UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft
Faculty for Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Research Group DICE

Seminar Report

Submitted to the DICE Research Group
in Partial Fullfilment of the Completion of

Seminar on

A Hybrid Graph Model for Distant Supervision Relation Extraction : A Report

by
Varun Maitreya Eranki

Thesis Supervisor:
Diego Moussalem

Paderborn, February 28, 2020

Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen worden ist. Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift

Abstract.
Write at the last

Contents

Chapter 1

Introduction

From plain text, the task of extracting semantic relations between two or more entities, is known as Relation Extraction (RE). These relations can exist in different types. For example, *"Germany is in Europe"* states a *"is in"* relationship between *"Germany"* and *"Europe"*. With this information, a triple can be formed, <*Germany, is in, Europe*>. Efficient RE is useful for applications like Knowledge Graph (KG) completion and question answering, which are in turn responsible for dependent applications.

Traditionally, supervised RE techniques produce elevated performance for RE[**?**]. They solely rely on labeled data that is manually annotated. Manual annotation is time consuming and need an army of annotators. It is generally annotated into entities and relationships(entities: *"Germany"*, *"Europe"* relationship: *"is in"*). This limitations strongly suggest need for semi-supervised or unsupervised RE techniques that are reliable enough and can mimic manual annotation.

Distant Supervision (DS) aims at solving this limitation by automatic production of labeled data by aligning KGs and plain text. In this process, it makes an assumption that, if there exists a relationship between two entities (*e1, e2*) in Knowledge Base (KB), then all the sentences that consist *e1, e2* express that relationship in some way[ZLCZ15]. In the Fig.1.1, if there exists, a triplet <*John Doe, FounderOf, John Doe Inc.*> then, all the sentences in the plain text that contain *John Doe* and *John Doe Inc.* are considered as the training instances of *FounderOf* relation. From the Table1.1, if the following sentences are considered, DS cannot have better performance as it categorizes relations like, *Recalled*, *ResignedFrom* into *CreatorOf*, *FounderOf* relations respectively. This introduces noise problem. It can be countered using Deep Neural Network (DNN) models, which try to provide a significant improvement, but fail in making predictions due to lack of sufficient background information associated with entities and relationships. For example, from Fig.1.1, there are two sentences generated using DS, they both use *"create"* relation but corresponding relations do not match. They in fact, represent *FounderOf* and *AuthorOf* relations respectively.

DNNs create bias at each stage and especially with long-tail relations, background information tend to be unusable for making predictions. They are mostly constructed for customized models to join knowledge that is limited to incorporate heterogeneous background information in parallel. Some of the methods did not handle the side effect caused due to introduced noise.

Graph-based model for DSRE was proposed by Duan et al.[DGLQ19] to solve the problems by DSRE. It fuses heterogeneous background information, as well, reduces side effects due to brought in noisy data. Typical process for neural networks start with conversion of different types

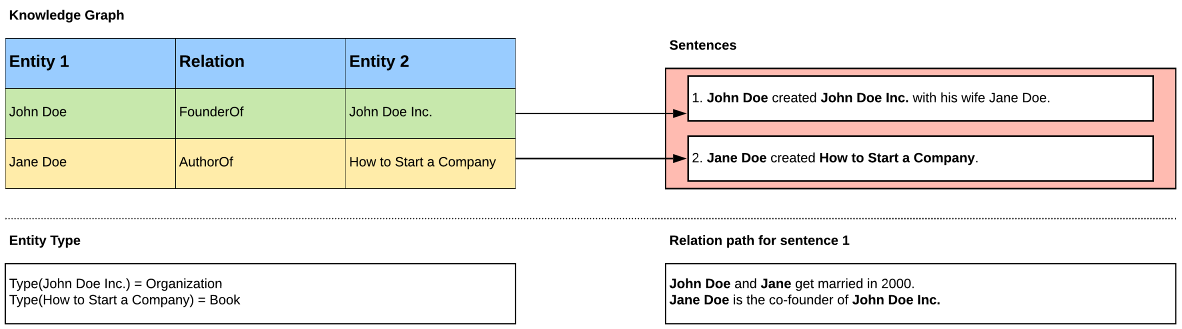


Figure 1.1: DS Example 1

Entity 1	Relation	Entity 2
John Doe	FounderOf	John Doe Inc.
John Doe	CreatorOf	Doe Glasses.
John Doe	Recalled	Doe Glasses.
John Doe	ResignedFrom	John Doe Inc.

