098079. 23

[90] Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *Proceedings of the 2016 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1490–1500, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1175. URL https://www.aclweb.org/anthology/N16-1175. 59, 60, 61, 62, 66, 68

[91] Dimitrios Kotsakos, Theodoros Lappas, Dimitrios Kotzias, Dimitrios Gunopulos, Nattiya Kanhabua, and Kjetil Nørvåg. A burstiness-aware approach for document dating. In *Proceedings of the 37th International ACM SIGIR Conference on Research &#38; Development in Information Retrieval*, SIGIR '14, pages 1003–1006, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2257-7. doi: 10.1145/2600428.2609495. URL http://doi.acm.org/10.1145/2600428.2609495. 47, 49, 50, 55, 56

[92] Jayant Krishnamurthy and Tom M. Mitchell. Which noun phrases denote which concepts? In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 570–580, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. URL http://dl.acm.org/citation.cfm?id=2002472.2002545. 17

[93] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=2999134.2999257. 1, 48

[94] Erdal Kuzey, Vinay Setty, Jannik Strötgen, and Gerhard Weikum. As time goes by: Comprehensive tagging of textual phrases with temporal scopes. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 915–925, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-4143-1. doi: 10.1145/2872427.2883055. URL https://doi.org/10.1145/2872427.2883055. 58

[95] Theodoros Lappas, Benjamin Arai, Manolis Platakis, Dimitrios Kotsakos, and Dimitrios Gunopulos. On burstiness-aware search for document sequences. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 477–486, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9. doi: 10.1145/1557019.1557075. URL http://doi.acm.org/10.1145/1557019.1557075. 47, 49, 55

[96] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*, pages 319–, London, UK, UK, 1999. Springer-Verlag. ISBN 3-540-66722-9. URL `http://dl.acm.org/citation.cfm?id=646469.691875`. 3, 55

[97] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Comput. Linguist.*, 39(4):885–916, December 2013. ISSN 0891-2017. 49

[98] Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/N18-2108`. 67

[99] Jing Lei. Classification with confidence. *Biometrika*, 101(4):755–769, 2014. doi: 10.1093/biomet/asu038. URL `http://dx.doi.org/10.1093/biomet/asu038`. 76

[100] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308. Association for Computational Linguistics, 2014. doi: 10.3115/v1/P14-2050. URL `http://www.aclweb.org/anthology/P14-2050`. 59, 60, 61, 62, 66, 68

[101] Chen Li, Jianxin Li, Yangqiu Song, and Ziwei Lin. Training and evaluating improved dependency-based word embeddings. In *AAAI*, 2018. 59, 61

[102] Mingkun Li and Ishwar K. Sethi. Confidence-based active learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(8):1251–1261, August 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.156. URL `https://doi.org/10.1109/TPAMI.2006.156`. 76

[103] Xiaoyan Li and W. Bruce Croft. Time-based language models. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, pages 469–475, New York, NY, USA, 2003. ACM. ISBN 1-58113-723-0. doi: 10.1145/956863.956951. URL `http://doi.acm.org/10.1145/956863.956951`. 3, 47

[104] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015. 80, 81

[105] Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter W. Battaglia. Learning deep generative models of graphs. *ArXiv*, abs/1803.03324, 2018. 7

[106] Renjie Liao, Marc Brockschmidt, Daniel Tarlow, Alexander Gaunt, Raquel Urtasun, and Richard S. Zemel. Graph partition neural networks for semi-supervised classification, 2018. URL `https://openreview.net/forum?id=rk4Fz2e0b`. xv, 73, 76, 79, 80, 81, 82

[107] Dekang Lin and Patrick Pantel. Dirt @sbt@discovery of inference rules from text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 323–328, New York, NY, USA, 2001. ACM. ISBN 1-58113-391-X. doi: 10.1145/502512.502559. URL `http://doi.acm.org/10.1145/502512.502559`. 17

[108] Thomas Lin, Mausam, and Oren Etzioni. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 84–88, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2391200.2391216`. 17

[109] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133, 2016. 35, 36, 41, 42, 43, 44

[110] Xiao Ling and Daniel S. Weld. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 94–100. AAAI Press, 2012. URL `http://dl.acm.org/citation.cfm?id=2900728.2900742`. 36, 39, 40

