Investigating the Impact of Different Graph Representations for Relation Extraction with Graph Neural Networks

[1]Moritz Blum, Gennaro Nolano, [1,2]Basil Ell, [1]Philipp Cimiano
[1]Bielefeld University, [2]University of Oslo
Germany, Norway
{mblum, bell, cimiano}@techfak.uni-bielefeld.de, nolanogenn@gmail.com

Abstract

Graph Neural Networks (GNNs) have been applied successfully to various NLP tasks, particularly Relation Extraction (RE). Even though most of these approaches rely on the syntactic dependency tree of a sentence to derive a graph representation, the impact of this choice compared to other possible graph representations has not been evaluated. We examine the effect of representing text though a graph of different graph representations for GNNs that are applied to RE, considering, e. g., a fully connected graph of tokens, of semantic role structures, and combinations thereof. We further examine the impact of background knowledge injection from Knowledge Graphs (KGs) into the graph representation to achieve enhanced graph representations. Our results show that combining multiple graph representations can improve the model's predictions. Moreover, the integration of background knowledge positively impacts scores, as enhancing the text graphs with *Wikidata* features or *WordNet* features can lead to an improvement of close to 0.1 in F_1.

Keywords: Relation Extraction, Graph Neural Networks, Background Knowledge

1. Introduction

The task of Relation Extraction (RE) consists of predicting the relation between two entities mentioned in a text. It represents an essential subtask for Information Extraction from text, and the result is used in several downstream tasks such as Question Answering (Yu et al., 2017; Xu et al., 2016) or Knowledge Base Population (Nguyen et al., 2018). Recently, approaches based on LSTMs (Hochreiter and Schmidhuber, 1997) and Transformers such as *BERT* (Devlin et al., 2019) have achieved state-of-the-art performance on RE by exploiting contextual information contained in the text around the entities (Wang and Yang, 2020; Baldini Soares et al., 2019; Wu and He, 2019).

A separate line of works makes use of Graph Neural Networks (GNNs), using neural network-based techniques to process graph-structured inputs. GNNs have been applied to RE, typically relying on the syntactic dependency tree of a sentence as graph representation. It has been argued that relying on a syntactic dependency tree i) facilitates dealing with long-distance phenomena (Tian et al., 2021; Miwa and Bansal, 2016), and ii) increases the robustness and generalizability of models (Xu et al., 2015; Marcheggiani and Titov, 2017).

So far, most GNN approaches relied on the syntactic dependency tree of a sentence as a graph, and the impact of different graph representations has not been systematically evaluated. To address this gap, in this work, the impact of different graph representations, as well as combinations thereof, are investigated on three separate datasets.

Most RE approaches do not take into account background knowledge, e. g., from Knowledge Graphs (KGs). GNN-based approaches for RE generally emphasize on the graph representation of sentences (e.g., syntactic trees), and do not use the entity information and the graph context contained in external KGs. However, KGs may provide valuable knowledge about the entities for the RE task (Sun et al., 2020). Moreover, if we train a model such that it can make use of background knowledge, then, under some circumstances, this enables to improve the performance of a model without full retraining. For example, if a fact is missing that a model could use to correctly classify a relation, or if a wrong fact leads to a model incorrectly classifying a relation, than adding or replacing that fact can lead to the model making better predictions.

Therefore, in addition to different graph representations of the sentence, we also investigate enhanced graph representations by injection of KG facts into these graph representations by adding nodes and edges form the KG.

We show that combining multiple graph representations can outperform the models that only use the regular syntactic dependencies. Furthermore, we show that incorporating information from KGs like *Wikidata* (Vrandečić and Krötzsch, 2014) or *WordNet* (Fellbaum, 1998) improves results significantly.

2. Related Work

The integration of structured information, such as syntactic dependencies (Tian et al., 2021), semantic dependencies (Chan and Roth, 2011), and back-

DLnLD: Deep Learning and Linked Data @LREC-COLING-2024, pages 1–13

ground knowledge (Zhang et al., 2021; Peters et al., 2019; Tokuhisa et al., 2022; Wang and Pan, 2020; Sun et al., 2020; Wang and Pan, 2020), is an important topic in NLP.

Recently, much attention has been paid to the incorporation of KG information in language models (Yasunaga et al., 2022; Peters et al., 2019; Tokuhisa et al., 2022). For example, Yasunaga et al. (2022) use a joint language-knowledge foundation model in order to allow the NLP component to incorporate facts from the KG.

While this integration can be implemented as a training task (Yasunaga et al., 2022; Tokuhisa et al., 2022) or by finetuning and adapting pre-trained linguistic models (Houlsby et al., 2019; Wang et al., 2020), this usually requires complex architectures and comes with increased computational costs (Hamilton et al., 2022).

Another option is to directly operate on the symbolic graph structure by encoding the information in a graph and then processing it with Graph Neural Networks (GNNs) (Zhang et al., 2018a). GNNs allow to directly learn over graph structure (Dai et al., 2016; Gori et al., 2005; Li et al., 2016; Scarselli et al., 2009; Hamilton et al., 2017) and can be easily combined with standard neural network layers (Defferrard et al., 2016; Gong and Cheng, 2019).