Table 1.1: DS Example 2

data into vectors using various types of encoders and store each information in a graph node. Related nodes of graph are combined using a Graph Convolution Network(GCN). This model eases the incorporation of missing data and incorporate it with other graph nodes. Attention mechanism for the graph reduces the introduced noise problem by assigning higher weight for known true information.

Chapter 2

Concepts

2.1 Background Information for CNN and GCN

Sentence Bag: Sentence Bag(SB) is similar to Bag-of-Words(BoW) representation, with preserving the order of word occurrences. Many Deep Learning algorithms use BoW, but the word order is never stored, as a result, prediction of next word occurrence is hindered. This model generated a data set DS represented as $D = \{S_{(x_i,y_i)}|, i = 1, 2, ...\}$ and a sentence bag $S_{(x_i,y_i)}$ is a set of sentences with both entities x_i and y_i. Yet, it does not store the relation of x_i and y_i, and only the order in which entities occurred in a sentence.

Knowledge Graph: Knowledge Graph (KG) consists of triples $<x_i, r_i, y_i>$ where r_i is the relationship of entities x_i and y_i. This form of representation for KG represents a way to learn about vector embedding of both entities along with relation in a low-dimensional space.

Entity Type: An entity type T_{e_i} for any entity e, helps distinguish e from other entities. Fine-grained entity types suggested by Mintz et al.[MBSJ09] increase the chances for prediction of relation between two entities. For example, as shown in Fig.[**?**], *"How to Start a Company"* is *"book"*. With the T_{e_i} knowledge, prediction of relation between *Jane Doe* and *"How to Start a Company"* from *FounderOf* and *AuthorOf*.

Relation Path: Relation path p in DSRE is defined as a path between a set of entities ($p = x, e_1, ...e_{ln}, y$) whose relationship flows between entities x and y via ln entities. More the number of entities, larger the path. This is of a concern, when used in a Deep Neural Networks, which will be discussed later. Path p is shown in the figure[**?**]

2.2 Encoder

An encoder is a network (CNN, FC, RNN, etc.) that takes the input, and gives a feature map/vector/tensor as output. The feature vector contains the information and features, that represents the input. An encoder is often coupled with a decoder that takes feature vector as input and tries to produce output that is closest to the original input before encoding.

Encoder as a common usage pattern of RNN is utilized to explain encoder in brief. Specifically, an unrolled RNN is considered apt for the use-case.

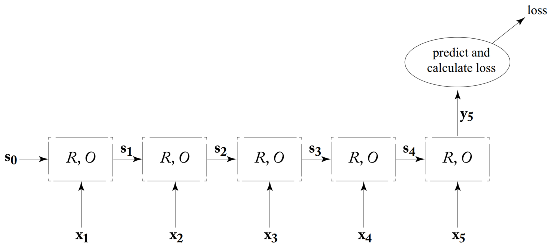


Figure 2.1: RNN Encoder

2.3 LSTM Cell: Long-Short-term Memory Cell

Long Short Term Memory networks(LSTMs)[HS97] are a type of RNN, enable to learn long-term dependencies. A simple LSTM cell consists of four gates, namely, Forget gate f_t, Input gate i_t, Gate gate $\widetilde{C_t}$ and Output gate o_t. $h_{(t-1)}$ is the output from previous state, x_t is the input, h_t is the output of present state, σ represents a sigmoid function, C_t and C_{t-1} represent cell state. Equations eq. (2.1) represent the gate operations where b and W are model parameters.