[111] Tianyu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. A soft-label method for noise-tolerant distantly supervised relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1790–1795. Association for Computational Linguistics, 2017. URL `http://aclweb.org/anthology/D17-1189`. 44

[112] Yang Liu, Kang Liu, Liheng Xu, and Jun Zhao. Exploring fine-grained entity type constraints for distantly supervised relation extraction. In *COLING*, 2014. 36

[113] D. Llidó, R. Berlanga, and M. J. Aramburu. Extracting temporal references to assign document event-time periods*. In Heinrich C. Mayr, Jiri Lazansky, Gerald Quirchmayr,

and Pavel Vogel, editors, *Database and Expert Systems Applications*, pages 62–71, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44759-7. 3, 47

[114] Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. Semeval-2015 task 5: Qa tempeval-evaluating temporal information understanding with question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 792–800, 2015. 48

[115] Minh-Thang Luong, Richard Socher, and Christopher D. Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, Sofia, Bulgaria, 2013. 66

[116] Xuezhe Ma and Eduard H. Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *CoRR*, abs/1603.01354, 2016. 59

[117] Y. Ma, P. A. Crook, R. Sarikaya, and E. Fosler-Lussier. Knowledge graph inference for spoken dialog systems. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5346–5350, April 2015. doi: 10.1109/ICASSP.2015.7178992. 1, 2

[118] Inderjeet Mani and George Wilson. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 69–76, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075228. URL `https://doi.org/10.3115/1075218.1075228`. 3, 47

[119] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. ISBN 0521865719, 9780521865715. 25

[120] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-5010`. 38, 52, 62

[121] Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515. Association for Computational Linguistics, 2017. URL `http://aclweb.org/anthology/D17-1159`. 2, 3, 7, 12, 35, 36, 38, 49, 57, 60, 61, 73, 76, 80, 81, 85, 87, 90, 91

[122] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The penn treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, pages 114–119, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. ISBN 1-55860-357-3. doi: 10.3115/1075812.1075835. URL `https://doi.org/10.3115/1075812.1075835`. 67

[123] Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 523–534, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390948.2391009`. 2, 36, 39

[124] Mausam Mausam. Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 4074–4077. AAAI Press, 2016. ISBN 978-1-57735-770-4. URL `http://dl.acm.org/citation.cfm?id=3061053.3061220`. 2, 15

[125] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA, 2013. Curran Associates Inc. URL `http://dl.acm.org/citation.cfm?id=2999792.2999959`. 40, 59, 60

[126] Tomas Mikolov, Scott Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*. Association for Computational Linguistics, May 2013. URL `https://www.microsoft.com/en-us/research/publication/linguistic-regularities-in-continuous-space-word-representations/`. 62, 63, 65, 66, 67

[127] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL `http://doi.acm.org/10.1145/219717.219748`. 1, 20, 60

[128] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009. 1, 3, 34, 35, 36, 38, 40, 42

[129] Paramita Mirza and Sara Tonelli. Classifying temporal relations with simple features. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 308–317. Association for Computational Linguistics, 2014. doi: 10.3115/v1/E14-1033. URL http://www.aclweb.org/anthology/E14-1033. 49

[130] Paramita Mirza and Sara Tonelli. Catena: Causal and temporal relation extraction from natural language texts. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 64–75. The COLING 2016 Organizing Committee, 2016. URL http://www.aclweb.org/anthology/C16-1007. 48, 49, 51, 53

[131] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*, 2015. 15, 17

[132] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. Never-ending learning. *Commun. ACM*, 61(5):103–115, April 2018. ISSN 0001-0782. doi: 10.1145/3191513. URL http://doi.acm.org/10.1145/3191513. 1

[133] Federico Monti, Oleksandr Shchur, Aleksandar Bojchevski, Or Litany, Stephan Günnemann, and Michael M. Bronstein. Dual-primal graph convolutional networks. *CoRR*, abs/1806.00770, 2018. URL http://arxiv.org/abs/1806.00770. 85

[134] Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *Proceedings of the Tenth International Workshop on Artificial Intelligence*

and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/208.pdf. 65

[135] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148. Association for Computational Linguistics, 2016. doi: 10.18653/v1/N16-1018. URL http://www.aclweb.org/anthology/N16-1018. 60, 61, 62, 64, 66