One of the first GNN approaches was proposed by Kipf and Welling (2016), namely a Graph Convolutional Network (GCN), followed by the extension Relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al., 2018), that takes into account edge types. Furthermore, the Relational Graph Attention Network (R-GAT) (Busbridge et al., 2019) adds an attention mechanism to the R-GCN model. GNNs have been applied to a variety of tasks, such as Link Prediction (Schlichtkrull et al., 2018), Neural Machine Translation (Bastings et al., 2017; Marcheggiani et al., 2018), and Semantic Role Labeling (Marcheggiani and Titov, 2017).

Zhang et al. (2018a) have been one of the first to apply GNNs to RE. Their model applies a GNN encoder over syntactic dependency paths with unlabeled edges, and achieves comparable results to approaches based on bidirectional LSTMs and LLMs. Guo et al. (2019) and Tian et al. (2021) extended the use of GNNs for RE by applying a GNN with an attention mechanism and the capacity to encode labeled edges. Nadgeri et al. (2021), instead, explores the integration of external textual information (e. g., from *Wikidata*) into a GNN model for RE.

Recently, Yu et al. (2022) have shown linguistic knowledge fusion for downstream tasks by comparing different kinds of graph structures for several tasks in the GLUE benchmark. They investigate syntactic dependencies, semantic dependencies, binary balance trees, and linear chains of tokens. The work by Yu et al. (2022) does not investigate the impact of the representations on RE approaches and previous work on RE still mainly focuses on syntactic dependency trees. Therefore, the literature lacks a thorough evaluation of different graph structures and their combinations for RE with GNNs.

We present a deep investigation of several graph representations for the RE task and analyze them individually and in combinations. We build upon the research conducted by Yu et al. (2022) as we investigate different graph representations for RE. Furthermore, we go beyond by examining enhanced graph representations that incorporate KG facts.

3. Models and Graph Representations

In our experiments, we utilize a GNN architecture comprising two stacked GNN layers with a linear layer for relation classification. The architecture is shown in Figure 1.

The GNN layers encode the graph representation of the input sentence containing the two entities to be classified. We use Glove token embeddings, or a pre-trained but non-trainable BERT to derive token embeddings, and RDF2Vec for the KG entities. These embeddings serve as node features for the given graph.

To focus this investigation on the different graph representations, we decided to freeze the encoding model and do not investigate trainable encoders, like an end-to-end trainable BERT encoder, to derive token embeddings. GNN-based RE models that use an end-to-end trainable encoder are able to achieve state-of-the-art performance (Zhang et al., 2018a; Guo et al., 2019; Tian et al., 2021).

After the two GNN layers, the resulting representations of the subject and object entities are used as input to the linear classification layer. In the case of multi-word entities, we rely on the representation of the token with the largest number of outgoing syntactic dependencies.

3.1. Graph Representations

In order to apply this GNN model for RE, we represent tokens as nodes and connect them through (typed) edges to obtain a graph. The investigated graph structures are:

1) Tokens connected in a linear chain (*chain*), in the same order as they occur in the text.

2) Every token connected to every other token, what leads to a fully connected graph (*fully*) and allows every token to access the features of every other token.

3) Tokens connected according to syntactic dependencies (*syn*).

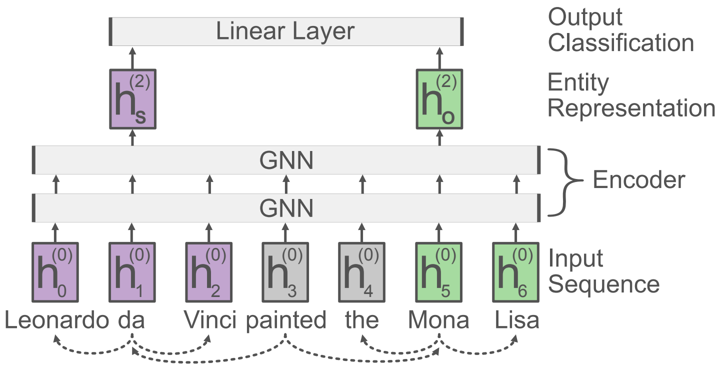

Figure 1: GNN model architecture.The model operates over a given graph with given input node features h_i^0 that are derived by embedding the token or KG entity by a suitable embedding model. $h_i^{(l)}$ denotes the features of token i at layer l of the GNN. In this context, $h_s^{(2)}$ and $h_o^{(2)}$ denote the feature representation of subject and object entity after two GNN layers. $h_s^{(2)}$ and $h_o^{(2)}$ are then feed into the linear layer for relation classification.

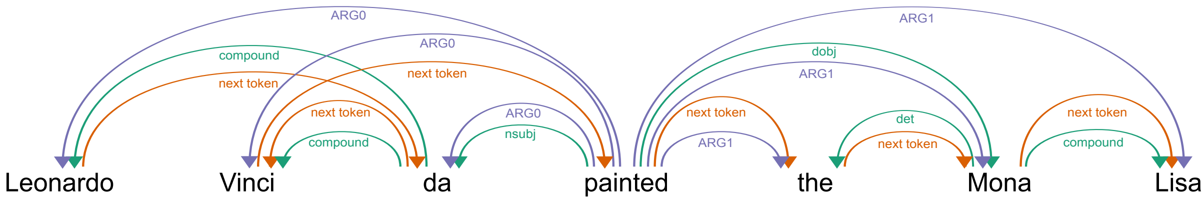

Figure 2: An example of a graph that combines the three graph representations chain, syn, and sem (colored *orange*, *green*, *purple*).

