Named Entity Disambiguation For Graphs

Sruthi Poddutur[*]
Department of Computer Science
Stanford University
`poddutur@stanford.edu`

Abstract

A supervised learning task of Named Entity Disambiguation (NED) is presented in this work where entities in short sentences are linked to Wikidata which is a graph-structured data. The key component of this work lies in the usage of graph data and its topology for entity encoding. In particular, different neural attention mechanisms along with Graph Convolution Network (GCN) were examined to create context based embedding for entities of interest in Wikidata graphs. Work shows that RNN-attention based models outperform regular RNN approaches. Also, one-hop GCN with the node-wise degree normalization outperforms the results obtained in the literature.

1 Introduction

I considered the problem of NED in this work, where the entity references in text (short-sentences) are resolved using Wikipedia graph knowledge base (KB). NED is challenging because of the inherent ambiguity between mentions in the text and referred entities in our KB. The results should depend on the context in which it appears. The inputs of this model i.e., *entity, correctid, wrongid and question* are clearly depicted in the sample dataset as shown in Table1 below. Output of this model is whether the given wikipedia id and mention inputs are relevant or not. The key element of this work lies in generating efficient context-based embedding for WikipediaId and question to resolve the relevancy between them. I demonstrate how to make use of graph-KB-based neural network model for fast and scalable NED. Also, I compared NED accuracy among attention-based vs graph-based techniques to create context-vector from Wikipedia KB graph.

2 Related work

NED is not a recent endeavor. The work of Bunescu and Pasca [Bunescu and Pasca 2006] introduced the idea of disambiguating text through the use of knowledge bases. Shortly thereafter, the authors of [Milne and Witten 2008] introduced the idea of learning to link entities on Wikipedia. Many ways of measuring the relatedness of context and entity, such as dot product, cosine similar- ity, Kullback-Leibler divergence, Jaccard distance, or more complicated ones (Zheng et al., 2010; Kulkarni et al., 2009; Hoffart et al., 2011; Bunescu and Pasca, 2006; Cucerzan, 2007; Zhang et al., 2011). However, these measures are often dupli- cate or over-specified, because they are disjointly combined and their atomic nature determines that they have no internal structure. Furthermore, recent neural methods (He et al., 2013; Sun et al., 2015; Yamada et al., 2016; Ganea and Hofmann, 2017; Le and Titov, 2018; Yang et al., 2018; Radhakrishnan et al., 2018, Sloan et al., 2018) have established state-of-the-art results, outperforming earlier similarity measure based models. Context aware embeddings, together with neural similarity functions, are essential in these frameworks. Architectures using Recurrent/recursive neural networks (RNNs) and convolutional neural networks (CNNs) along with

[*]http://github.com/spoddutur/deep-graph-ned

Table 1: Sample Record of Dataset

Entity	CorrectId	WrongId	Question
Victoria	Q36687	Q7926536	Fitzroy North is a suburb in Melbourne, Victoria, Australia, 4 km north-east from Melbourne's central business district. Its Local Government Area are the Cities of Yarra and Moreland.

attention mechanisms play an important role in building context aware embeddings to represent essential linguistic structures. This work is to evaluate different RNN without attention, RNN with attention and GCN mechanisms (where the novelty I have tried in this work lies in node degree normalization and self-node induction into adjacency matrix) to build context aware embeddings for the task of NED. In particular, we focus on global attention mechanism because the input for our model is short sentences.

3 Dataset and Features

Dataset: Among the datasets available for this task, I made use of WikidataDisamb [Sloan et al 2018] dataset. Training dataset size is around 250,000 records and test dataset is of size 10,000. Table1 above illustrates a sample record of dataset.
Features: To generate the input vectors, I employed 300-dimensional Glove vector (Pennington et al 2014). The first input vector is a sequence of word vectors v_i present in the question. Next input vector is for representing wikidata graph as embedding. For instance, embedding for the node *Donald Trump* is computed by averaging the glove vectors of *Donald* and *Trump*. Edge embedding is discussed later in detail in section 4.1.3.