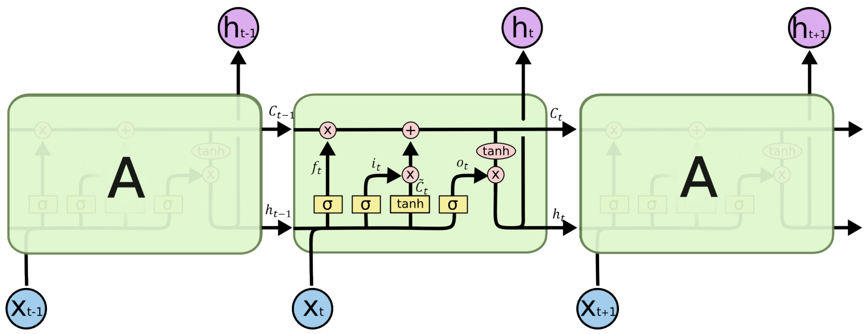


Figure 2.2: Simplified LSTM Example [Ola]

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2.1a}$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2.1b}$$
$$\widetilde{C_t} = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{2.1c}$$
$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{2.1d}$$
$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{2.1e}$$
$$h_t = o_t * \tanh\left(C_t\right) \tag{2.1f}$$

LSTMs are embedded into other networks in common to counter vanishing gradients problem and help learning long-term dependencies. This is done by enabling or disabling the gates depending on a model's use-case(target network). A more commonly used variant of LSTMs is Gate Recurrent Unit (GRU). Greff et. al.,[GSK+16] discuss LSTMs in detail about the popular variants.

2.4 GCN : Graph Convolution Network

The extention of deep neural networks to deal with arbitrary graph-structured data are known as Graph Neural Networks (GNNs)[GMS05, SGT+08]. In recent years, convolutional operations on graph-structured data are generalized into graph convolutional deep neural networks. Graph Convolutional Networks (GCNs) for short[KW16]. They are categorically divided into two main types, spectral domain and non-spectral domain. Spectral approaches work on the basis of spectral representation of graphs. Kipf et al.[KW16] proposed a spectral approach based on work done by Joan Bruna et al.[BZSL13] Michäel et al.[DBV16], which designs a GCN with a localized first-order approximation of spectral graph convolutions. Non-spectral approaches, perform convolutions directly on the graph.

Commonly, GCNs have two or more hidden layers, where information in the graph is embedded into as eigen vectors by learning some non-linear function. Convolution is the process of applying some filter on the graph data so as to reduce the size of feature matrix into a smaller vector representation. Feature matrix(adjacency matrix) of a graph as shown in Fig. has many empty cells. It is easier to compute on small graphs, but memory overhead increases as real world data-sets are very large. Matrix multiplication with an identity matrix, will reduce the size of adjacency matrix to a vector meanwhile preserving the data. In a GCN, Input will be the first layer and output will be the last layer, in between, there can be many hidden convolution layers with individual non-linear functions. A very simple form of layer-wise propagation rule would be of the form[KW16], where f is approximation , $\sigma(\cdot)$ is a non linear activation function, $H^{(l)}$ is the graph-level output and $W^{(l)}$ is the weight matrix for l-th neural network layer. L is the number of layers and $l \in L$. $H^{(0)} = X$ is the input $H^{(L)} = Z$ is the output

$$f\left(H^{(l)}, A\right) = \sigma\left(AH^{(l)}W^{(l)}\right) \tag{2.2a}$$

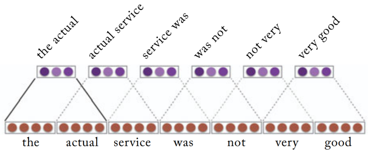


Figure 2.3: convolution Example

Chapter 3

Architecture

Architecture of Hybrid Graph Model is shown in Fig.3.1. This can be broadly divided into three segments. First segment consists of information encoders with individual encoders for various levels of data. Second or Middle segment consists of hybrid KG. This graph is constructed, by utilizing vector representations generated by encoders in previous step, and embedding them as, individual piece of information in a node that is relevant. In the Final segment, this hybrid graph is utilized by GCN with attention to extract features of the hybrid KG and final output is a probability distribution of the relations. The author, S. Duan et al.[DGLQ19], proposes to predict the relation between any entity pair $S_{(x_i,y_i)}$ and learn the probability distribution $P(r_i|x_i, y_i; \theta)$ over all relations $r_i \in \mathbb{R}$, where θ denotes the parameters of the model.