4) Tokens connected through higher order syntactic dependency relations (*highsyn*) according to Tian et al. (2022). Here, tokens are related if there are at most two tokens in between when traversing the syntactic dependency tree, directly connecting tokens that are syntactically close. We refer to Tian et al. (2022) and App. A for more details.
5) Tokens are connected according to their semantic dependencies (*sem*) in the form of the latent PropBank-based (Palmer et al., 2005) predicate argument structure derived by means of Semantic Role Labelling (Shi and Lin, 2019).

We evaluate all possible combinations of these methods. An example graph is shown in Figure 2.

3.2. Graph Representations with Additional Background Knowledge

We investigate the integration of background knowledge from a KG into the graphs. To do so, new nodes are created, representing the entities involved in the relations. The node features are derived from the background KG (*nodes*). These nodes are then connected to the corresponding entity mentions. We only connect subject and object entity mentions to their corresponding KG entities. Additionally, we consider adding the shortest paths (*s.p.*) to the graph. We use the shortest path between subject and object entity in the KG, and include any external entity present on these paths as additional nodes, as well as any edge connecting them. We refrain from explicitly adding an edge between a node on the shortest path and any node representing an entity mentioned in the text. An example is shown in Figure 3.

4. Experiments

We investigate the impact of using different graph structures as graph representations on the task of RE by training and evaluating multiple GNN models.[1]

In order to derive syntactic dependencies, we rely on Spacy,[2] while for semantic dependencies we make use of the AllenNLP library[3] described in (Gardner et al., 2018). As node features, we use 100 dimensional *Glove* embeddings[4] (Pennington et al., 2014), or 768 dimensional contextual *BERT* embeddings (Devlin et al., 2019).

We automatically determine the best GNN hyperparameter settings using the hyperparameter

[1]We use PyTorch Geometric to implement our GNNs, github.com/pyg-team/pytorch_geometric.

[2]See spacy.io/.

[3]See github.com/allenai/allennlp.

[4]We also experimented with the 300 dimensional embeddings, and found the results to be interchangeable. For runtime optimization reasons, we opted for the lower dimensional embeddings in the final experiments.

search framework ASHA (Li et al., 2020), which applies intelligent early-stopping and supports large-scale parallelization. The main hyperparameter of our model is the type of the GNN layers (i.e., GCN, R-GCN, R-GAT) as described in Section 2. Furthermore, these models have hyperparameters like the dimensionality of the GNN layers and linear layers (64, 120, 240), the learning rate (8 samples from 10^{-3} to 10^{-5}), and the batch size (32, 64, 128). In addition, we evaluate the impact of i) adding reverse edges, ii) adding self-loops to each node such that its previous feature vector can be accessed by itself, and iii) exploiting the labels of edges.

We used the F_1 score as the criterion to select the best model. Since unpromising runs are terminated at an early stage, not all model configurations are trained until convergence and evaluation results are not produced for all the considered model configurations.

We evaluate the graph representations on two English RE datasets that are linked to *Wikidata* and on the commonly used RE benchmark *SemEval 2010 Task 8* dataset to validate our models.

The required property of the evaluation datasets was that all subjects and objects of a relation are annotated with their corresponding *Wikidata* ID, such that background information can be used. However, there is a lack of RE datasets that are annotated with *Wikidata* entities as most datasets are annotated with *Freebase* entities and relations (Mintz et al., 2009). Therefore, we created our own datasets based on *FewRel* and *T-REx*.

Moreover, to validate that our models are solving the RE task sufficiently, we run the standard evaluation without background knowledge on the *SemEval 2010 Task 8* dataset.

In detail, we consider the following datasets:

1) FewRel (custom): *FewRel* (Han et al., 2018; Gao et al., 2019) is a large RE dataset with entity mentions and relations annotated with their corresponding *Wikidata* IDs. It was created through a combination of distant supervision and human annotation. Originally developed for few-shot RE, we repurpose it for standard RE by merging its `train` and `val` splits. These splits encompass sentences expressing 64 and 16 distinct relations, each with 700 examples, totaling $56,000$ sentences. The combined dataset is then randomly split into `train`/`dev`/`test` splits with percentages $70/15/15$. In *FewRel*, all subjects and objects of a relation are annotated with their corresponding *Wikidata* ID, and, therefore, there cannot be a subject or object which has no Wikidata ID in our *FewRel (custom)* dataset, too.

2) T-REx (custom): We randomly sampled 1000 sentences for each relation occurring at least $1,000$ times from the *T-REx* dataset (Elsahar et al., 2018), which was created by an automatic alignment of *Wikipedia* abstracts and *Wikidata* triples. We only selected sentences in which both, subject and object are annotated with *Wikidata* IDs. Therefore, all subjects and objects of a relation are annotated with their corresponding *Wikidata* ID. This dataset contains $228,000$ sentences expressing 228 different relations.