4 Methods

My architecture is inspired mainly by the work of [Luong et al. 2018], [Vaswani et al. 2017] and [Cetoli et al. 2018] which aims to combine GRU-based embedding of question(Y_{text}) with triplets-of-graph vector (Y_{graph}), using softmax function in the last layer to evaluate relevancy between them. Among lot of models I tried, am show casing a total of seven varied models that differ mainly in the way Y_{graph} is computed. In that, these methods can be broadly categorized into basic RNN-based, RNN+Attention-based and GCN-based architectures.

4.1 Common Layers used across all the models:

All these models use two things in common.

1. **Encoding Mention in Question** Y_{text}: Firstly, the method to encode mention in the question remains same. Input question vectors v_i is fed through two stack GRU layer to generate outputs y_i. I then apply a mask on y_i which acts like hardmax attention layer and get the masked vector representing the entity in question. For example, consider the input question *United States president Donald Trump is..* and the entity of interest for us is *Donald Trump*. In this case, our mask will be an array of $a_i = [0, 0, 0, 1, 1, 0, ..]$. As you can see, this mask turns off vectors of all other words in question vector except for the entity of our interest. Lastly, we normalize the resulting masked vector to get the final embedding representing entity mention in the question where N_{text} is the length of question as shown here: $Y_{text} = 1/N_{text} * \sum_{n=1}^{N_{text}} a_i y_i$
2. **Relevancy Layer:** The next thing that is common among all the models I have tried is relevancy layer i.e., the final output layers which computes the relevancy between Y_{text} and Y_{graph}. Essentially, Y_{text} and Y_{graph} are concatenated and passed to a dense layer followed by the softmax function to get a binary vector output
3. **Embedding for Wiki Entity Triplets (except for GCN-based model):** For all the RNN based models tried in this work, entity triplets from wiki graph data are given as a sequence

of triplets $[x_i, e_{i,j}, x_j]$ where i, j are the indices of all the connected nodes of the entity in wiki graph. Every such triplet is converted into vector by simply concatenating 300-dimensional glove vectors of both the nodes and the edge connecting them as shown below $x_{i,j}^{triplet} = x_i \oplus e_{i,j} \oplus x_j$

4.2 Model Variations

The model variations tried in this work has been categorized into three groups based on how Y_{graph} is computed as listed below:

4.2.1 RNN Based Models for Y_{graph}

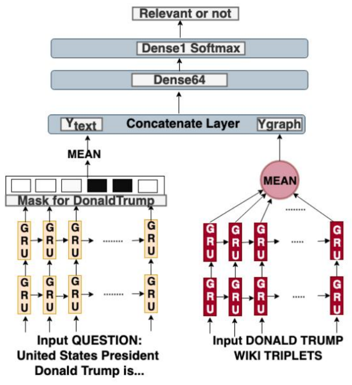
1. **GRU for both Y_{text} and Y_{graph}:** In this model, represented in Figure 1a, each of the triplet embeddings which are computed as discussed in section 4.1.3, are passed through a two stack GRU [Cetoli et al. 2018] followed by mean layer to get Y_{graph}.
2. **GRU for both Ytext and Bi-LSTM for Y_{graph}:** In this model, represented in Figure 1b, we improve upon the prior model by deploying a bidirectional-LSTM layer [Augenstein et al. 2016] with *return-sequence=False* to compute Y_{graph}. This was done to compare Uni-Directional RNN with mean of all triplets where all the triplets contribute directly to Y_{graph} vs Bi-Directional LSTM which outputs forward and backward layer's final cell output.