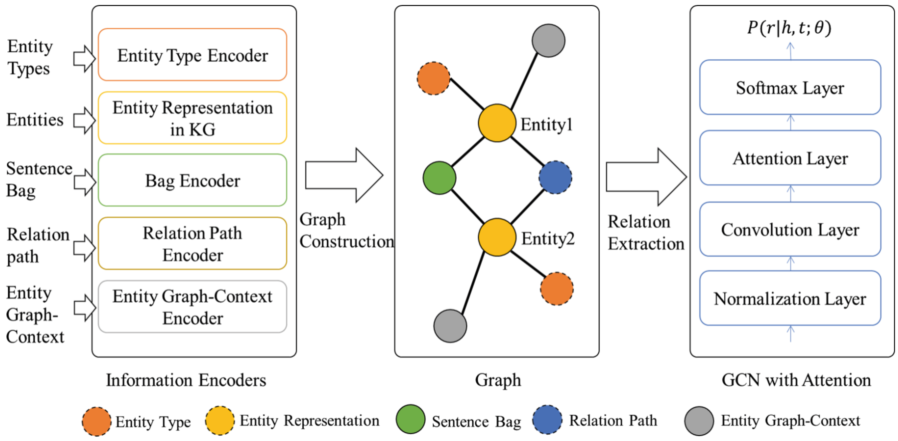
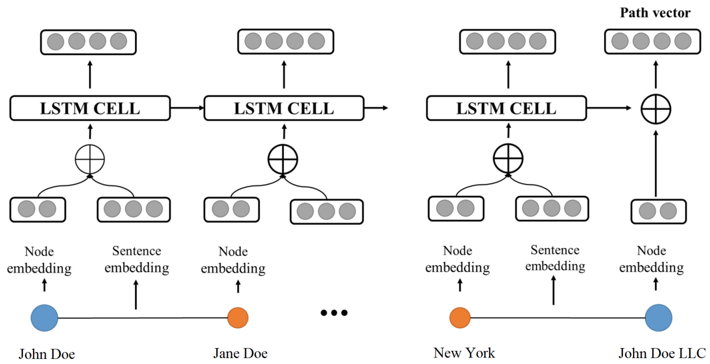


Figure 3.1: The Architecture

Before the process, from a given DS generated data set $D = \{S_{(x_i,y_i)}|, i = 1, 2, ...\}$, the background information from KG for every entity pair (x_i, y_i) is extracted and stored in a different data set $\mathbb{I} = \{I_{(x_1,y_1)}, I_{(x_2,y_2)}, ...\}$. Label of each instance corresponds to the label of $S_{(x_i,y_i)}$ during the extraction.


Path vector
LSTM CELL
LSTM CELL
LSTM CELL
Node embedding
Sentence embedding
Node embedding
Node embedding
Sentence embedding
Node embedding
John Doe
Jane Doe
New York
John Doe LLC

Chapter 4

Discussion

Convolutional neural networks serve as a powerful tool to solve high dimensional problems. CNNs give good results when, used with euclidean data (images, videos, sounds) that are compositional. Compositional features can be extracted and fed to a classifier, etc. These can be represented using euclidean domain that are regular spatial structures (for Eg. distance of adjacent pixels in an image are the same). To learn from non-euclidean data (real world data) like social networks web, Knowledge Graphs, etc. which are in the form of graphs, CNNs are adapted into Graph Convolutional Neural Networks (GCNs). They are similar to CNNs, but, instead of aggregating individual weights (distance of a node to adjacent nodes), a single shared weight is used by approximation. This solves the limitations of CNNs when nodes have non uniform neighbors. Most DSRE approaches are aimed towards semi-supervised models[SCM18]. The author Duan et. al.[DGLQ19], aims to show that GCNs are useful for unsupervised DSRE. Limitations of semi-supervised model proposed by Kipf et. al.[KW16], have been overcome with this HG model.