3) SemEval 2010 Task 8: This dataset consists of $8,000$ human-annotated training and $2,717$ human-annotated test sentences with a relation between two given nominals. We use 20% of the train set for validation (Hendrickx et al., 2010). However, the publicly available test set was not modified to ensure comparability to other work on RE. Since this dataset is not annotated with any KG IDs, we use it only to evaluate the different types of graph representations for RE, and not the knowledge injection.

As background knowledge, we rely on two KGs, namely *Wikidata*[5] (Vrandečić and Krötzsch, 2014) and *WordNet* (Fellbaum, 1998). *Wikidata* is build by many editors and partially automatic. It encompasses data about entities such as people, places, organizations, or abstract topics, along with details about their interconnections and relationships. *WordNet* is a manually created lexical database that categorizes nouns, verbs, adjectives, and adverbs into synsets. These synsets are connected through conceptual-semantic and lexical relations, forming a KG that captures the interconnections between different linguistic elements.

The features for the added KG nodes are derived via *RDF2Vec* (Ristoski and Paulheim, 2016).[6] *RDF2Vec* is a method that derives embeddings for the entities and relations in a KG. In case that the KG contains facts that are relations-to-be-predicted, they are removed from the dataset, so they do not affect the embeddings. We remove triples contained in the RE datasets from our *Wikidata* graph before we derive the embeddings. No triples needed to be removed to derive the *WordNet* features, as no relation in *WordNet* can inadvertently reveal the relations that will be predicted for any of our datasets. The derived features share the same dimensionality of the other nodes' embeddings and are used as vector of newly created nodes which are connected to their associated entity mention's tokens.

For *Wikidata*, the integration of shortest paths between entities in KGs can be valuable for RE. Therefore, nodes are created for every entity on the path between the mentions, and connected among themselves, as shown in Figure 3. In the majority of cases, the shortest paths consist of only one

[5] We use the *Wikidata* dump from October 2022.

[6] *RDF2Vec* embeddings are trained using the *jRDF2Vec* implementation described by Portisch et al. (2020), `https://github.com/dwslab/jRDF2Vec`.

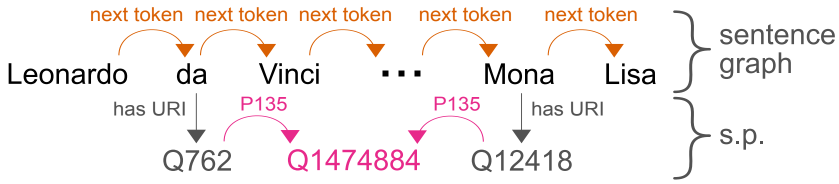

Figure 3: Integration of the shortest path (shown in pink) between *Leonardo da Vinci* and *Mona Lisa* in *Wikidata* into the chain graph. The *Wikidata* entity IDs *Q762*, *Q12418*, and *Q1474884* represent *Leonardo da Vinci*, *High Renaissance*, and *Mona Lisa*, whereas the *Wikidata* property ID *P135* expresses the *movement* relation.

entity positioned between the subject and object of the relation intended for classification.

We evaluated two simple models, both using a feedforward neural network with two layers and a classification layer (denoted as *Linear-NN*), to compare our GNN models that encode graph structure against those operating on text-based embeddings. One model takes as input the concatenation of the word embeddings of the two entity mentions, while the second one uses *RDF2Vec* features for subject and object.

Our code is available on GitHub.[7]

5. Results

The best performing model across all experiments is based on a two layer R-GCN encoder with self-loops and reverse edges, and a hidden dimension of 120 (*Glove*), respectively 240 (*BERT*) for the GNN layers and linear layers. The model is trained with a batch size of 64 and a learning rate of 0.0001.

Graph Representations Our results, shown in Table 1, Table 3, and Table 4, show that the models using *BERT* features perform better compared to those using *Glove* features for all graph representations. The input graph representations do not lead to consistent performance across all datasets.

The evaluation results of the graph structures on *FewRel (custom)*, displayed in Table 1, shows that the best performance is reached by using syntactic dependencies with an F_1 score of 0.754, followed by higher order syntactic dependencies (F_1 of 0.745), and by the linear chain (F_1 of 0.703). Regarding the models that operate on a combination of multiple graph representations, the combination of syntactic dependencies and the fully connected token graph leads the best results and achieves an F_1 score of 0.764.

On the *T-REx (custom)* dataset, the evaluation scores are shown in Table 3, the best performance is achieved by the model operating over syntactic dependencies with an F_1 score of 0.697, followed by the fully connected graph (F_1 of 0.693) and by the linear chain (F_1 of 0.689). For the combined graph representations, the combination of the linear chain and higher order syntactic dependencies shows the best results with an F_1 score of 0.761.

The scores for the *SemEval* dataset, shown in Table 4, show that the best performance is reached by a model operating over the graph of syntactic dependencies with an F_1 score of 0.786. The second-best model uses the higher order dependencies (F_1 of 0.766), and the third-best model uses the semantic dependencies (F_1 of 0.745). For the combined representations, the combination of syntactic dependencies and semantic dependencies shows the best results with an F_1 score of 0.786.