[136] Tushar Nagarajan, Sharmistha Jat, and Partha Talukdar. Candis: Coupled & attention-driven neural distant supervision. *CoRR*, abs/1710.09942, 2017. 36

[137] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. Patty: A taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1135–1145, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2390948.2391076. 17

[138] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 327–333. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-2053. URL http://aclweb.org/anthology/N18-2053. 91, 92

[139] Thien Nguyen and Ralph Grishman. Graph convolutional networks with argument-aware pooling for event detection, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16329. 2, 12, 36, 38, 49

[140] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, Jan 2016. ISSN 0018-9219. doi: 10.1109/JPROC.2015.2483592. 85, 87, 88

[141] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, USA,

2011. Omnipress. ISBN 978-1-4503-0619-5. URL http://dl.acm.org/citation.cfm?id=3104482.3104584. 17

[142] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 1955–1961. AAAI Press, 2016. URL http://dl.acm.org/citation.cfm?id=3016100.3016172. 16, 17, 21, 22, 27, 87, 88, 93

[143] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 2014–2023. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045603. 91

[144] Madhav Nimishakavi, Uday Singh Saini, and Partha Talukdar. Relation schema induction using tensor factorization with side information. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 414–423. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1040. URL http://www.aclweb.org/anthology/D16-1040. 17

[145] MA Olson, K Bostic, MI Seltzer, and DB Berkeley. Usenix annual technical conference, freenix track, 1999. 3, 47

[146] Matan Orbach and Koby Crammer. Graph-based transduction with confidence. In *ECML/PKDD*, 2012. 74, 76, 77, 78, 80

[147] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition ldc2011t07. dvd. *Philadelphia: Linguistic Data Consortium*, 2011. 54

[148] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition, linguistic data consortium. Technical report, Technical report, Technical Report. Linguistic Data Consortium, Philadelphia, 2011. 27

[149] Heiko Paulheim and Johannes Fümkranz. Unsupervised generation of data mining features from linked open data. In *Proceedings of the 2Nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 31:1–31:12, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0915-8. doi: 10.1145/2254129.2254168. URL http://doi.acm.org/10.1145/2254129.2254168. 91

[150] Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 425–430, 2015. URL http://aclweb.org/anthology/P/P15/P15-2070.pdf. 20, 27, 40

[151] Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *In Proceedings of the Association for Computational Linguistics (ACL-2015)*, pages 425–430. Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-2070. URL http://www.aclweb.org/anthology/P15-2070. 60

[152] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL http://www.aclweb.org/anthology/D14-1162. 23, 27, 38, 40, 51, 59, 60, 66

[153] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 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, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623732. URL http://doi.acm.org/10.1145/2623330.2623732. 76, 80, 81

[154] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL*, 2018. 60, 67, 68, 71, 97

[155] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, CoNLL '12, pages 1–40, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=2391181.2391183. 67

[156] Jay Pujara, Hui Miao, Lise Getoor, and William Cohen. Knowledge graph identification. In *Proceedings of the 12th International Semantic Web Conference - Part I*, ISWC '13,

pages 542–557, New York, NY, USA, 2013. Springer-Verlag New York, Inc. ISBN 978-3-642-41334-6. doi: 10.1007/978-3-642-41335-3_34. URL http://dx.doi.org/10.1007/978-3-642-41335-3_34. 17

[157] James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK., 2003. 48

[158] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1264. URL http://aclweb.org/anthology/D16-1264. 67

[159] Bharath Ramsundar, Peter Eastman, Patrick Walters, Vijay Pande, Karl Leswing, and Zhenqin Wu. *Deep Learning for the Life Sciences*. O'Reilly Media, 2019. https://www.amazon.com/Deep-Learning-Life-Sciences-Microscopy/dp/1492039837. 7

[160] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1375–1384, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. URL http://dl.acm.org/citation.cfm?id=2002472.2002642. 17

[161] Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer, 2010. 34, 36, 42

[162] Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*, pages 498–514. Springer, 2016. 91

[163] Petar Ristoski, Gerben Klaas Dirk de Vries, and Heiko Paulheim. A collection of benchmark datasets for systematic evaluations of machine learning on the semantic web. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web – ISWC 2016*, pages 186–194, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46547-0. 95