4.2.2 RNN+Attention Based Models for YGraph

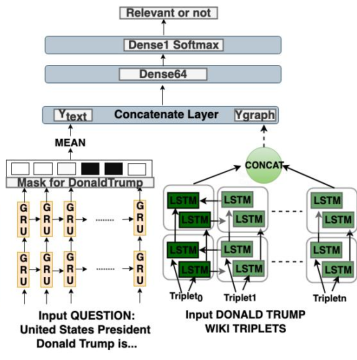
1. **RNN + ConditionalAttention:** This model, represented in Figure 1f, is simplest of all the attention mechanisms (inspired by Luong et al. 2016). Similar to *GRU for both Y_{text} and Y_{graph}* model shown in Figure 1a, triplet vectors are passed through two layer GRU followed by mean. However, these GRU cells are passed in with Y_{text} as initial state.
2. **RNN + Basic Attention:** This attention mechanism [Vaswani et al. 2017; Bahdanau et al. 2014], represented in Figure 1c, emphasizes important triplets by computing weighted average of GRU embeddings using attention weights a_t. Attention weights are computed by applying an unbiased softmax dense layer on Y_{text} as shown:
$$a_t = softmax(W_{text} \cdot Y_{text}) \tag{1}$$
$$Y_{graph} = 1/N \cdot \sum_{n=1}^{N_{triplets}} a_t \cdot Y_{triplets} \tag{2}$$
where $Y_{triplets}$ are the GRU embeddings of triplets. Weights W_{text} are learned in training. I expect this to outperform earlier models as it should learn to apply more weights to relevant triplets.
3. **RNN+Attention on dot and general scoring function** In this approach, shown in Figure 1d and 1e respectively, alignment weights a_t are learned by comparing source and target hidden states i.e., Y_{text} and $Y_{triplets}$ as shown below:
$$a_t = align(Y_{triplets}, Y_{text}) = \frac{exp(score(Y_{triplets}, Y_{text}))}{sum(exp(score(Y_{triplets}, Y_{text})))} \tag{3}$$
Here, I tried two alternatives of score function which are dot and general attention mechanisms [Luong et al. 2015] as shown below:
   (a) **Dot Scoring**: score($Y_{triplets}, Y_{text}$) = $Y_{triplets} \cdot Y_{text}$
   (b) **General Scoring**: score($Y_{triplets}, Y_{text}$) = $Y_{text} \cdot W_a \cdot Y_{triplets}$

4.2.3 GCN Based Model for Y_{graph}

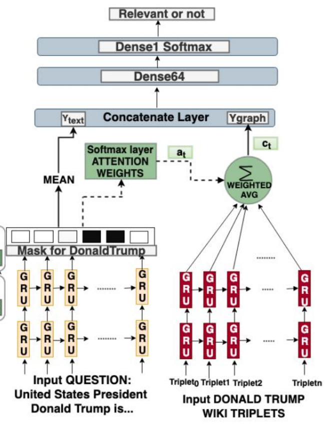
In this model, as shown in Figure 1g, using hop processing $(D^-1 \cdot (A_adjacency + IdentityMatrix))$ where D is node degree matrix in GCN [Kipf et al. 2016] was used to generate Y_{graph}. GCN not only learns representation which preserves graph topology but it also takes into account the feature description of the node. For this work, I have considered only one-hop edges of a given node and passed it into GCN to get an aggregated vector Y_{graph}.



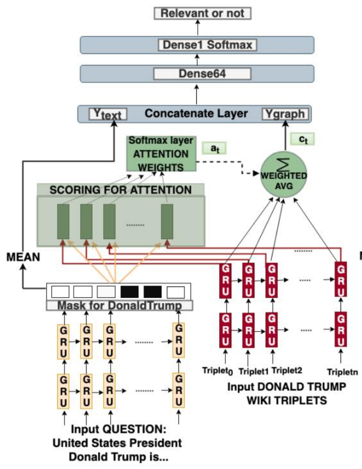
(a) RNN-GRU Based Model



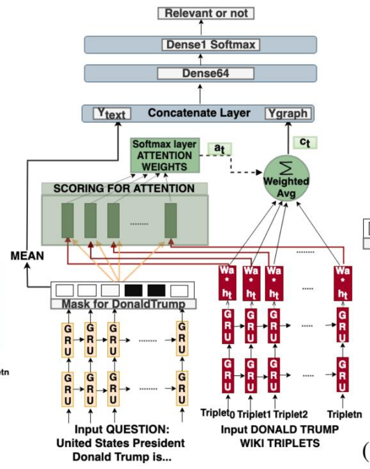
(b) RNN-BiLSTM Based Model


Relevant or not
Dense1 Softmax
Dense64
Ytext
Concatenate Layer
Ygraph
Softmax layer ATTENTION WEIGHTS
at
ct
WEIGHTED AVG
MEAN
Mask for DonaldTrump
GRU
Triplet0 Triplet1 Triplet2
Tripletn
Input QUESTION: United States President Donald Trump is...
Input DONALD TRUMP WIKI TRIPLETS