Graph Representations with Additional Background Knowledge The impact of adding KG features to the graph consisting of the fully connected graph and the syntactic dependency graph (denoted *fully+syn*) on the performance of the models is shown in Table 2 for the *FewRel (custom)* dataset and the *T-REx (custom)* dataset. The *SemEval* dataset was not evaluated with additional KG features, as the entities in this dataset are not linked to a KG.

The additional *Wikidata* or *WordNet* features lead to an improvement of the scores in all cases.

All GCN models that use *Wikidata* or *WordNet* features outperform the NN baselines that use *Wikidata RDF2Vec* features only or *word embedding* features only.

On the *FewRel (custom)* dataset, the *BERT* model that uses the combined graph of syntactic dependencies and the fully connected graph can be improved from an F_1 score of 0.764 to an F_1 of 0.82 (additional Wikidata nodes), 0.859 (additional Wikidata shortest path), respectively 0.763 (additional WordNet nodes) by using additional background knowledge.

By adding additional background knowledge to the combined graph of syntactic dependencies and the fully connected graph, the F_1 scores of the *BERT* model on the *T-REx (custom)* dataset improve from 0.714 to 0.746 (additional Wikidata nodes), 0.791 (additional Wikidata shortest path), respectively 0.729 (additional WordNet nodes).

[7] See github.com/Nolanogenn/re_with_gcn.

Table 1: General evaluation of the different graph representation and their combinations on the *FewRel (custom)* dataset.

Graph Representation	Glove			BERT		
	F_1	P	R	F_1	P	R
chain	0.44	0.445	0.468	0.703	0.704	0.71
fully	0.484	0.476	0.515	0.699	0.705	0.71
syn	0.566	0.566	0.579	0.754	0.757	0.757
sem	0.388	0.407	0.418	0.667	0.666	0.676
highsyn	0.594	0.539	0.612	0.745	0.746	0.749
chain + syn	0.571	0.57	0.589	0.75	0.753	0.754
chai + sem	0.516	0.525	0.533	0.723	0.726	0.728
fully + syn	**0.611**	**0.612**	**0.627**	**0.764**	**0.766**	**0.767**
fully + sem	0.489	0.483	0.516	0.708	0.711	0.715
syn + sem	0.574	0.569	0.591	0.753	0.754	0.756
chain + highsyn	0.574	0.565	0.6	0.745	0.747	0.749
fully + highsyn	0.603	0.6	0.622	0.75	0.753	0.754
highsyn + sem	0.577	0.582	0.608	0.743	0.744	0.748
chain + syn + sem	0.583	0.587	0.6	0.751	0.754	0.754
fully + syn + sem	0.608	0.612	0.623	0.753	0.755	0.757
chain + highsyn + sem	0.578	0.572	0.603	0.745	0.747	0.748
fully + highsyn + sem	0.586	0.576	0.611	0.746	0.749	0.748

Table 2: Evaluation of graph representations enhanced with additional KG features from *Wikidata* and *WordNet* on *FewRel (custom)* and *T-REx (custom)*. The *Glove* models are provided with 100 dimensional embeddings, whereas the *BERT* models are provided with 768 dimensional embeddings.

Model & Graph Representation	Glove			BERT		
	F_1	P	R	F_1	P	R
	FewRel (custom)					
Linear-NN: word embeddings	0.277	0.323	0.294	0.382	0.436	0.3
Linear-NN: RDF2Vec embeddings	0.597	0.618	0.606	0.664	0.675	0.669
GCN: syn	0.566	0.566	0.579	0.754	0.757	0.757
GCN: fully + syn	0.611	0.612	0.627	0.764	0.766	0.767
+ *Wikidata* nodes	0.784	0.778	0.803	0.82	0.823	0.823
+ *Wikidata* shortest path	**0.835**	**0.834**	**0.845**	**0.859**	**0.86**	**0.861**
+ *WordNet* nodes	0.684	0.685	0.697	0.763	0.765	0.766
	T-REx (custom)					
Linear-NN: BERT	0.082	0.15	0.103	0.239	0.304	0.251
Linear-NN: RDF2Vec	0.438	0.488	0.475	0.506	0.548	0.525
GCN: syn	0.406	0.399	0.451	0.697	0.698	0.72
GCN: fully + syn	0.45	0.436	0.498	0.714	0.708	0.735
+ *Wikidata* nodes	0.661	0.64	0.712	0.746	0.35	0.776
+ *Wikidata* shortest path	**0.685**	**0.664**	**0.734**	**0.791**	**0.782**	**0.814**
+ *WordNet* nodes	0.561	0.572	0.592	0.729	0.73	0.746

Overall, the best result can be achieved when adding the shortest path in *Wikidata* between subject and object to the graph representation.

6. Discussion

Graph Representations The input graph representations do not lead to consistent performance across all datasets. This might be caused by the sentence structure or sentence complexity in the datasets.

We observe that the models that use *Glove* features show good scores for the combined representation of *fully* + *syn* + *sem* across all datasets, and always perform slightly better than the individual representations.