[164] Swarnadeep Saha, Harinder Pal, and Mausam. Bootstrapping for numerical open ie. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 317–323. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-2050. URL http://www.aclweb.org/anthology/P17-2050. 15

[165] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. *arXiv preprint arXiv:1703.06103*, 2017. xvi, 7, 85, 87, 89, 90, 91, 92, 93, 95

[166] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008. 79

[167] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion, 2019. 87, 90, 91, 92

[168] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.*, 12: 2539–2561, November 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2078187. 91

[169] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013. doi: 10.1109/MSP.2012.2235192. 7

[170] A. P. Singh, R. Nath, and S. Kumar. A survey: Speech recognition approaches and techniques. In *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pages 1–4, Nov 2018. doi: 10.1109/UPCON.2018.8596954. 1

[171] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing With Compositional Vector Grammars. In *ACL*. 2013. 59

[172] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In C. J. C.

Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 926–934. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5028-reasoning-with-neural-tensor-networks-for-knowledge-base-completion.pdf. 87, 88

[173] Gaurav Sood. Parsed dmoz data, 2016. URL http://dx.doi.org/10.7910/DVN/OMV93V. 32

[174] Daniel A. Spielman. Spectral graph theory and its applications. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 29–38, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3010-9. doi: 10.1109/FOCS.2007.66. URL http://dx.doi.org/10.1109/FOCS.2007.66. 7

[175] Valentin I. Spitkovsky and Angel X. Chang. A cross-lingual dictionary for english wikipedia concepts. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7. 20, 27

[176] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/D18-1548. 59

[177] Amarnag Subramanya and Partha Pratim Talukdar. Graph-based semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(4):1–125, 2014. 73

[178] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *16th International Conference on the World Wide Web*, pages 697–706, 2007. 1, 15

[179] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkgEQnRqYQ. 91, 92

[180] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 455–465. Association for Computational Linguistics, 2012. 21, 34, 36, 42

[181] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2969033.2969173`. 38, 51, 52

[182] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566. Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-1150. URL `http://aclweb.org/anthology/P15-1150`. 62

[183] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005. ISBN 0321321367. 18

[184] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *WWW*. ACM, 2015. 76

[185] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1119176.1119195. URL `https://doi.org/10.3115/1119176.1119195`. 67

[186] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015. 90

[187] Salvatore Trani, Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. Dexter 2.0: An open source tool for semantically enriching data. In *Proceedings of the 2014 International Conference on Posters &#38; Demonstrations Track - Volume*

1272, ISWC-PD'14, pages 417–420, Aachen, Germany, Germany, 2014. CEUR-WS.org. URL http://dl.acm.org/citation.cfm?id=2878453.2878558. 17

[188] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 2071–2080. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045609. 91, 92

[189] Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 1–9, 2013. 48

[190] L.J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. 2008. 32

[191] Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. Dating documents using graph convolution networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1605–1615. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-1149. 36, 38, 60, 73

[192] Shikhar Vashishth, Prince Jain, and Partha Talukdar. CESI: Canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 1317–1327, Republic and Canton of Geneva, Switzerland, 2018. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-5639-8. doi: 10.1145/3178876.3186030. URL https://doi.org/10.1145/3178876.3186030. 39

[193] Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. Reside: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/D18-1157. 61, 73

[194] Shikhar Vashishth, Prateek Yadav, Manik Bhandari, and Partha Talukdar. Confidence-based graph convolutional networks for semi-supervised learning. In Kamalika Chaudhuri

and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1792–1801. PMLR, 16–18 Apr 2019. URL http://proceedings.mlr.press/v89/vashishth19a.html. 61

[195] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. accepted as poster. xv, 73, 75, 76, 80, 81, 82, 85, 87

[196] Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th international workshop on semantic evaluations*, pages 75–80. Association for Computational Linguistics, 2007. 48

[197] Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 57–62. Association for Computational Linguistics, 2010. 48

[198] Luke Vilnis and Andrew McCallum. Word representations via gaussian embedding. *CoRR*, abs/1412.6623, 2014. URL http://arxiv.org/abs/1412.6623. 76