Proposed HG model is based on models proposed by Daojian Zeng et. al.,[ZLL+14] and Wenyuan Zeng et. al.,[ZLLS16]. In a novel way, incorporating relation path information in neural RE show a baseline results of efficient DSRE. Common limitation in dealing with non-euclidean data is noise problem. There are standard methods to solve noise problem for euclidean data[ZSK12], but not for the later case. Author Duan et. al.,[DGLQ19] uses an attention mechanism by selecting more relevant features to reduce the affect of noise using weighted sum operation over all features. Higher weight is assigned to important features, and can be altered to select or deselect those features that have high noise. Noise is a vague concept. To understand noise, let us consider two features, entity types and relations. For a certain use-case, that has more information on relations, but not much information about entity types. Model is said to have more noise, if entity types feature is given more weight. Prediction based on such training data would be less efficient. This feature of GCNs is used in HG model, which yields higher precision compared previous approaches. Compared to previous approaches, feature information is encoded using various feature encoders and the resultant eigen vectors(simply vectors) are embedded into hybrid graph. This embedding also embeds noisy data during encoding phase, that gets carried on to next step. The only way of controlling the affects of noise is using weight mechanism in attention layer for a hidden layer of GCN. This improves the precision but does not completely eliminate the noise problem which is the suggested future work.

One major achievement in HG model is, fusing heterogeneous information from the feature encoders, into a single hybrid graph, as they have different vector embedding. Where each vector represents a node in the graph. An adjacency matrix is used to explain the correlation between nodes. For each instance, and each vector embeddings, this varying structure is converted into a fixed structure using adjacent matrix. Then, high-level features are extracted by using GCN. The training phase for huge corpus takes a very large time. Decent sized dataset was used by aligning Wikidata relations with New York Times Corpus (NYT) for PCCNs[ZLLS16]. Wikidata has more than 80 million triple facts and 20 million entities which is a large sample size to do training. The experimental setup used for PCCNs is used as is, with small changes in parameters like learning rate for SGD, word embedding size, etc. This will not affect the overall results as main goal was comparison. Core advantage of deep learning techniques is, less time for testing phase. Other machine learning techniques which try to learn based on existing data take less training time and more testing time. Author Duan et. al.,[DGLQ19] discusses about run-time in less detail as it is prevalent that run-time depends on configuration of machine used, so not applicable in this case.

Author Duan et. al.,[DGLQ19] discusses about examples of testing dataset that do not occur. Using long-tail relations, HG model is able to out perform other models and find some score where other models fail. This proves author's assumption that, embedding additional information apart from a specific feature will yield better results for unseen data.

Chapter 5

Results and Comparisons

Chapter 6

Conclusion and Future work

This model can be considered as a semi-supervised learning model[LHW18].

this is the citing of neural networks[BDH96]

To Summarize succinctly, authors *S. Duan* et al., in the work *"A Hybrid Graph Model for Distant Supervision Relation Extraction"* propose a different approach for DSRE

Bibliography

[BDH96] Hagan Demuth Beale, Howard B Demuth, and MT Hagan. Neural network design. *Pws, Boston*, 1996.

[BZSL13] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[DBV16] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[DGLQ19] Shangfu Duan, Huan Gao, Bing Liu, and Guilin Qi. A hybrid graph model for distant supervision relation extraction. In *European Semantic Web Conference*, pages 36–51. Springer, 2019.

[GMS05] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[GSK+16] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[KW16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[LHW18] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[MBSJ09] 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.

[Ola] Christopher Olah. Simple LSTM. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

[SCM18] Alisa Smirnova and Philippe Cudré-Mauroux. Relation extraction using distant supervision: A survey. *ACM Computing Surveys (CSUR)*, 51(5):1–35, 2018.

[SGT+08] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[ZLCZ15] 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.

[ZLL+14] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. Relation classification via convolutional deep neural network. 2014.

[ZLLS16] Wenyuan Zeng, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Incorporating relation paths in neural relation extraction. *arXiv preprint arXiv:1609.07479*, 2016.

[ZSK12] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012.