For the *BERT* models, the different representations do lead to different performances across the datasets, and we can not observe a general trend. However, on the *FewRel (custom)* dataset and the *T-REx (custom)* dataset, combining syntactic dependencies with other graph representations leads

Table 3: General evaluation of the different graph representation and their combinations on the *T-REx (custom)* dataset.

Graph Representation	**Glove**			**BERT**		
	F_1	P	R	F_1	P	R
chain	0.351	0.362	0.386	0.689	0.695	0.709
fully	0.366	0.359	0.417	0.693	0.696	0.712
syn	0.406	0.399	0.451	0.697	0.698	0.72
sem	0.33	0.338	0.362	0.663	0.665	0.688
highsyn	0.433	0.419	0.483	0.674	0.673	0.703
chain + syn	0.42	0.408	0.465	0.714	0.718	0.731
chain + sem	0.389	0.38	0.431	0.688	0.686	0.711
fully + syn	0.45	0.436	**0.498**	0.714	0.708	0.735
fully + sem	0.39	0.386	0.433	0.687	0.686	0.712
syn + sem	0.426	0.41	0.472	0.699	0.696	0.721
chain + highsyn	0.431	0.414	0.483	**0.761**	**0.763**	**0.762**
fully + highsyn	0.431	0.411	0.488	0.666	0.66	0.699
highsyn + sem	0.424	0.41	0.477	0.645	0.645	0.693
chain + syn + sem	0.436	0.417	0.483	0.697	0.689	0.721
fully + syn + sem	**0.453**	**0.443**	0.497	0.658	0.646	0.689
chain + highsyn + sem	0.43	0.409	0.484	0.653	0.639	0.687
fully + highsyn + sem	0.427	0.407	0.482	0.646	0.63	0.681

Table 4: General evaluation of the different graph representation and their combinations on the *SemEval 2010 Task 7* dataset.

Graph Representation	**Glove**			**BERT**		
	F_1	P	R	F_1	P	R
chain	0.686	0.689	0.688	0.715	0.716	0.718
fully	0.656	0.669	0.658	0.69	0.691	0.692
syn	0.756	0.76	0.757	**0.786**	0.783	**0.791**
sem	0.705	0.72	0.702	0.745	0.749	0.745
highsyn	0.752	0.758	0.75	0.766	0.769	0.766
chain + syn	0.752	0.757	0.749	0.776	0.775	0.779
chain + sem	0.746	0.752	0.743	0.777	0.777	0.78
fully + syn	0.76	0.766	0.756	0.776	0.775	0.781
fully + sem	0.716	0.722	0.718	0.748	0.75	0.749
syn + sem	0.764	0.768	0.762	**0.786**	**0.788**	0.787
chain + highsyn	0.745	0.747	0.746	0.761	0.763	0.762
fully + highsyn	0.742	0.747	0.741	0.771	0.772	0.773
highsyn + sem	0.751	0.754	0.749	0.771	0.77	0.776
chain + syn + sem	0.759	0.765	0.755	0.781	0.779	0.784
fully + syn + sem	**0.768**	**0.773**	**0.766**	0.785	0.785	0.789
chain + highsyn + sem	0.741	0.752	0.747	0.768	0.766	0.775
fully + highsyn + sem	0.754	0.757	0.753	0.764	0.768	0.763

to improved scores compared to using only syntactic dependencies, whereas this worsens the scores on the *SemEval* dataset.

For the *Glove* models, the combination of multiple graph representations generally leads to better scores than using the individual representations. This trend can not be observed for the *BERT*, and we can assume that this information is already encoded in the *BERT* embeddings. Therefore, the graph representations do not provide additional context information, but rather confuse the model by adding redundant information.

All in all, combinations of graph representations can add additional information that can be used by GNNs for RE. But it must be noted, that the benefit is low and differs depending on the dataset. However, we were able to show that even simple graph representations without linguistic knowledge, like a linear chain of tokens or the fully connected graph of tokens, still lead to adequate models.

Our GNN model is limited to two layers, which leads to a receptive field of two graph hops. Therefore, models operating over representations that connect distant entities should clearly outperform those models that can only access a certain number of tokens in the graph neighborhood. However, this assumption is not always the case. Therefore, we assume carefully selecting a suitable graph representation instead of simply providing all available tokens might be valuable.

Graph Representations with Additional Background Knowledge We evaluate different representations acquired through parsing the sentence structure, and enriched by background knowledge.

Adding *Wikidata* information could make a direct comparison seem unfair. The additional KG information could be helpful for the model as they provide additional information about the subject and object entity not expressed in the sentence. But the additional information could also be a drawback to the models. Background knowledge might contain irrelevant information for the task, potentially introducing noise and complicating the model's focus on relevant features.

In contrast, the integration of *WordNet* features does not add any unfair advantages to the model, as this is commonly done in NLP. Providing additional external resources like *WordNet* information, part-of-speech tags, dependency information, and named entity tags is often done for RE (Shen and Huang, 2016; Zhang et al., 2015).

Nevertheless, to prevent confusion, we present the results of models with additional background knowledge separately in dedicated tables.

In general, the incorporation of information from *Wikidata* as additional nodes connected to the subject and object nodes, or as the shortest path between both, has a positive impact. For instance, on *FewRel (custom)*, the best model that uses the *Wikidata* shortest paths achieves an F_1 score of 0.859. This is an increase of 0.095 in F_1 compared to the base *fully+syn* GNN model without KG features.