[199] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, September 2014. ISSN 0001-0782. doi: 10.1145/2629489. URL http://doi.acm.org/10.1145/2629489. 34

[200] Xiaojun Wan. Timedtextrank: Adding the temporal dimension to multi-document summarization. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 867–868, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-597-7. doi: 10.1145/1277741.1277949. URL http://doi.acm.org/10.1145/1277741.1277949. 3, 47

[201] Q. Wang, Z. Mao, B. Wang, and L. Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, Dec 2017. ISSN 1041-4347. doi: 10.1109/TKDE.2017.2754499. 85, 87

[202] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press, 2014. URL http://dl.acm.org/citation.cfm?id=2893873.2894046. 87

[203] Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1168–1175, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390303. URL http://doi.acm.org/10.1145/1390156.1390303. 73, 76, 80, 81

[204] William E Winkler. The state of record linkage and current research problems. In *Statistical Research Division, US Census Bureau*. Citeseer, 1999. 27

[205] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. Rc-net: A general framework for incorporating knowledge into word representations. November 2014. URL https://www.microsoft.com/en-us/research/publication/rc-net-a-general-framework-for-incorporating-knowledge-into-word-representations/ 61

[206] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5453–5462, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/xu18c.html. 2, 74, 84, 97

[207] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km. xvi, 2, 87, 91, 95

[208] Prateek Yadav, Madhav Nimishakavi, Naganand Yadati, Shikhar Vashishth, Arun Rajkumar, and Partha Talukdar. Lovasz convolutional networks. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1978–1987. PMLR, 16–18 Apr 2019. URL http://proceedings.mlr.press/v89/yadav19a.html. 61, 76

[209] Yadollah Yaghoobzadeh, Heike Adel, and Hinrich Schütze. Noise mitigation for neural entity typing and relation extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1183–1194. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/E17-1111. 36, 40

[210] Pinar Yanardag and S.V.N. Vishwanathan. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1365–1374, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3664-2. doi: 10.1145/2783258.2783417. URL http://doi.acm.org/10.1145/2783258.2783417. 91, 95

[211] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *CoRR*, abs/1412.6575, 2014. URL http://arxiv.org/abs/1412.6575. 87, 88, 91, 92, 93

[212] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 40–48. JMLR.org, 2016. URL http://dl.acm.org/citation.cfm?id=3045390.3045396. 76, 80, 81

[213] Liang Yao, Chengsheng Mao, and Yuan Luo. Graph Convolutional Networks for Text Classification. *ArXiv e-prints*, art. arXiv:1809.05679, September 2018. 60, 61

[214] Alexander Yates and Oren Etzioni. Unsupervised methods for determining object and relation synonyms on the web. *J. Artif. Int. Res.*, 34(1):255–296, March 2009. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=1622716.1622724. 18

[215] Rui Ye, Xin Li, Yujie Fang, Hongyu Zang, and Mingzhong Wang. A vectorized relational graph convolutional network for multi-relational network alignment. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4135–4141. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/574. URL https://doi.org/10.24963/ijcai.2019/574. 87, 91, 92

[216] Li Yi, Hao Su, Xingwen Guo, and Leonidas Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. *arXiv preprint arXiv:1612.00606*, 2016. 3, 73

[217] Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 405–413. Association for Computational Linguistics, 2009. URL http://www.aclweb.org/anthology/P09-1046. 49

[218] Mo Yu and Mark Dredze. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–550. Association for Computational Linguistics, 2014. doi: 10.3115/v1/P14-2089. URL http://www.aclweb.org/anthology/P14-2089. 60

[219] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, 2014. 35, 36, 38

[220] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, 2015. 35, 36, 42

[221] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 353–362, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939673. URL http://doi.acm.org/10.1145/2939672.2939673. 1

[222] Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *AAAI*, pages 4438–4445, 2018. 91

[223] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/D18-1244. 3, 62, 73

[224] Z. Zhao, P. Zheng, S. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, 30(11):3212–3232, Nov 2019. doi: 10.1109/TNNLS.2018.2876865. 1

[225] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International*

Conference on International Conference on Machine Learning, ICML'03, pages 912–919. AAAI Press, 2003. ISBN 1-57735-189-4. URL `http://dl.acm.org/citation.cfm?id=3041838.3041953`. 73, 76, 80, 81