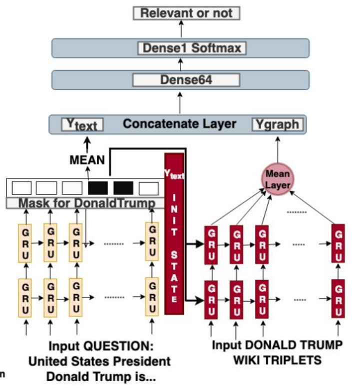
(c) Basic Attention Model


Relevant or not
Dense1 Softmax
Dense64
Ytext
Concatenate Layer
Ygraph
Softmax layer ATTENTION WEIGHTS
at
ct
WEIGHTED AVG
SCORING FOR ATTENTION
MEAN
Mask for DonaldTrump
GRU
Triplet0 Triplet1 Triplet2
Tripletn
Input QUESTION: United States President Donald Trump is...
Input DONALD TRUMP WIKI TRIPLETS

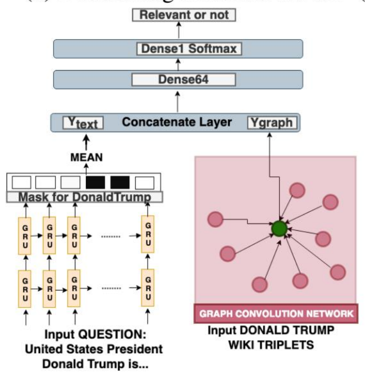
(d) Dot Scoring Attention Model


Relevant or not
Dense1 Softmax
Dense64
Ytext
Concatenate Layer
Ygraph
Softmax layer ATTENTION WEIGHTS
at
ct
Weighted Avg
SCORING FOR ATTENTION
MEAN
Mask for DonaldTrump
Wa * ht
GRU
Triplet0 Triplet1 Triplet2
Tripletn
Input QUESTION: United States President Donald Trump is...
Input DONALD TRUMP WIKI TRIPLETS

(e) General Scoring Attention Model



(f) Conditional State Attention Model


Relevant or not
Dense1 Softmax
Dense64
Ytext
Concatenate Layer
Ygraph
MEAN
Mask for DonaldTrump
GRU
GRAPH CONVOLUTION NETWORK
Input QUESTION: United States President Donald Trump is...
Input DONALD TRUMP WIKI TRIPLETS

(g) GCN Based Model

```
             precision    recall  f1-score   support

        0.0       0.90      0.89      0.89      1100
        1.0       0.89      0.90      0.90      1100
```

(h) ConfusionMatrix for Model (e)

Figure 1: Model Architectures

Experiment Results on test dataset for various models

	positive records			negative records		
Description	**prec**	**rec**	**f1**	**prec**	**rec**	**f1**
GRU for both Y_{text} and Y_{text}	0.77	0.73	0.75	0.74	0.78	0.76
GRU for Y_{text} and Bi-LSTM for Y_{text}	0.80	0.76	0.78	0.77	0.81	0.79
RNN + Conditional Attention	0.84	0.82	0.83	0.83	0.84	0.83
RNN+Basic Attention	0.80	0.81	0.80	0.81	0.80	0.80
RNN+Attention on dot scoring function	0.88	0.88	0.88	0.88	0.88	0.88
RNN+Attention on general scoring function	0.90	0.89	0.89	0.89	0.90	0.90
GCN Based Model for Y_{text}	0.83	0.85	0.84	0.85	0.82	0.84

Figure 2: Experimental Results on TestDataset with Precision, Recall and F1 Score of Positive and Negative samples given separately

5 Experiments/Results/Discussion

A total of quarter million triplets data from Wiki knowledge base is queried and downloaded (taken 10 days on single thread process to download). Training the models with Glove embedding layer needed more than 16GB ram to not have memory issues, hence entire embedding layer is done as an offline pre-process step. Furthermore, the dataset was split into files of 100 records each (single file took around 0.2GB and data set needed 0.52TB). Adam optimizer with default learning rate of 0.01, beta1 0.9, beta2 0.999, epsilon 1.0e-08 which gave 98% accuracy on a single file in 30 epochs using binary cross entropy with accuracy metric. Since the entire dataset cannot be in memory, a batch size of 5 files (500 data records) is run for 10 iterations in each epoch. All the results presented used 5 epochs (compared to 60 used by Cetoli 2018) and dev predictions confirmed no overfitting. Please find the experiment results in Figure2 above.