WordNet features do increase the performance of all models, too. *WordNet* provides additional information about synonyms and related concepts, as well as various semantic relationships between words to the RE model. According to our results, the *WordNet* information is helpful for GNN-based RE.

However, adding the richer and more diverse *Wikidata* features to the graph increases the scores more than adding *WordNet* features. This might be because Wikidata provides more background knowledge, i.e., the relations entities are involved in the KG, which might be more valuable than the *WordNet* information.

All GCN models that use some sentence graph representation and additional KG features outperform the *Wikidata RDF2Vec*-based and *word embedding based* NN model. Especially, adding the *Wikidata shortest path* leads to best scores. This shows a successful fusion of text and KG information in a common graph representation, as the model that applies fusion outperforms the individual models.

7. Conclusion

Our results show that combining multiple graph representations can improve the model's predictions. Although our experiments revealed that none of the graph representations consistently performs best across multiple datasets, we can clearly see i) that most representations improve the performance compared to the standard graph representation, and ii) that the representations have a strong impact on performance, which makes the type of graph representation an important hyperparameter that is worth to be tuned.

Furthermore, the integration of background knowledge from *Wikidata* or *WordNet* positively impacts scores and can lead to an improvement of close to 0.1 in F_1.

In future work, we will investigate methods to integrate structured background knowledge beyond additional subject and object nodes and shortest paths between them. Furthermore, we will investigate how the model performance can be improved by removing wrong facts and adding missing facts to the KG.

8. Limitations

The present work has some minor limitations that should be acknowledged.

Firstly, our models do not reach state-of-the art performance. However, beating state-of-the-art performance was not the goal of this work. Instead, we investigate of different graph representations. As the difference to state-of-the-art is small, one can assume our GNN model to be set up correctly.

Secondly, even though our GNN models have significantly fewer parameters than *BERT* (*9M* vs. *110M*), our best models rely on token features derived from *BERT*. However, our training is faster than training *BERT* from scratch.

Thirdly, it is important to note that incorporating facts from a KG could make the model biased to the information stored in the form of triples in the KG instead of the information expressed in the sentence context. Future research could use explainability methods or attention mechanisms to determine which information the model prioritizes.

Ethics

Any potential biases present in the relation extraction datasets or knowledge graphs used in our approach can impact the fairness and accuracy of the extracted relations. However, it is important to note that our work primarily focuses on the evaluation of different graph representations, which do not introduce new ethical biases themselves. Nevertheless, careful consideration should still be given to the potential biases inherited from the datasets and knowledge graph sources to ensure the ethical and unbiased nature of our approach.

Acknowledgements

This work was carried out in the context of DiProMag, a BMBF (German Federal Ministry of Education and Research) funded research project under Grant No. 13XP5120B. This work was partially supported by: the Research Council of Norway through its Centres of Excellence scheme, Integreat – Norwegian Centre for knowledge-driven machine learning, project number 332645; the Ministry of Culture and Science of the State of North Rhine-Westphalia (Germany) through SAIL, Grant No. NW21-059A; the Deutsche Forschungsgemeinschaft (DFG) through the priority program RATIO (SPP-1999), Grant No. 376059226.

9. Bibliographical References

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Jasmijn Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an. 2017. 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. Association for Computational Linguistics.

Razvan Bunescu and Raymond Mooney. 2005. A Shortest Path Dependency Kernel for Relation Extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. 2019. Relational graph attention networks. *arXiv preprint arXiv:1904.05811*.

Yee Seng Chan and Dan Roth. 2011. Exploiting Syntactico-Semantic Structures for Relation Extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 551–560, Portland, Oregon, USA. Association for Computational Linguistics.

Hanjun Dai, Bo Dai, and Le Song. 2016. Discriminative embeddings of latent variable models for structured data. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2702–2711. JMLR.org.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3844–3852, Red Hook, NY, USA. Curran Associates Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019. FewRel 2.0: Towards More Challenging Few-Shot Relation Classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6250–6255, Hong Kong, China. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Liyu Gong and Qiang Cheng. 2019. Exploiting edge features for graph neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9203–9211.

M. Gori, G. Monfardini, and F. Scarselli. 2005. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734 vol. 2.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention Guided Graph Convolutional Networks for Relation Extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251, Florence, Italy. Association for Computational Linguistics.

Kyle Hamilton, Aparna Nayak, Bojan Božić, and Luca Longo. 2022. Is neuro-symbolic AI meeting its promises in natural language processing? A structured review. *Semantic Web*, pages 1–42.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035, Red Hook, NY, USA. Curran Associates Inc.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809, Brussels, Belgium. Association for Computational Linguistics.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, Uppsala, Sweden. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. 2020. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems*, 2:230–246.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. 2018. Exploiting Semantics in Neural Machine Translation with Graph Convolutional Networks. 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 486–492, New Orleans, Louisiana. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2017. 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, Copenhagen, Denmark. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. 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*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In *Proceedings of the 54th*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Abhishek Nadgeri, Anson Bastos, Kuldeep Singh, Isaiah Onando Mulang', Johannes Hoffart, Saeedeh Shekarpour, and Vijay Saraswat. 2021. KGPool: Dynamic Knowledge Graph Context Selection for Relation Extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 535–548, Online. Association for Computational Linguistics.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. 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, New Orleans, Louisiana. Association for Computational Linguistics.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Jan Portisch, Michael Hladik, and Heiko Paulheim. 2020. RDF2Vec Light–A Lightweight Approach for Knowledge Graph Embeddings. *International Semantic Web Conference, Posters and Demos*.

Petar Ristoski and Heiko Paulheim. 2016. RDF2Vec: RDF Graph Embeddings for Data Mining. In *The Semantic Web – ISWC 2016*, pages 498–514, Cham. Springer International Publishing.

Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. 2015. Injecting Logical Background Knowledge into Embeddings for Relation Extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1119–1129, Denver, Colorado. Association for Computational Linguistics.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Yatian Shen and Xuanjing Huang. 2016. Attention-based convolutional neural network for semantic relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2526–2536, Osaka, Japan. The COLING 2016 Organizing Committee.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3660–3670, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yuanhe Tian, Guimin Chen, Yan Song, and Xiang Wan. 2021. Dependency-driven Relation Extraction with Attentive Graph Convolutional Networks. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Yuanhe Tian, Yan Song, and Fei Xia. 2022. Improving Relation Extraction through Syntax-induced Pre-training with Dependency Masking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1875–1886, Dublin, Ireland. Association for Computational Linguistics.

Ryoko Tokuhisa, Keisuke Kawano, Akihiro Nakamura, and Satoshi Koide. 2022. Enhancing Contextual Word Representations Using Embedding of Neighboring Entities in Knowledge Graphs. In

Proceedings of the 29th International Conference on Computational Linguistics, pages 3175–3186, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM*, 57(10):78–85.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. Cite arxiv:2002.01808.

Wenya Wang and Sinno Jialin Pan. 2020. Integrating deep learning with logic fusion for information extraction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9225–9232.

Zihan Wang and Bo Yang. 2020. Attention-based Bidirectional Long Short-Term Memory Networks for Relation Classification Using Knowledge Distillation from BERT. In *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pages 562–568.

Shanchan Wu and Yifan He. 2019. Enriching Pre-Trained Language Model with Entity Information for Relation Classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 2361–2364, New York, NY, USA. Association for Computing Machinery.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 536–540, Lisbon, Portugal. Association for Computational Linguistics.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question Answering on Freebase via Relation Extraction and Textual Evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2326–2336, Berlin, Germany. Association for Computational Linguistics.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. 2022. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems*, 35:37309–37323.

Changlong Yu, Tianyi Xiao, Lingpeng Kong, Yangqiu Song, and Wilfred Ng. 2022. An Empirical Revisiting of Linguistic Knowledge Fusion in Language Understanding Tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10064–10070, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved Neural Relation Detection for Knowledge Base Question Answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 571–581, Vancouver, Canada. Association for Computational Linguistics.

Congcong Zhang, Gaofei Xie, Ning Liu, Xiaojie Hu, Yatian Shen, and Xiajiong Shen. 2021. Automatic Hypernym-Hyponym Relation Extraction With WordNet Projection. In *2021 7th International Conference on Systems and Informatics (ICSAI)*, pages 1–6.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78, Shanghai, China.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018a. 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, Brussels, Belgium. Association for Computational Linguistics.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018b. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*,

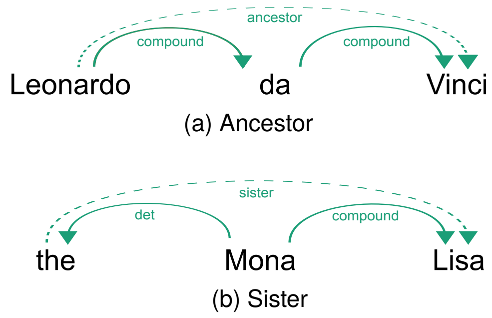

Figure 4: Higher order syntactic dependencies. Solid lines represent the syntactic first order dependencies, whereas the dashed lines represent the second order dependencies.

pages 2205–2215, Brussels, Belgium. Association for Computational Linguistics.

A. Higher Order Syntactic Dependencies

We implement higher order syntactic dependencies as proposed by Tian et al. (2022). The syntactic dependencies serve as first order dependencies. Based on those, second and third order dependencies are added.

For example, second order dependencies establish directed connections between two tokens, $token_i$ and $token_j$, if there exists a single token, $token_x$, along the non-directional shortest path connecting $token_i$ and $token_j$. In detail, we define two distinct relation types based on the direction of the edges in the graph. If the connection between the tokens is $token_i \rightarrow token_x \rightarrow token_j$, we establish the *ancestor* relation pointing from $token_i$ to $token_j$. If the relations are $token_i \leftarrow token_x \rightarrow token_j$, we add the *sister* relation between $token_i$ and $token_j$. Examples of the two relations are shown in Figure 4.

The third order dependencies are defined similarly for the case of two tokens in between $token_i$ and $token_j$, along the non-directional shortest between them.

We do not add inverse relations, as this is a hyperparameter of the graph preprocessing.