GRU for Y_{text} and Bi-LSTM for Y_{graph} model gave slightly better results compared to **GRU for both Y_{text} and Y_{graph}** because Bi-LSTM model without sequence output is expected to learn more (similar to learning of typical Neural Machine Translation models). Attention based models outperformed non-attention based models which is expected as RNNs beyond 30 sequence length will struggle to remember and the dataset used here has on average 250 sequence length. Moreover, results also show that **Attention with general and dot scoring functions** (expected as per discussion by Minh-thang et al. 2015) out performs **basic attention scoring** (Cetoli et al. 2018) as attention weights modulated by triplets and question makes intuitive sense compared to just attention derived just from question. Please find confusion matrix of the best performing model, which is Attention with General scoring model, on test data in Figure 1h. I am very optimistic on **GCN based triplet encoding and aggregation** as unlike above schemes it captures the graph topology very well. GCN model in this work obtained improved results compared to GCN model in (Cetoli et al. 2018) which can be attributed to improvements that are tried here such as self addition and node degree normalization (as suggested by Minh-thang et al. 2015). Although one hop GCN (based on Kipf et al. 2017) underperformed compared to general attention based approach (couldn't explore other combinations) I feel this approach combined with attention on top of GCN nodes has a lot of promise and intuitively might outperform over all discussed approaches in this work.

6 Conclusion/Future Work

For the task of NED, this paper contributes to using graph knowledge base and compares performances of basic RNN, RNN with attention and GCN architectures. Clearly, attention based models performed better than basic RNN models agreeing with (Cetoli et al. 2018) observations. However, attention with general scoring performed the best as it captures question modulated triplet focus. Although RNN with attention models gave better results than GCN-based model, note that compared to the recent literatures such as Cetoli et al 2018, who attempted GCN for NED, the GCN-model in this work has shown improved accuracy. This can be attributed to improvements such as self identity computation along with Node degree Normalization in graph convolution operation.

In future, I would like to experiment with various other GCN architectures like message-passing, attention-GCN and multi-hop GCN. I would also, like to run on the entire 2 million dataset for more accurate literature comparison.

References

[1] Minh-Thang Luong, Hieu Pham, & Christopher D. Manning. (2015) Effective Approaches to Attention-based Neural Machine Translation. *In Empirical Methods in Natural Language Processing (EMNLP).*

[2] Alberto Cetoli, Mohammad Akbari, Stefano Bragaglia, Andrew D. O'Harney & Marc Sloan. (2018) *Named Entity Disambiguation using Deep Learning on Graphs*

[3] Priya Radhakrishnan, Partha Talukdar & Vasudeva Varma (2018) *ELDEN: Improved Entity Linking using Densified Knowledge Graphs*

[4] Jeffrey Pennington, Richard Socher & Christopher D. Manning (2014) *GloVe: Global Vectors for Word Representation*

[5] Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, & Kalina Bontcheva (2016) Stance Detection with Bidirectional Conditional Encoding. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Austin, Texas, 876–885. https://aclweb.org/anthology/D16-1084

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473 [cs, stat] (Sept. 2014). http://arxiv.org/abs/1409.0473 arXiv: 1409.0473.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, & I. Polosukhin. (2017) Attention is all you need. In Neural Information Processing Systems (NIPS)

[8] Thomas N. Kipf & Max Welling. (2016) Semi-Supervised Classification with Graph Convolutional Networks. CoRR abs/1609.02907 (2016). http://arxiv.org/abs/1609.02907

[9] Nikolaos Kolitsas, Octavian Eugen Ganea & Thomas Hofmann (2018) End-to-End Neural Entity Linking. https://arxiv.org/pdf/1808.07699.pdf

[10] Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang (2013). Learning entity representation for entity disambiguation. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 30–34.

[11] Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang (2015). Modeling mention, context and entity with neural networks for entity disambiguation. In Twenty-Fourth International Joint Conference on Artificial Intelligence.

[12] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji (2016). Joint learning of the embedding of words and entities for named entity disambiguation. In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, pages 250–259.

[13] Octavian-Eugen Ganea and Thomas Hofmann (2017). Deep joint entity disambiguation with local neural attention. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2619–2629.

[14] Phong Le and Ivan Titov (2018). Improving entity linking by modeling latent relations between mentions. arXiv preprint arXiv:1804.10637.