Geometric Deep Learning Models for Linking Character Names in Novels

Marek Kubis
Adam Mickiewicz University in Poznań
Faculty of Mathematics and Computer Science
ul. Uniwersytetu Poznańskiego 4, Poznań 61-614, Poland
mkubis@amu.edu.pl

Abstract

The paper investigates the impact of using geometric deep learning models on the performance of a character name linking system. The neural models that contain graph convolutional layers are confronted with the models that include conventional fully connected layers. The evaluation is performed with respect to the perfect name boundaries obtained from the test set and in a more demanding end-to-end setting where the character name linking system is preceded by a named entity recognizer.

1 Introduction

Characters are mentioned in novels in various ways. They are referred by their first names, surnames, job titles, nick names, diminutives, honorifics and other forms that are ambiguous to some extent. This paper investigates the impact of using geometric deep learning for the purpose of linking names that refer to the same character. Geometric deep learning is a set of deep learning methods developed for non-Euclidean domains (Bronstein et al., 2017). In this preliminary study we analyze the impact of using graph convolutional networks (Kipf and Welling, 2017) and an edge based training objective on the performance of the character name linking system built upon a non-Euclidean domain of a character mentions graph.

The problem of linking all names that refer to the same character is a subject of multiple publications. Elson and McKeown (2010) perform name linking with a rule-based method as a preliminary step for attributing passages of quoted speech in narrative. The same method of linking is used by Bamman et al. (2014) for their work on modeling character types. Coll Ardanuy and Sporleder (2014) present another deterministic algorithm for character name resolution which is an integral part of their method of constructing social networks from literary artworks. Vala et al. (2015) propose a multistage pipeline for character detection that transforms the graph of character mentions by adding and removing edges between nodes that should refer to the same character. A more general problem of character identification which encompasses both name and pronoun resolution is discussed in the context of TV Shows by Chen and Choi (2016) and Choi and Chen (2018). A similar problem of character reference detection is addressed by Krug (2020) for German historic novels. A closely related task of coreference resolution was successfully approached with various deep learning models in recent years (Clark and Manning, 2016; Wiseman et al., 2016; Lee et al., 2017).

2 Dataset

For this study we use *Lalka* (English: *The Doll*) a Polish novel by B. Prus for the purpose of training and evaluation. All names that indicate the same character are annotated with a common *character identifier* which is unique within the scope of the book. The dataset is divided into training and test set that are roughly equal in size and consist of two consecutive sets of chapters. Although limited to one novel our corpus with its 150000 tokens and 192 unique characters is comparable in size to the dataset used by Choi and Chen (2018).

Proceedings of LaTeCH-CLfL 2020, pages 127–132
Barcelona, Spain (Online), December 12, 2020.

For the purpose of constructing geometric models the dataset is transformed into an adequate graphical representation of the problem. The conversion procedure has access to the entire text of the corpus and to the character identifiers of mentions that appear in the training set. We use all names that are mentioned in the text as a set of nodes. Two mentions l and r are connected by an edge, if one of the following cases holds:

1. l and r share the same lemmata *// e.g. Stanisława Wokulskiego – Stanisławem Wokulskim*
2. l is a prefix or a suffix of r *// Stanisław – Stanisław Wokulski*
3. l is a diminutive of r *// Stasiek – Stanisław*
4. l is an abbreviated form of r *// S. Wokulski – Stanisław Wokulski*
5. l and r share the same character identifier in the training set. *// Wokulski – Stanisław*

3 Models

We consider two training objectives with respect to the mentions graph that we refer to as *node* and *edge* in the rest of the paper. The first one is to predict directly the correct character identifier for every node in the graph. This objective does not require any post-processing steps, however its main disadvantage is the inability to predict identifiers for characters that are not present in the training set. The second objective is to predict the equivalence relation that holds if and only if two nodes in the mentions graph refer to the same character. This objective can be used to predict identifiers for characters that do not appear in the training set, but it requires two additional steps in order to obtain character identifiers for the nodes. First, one has to extract connected components from the graph induced by the predicted relation. Second, the character identifiers have to be determined on the basis of the mentions that belong to the same component.[1]

Figure 1 presents neural network architectures that are set up for the purpose of character name linking. All of them accept at the input vector representations of nodes that belong to the mentions graph. A single node is represented by a vector that encodes the words that belong to a mention (*word_vec*) and a vector that encodes its surrounding context (*ctx_vec*). Architectures (a) and (b) are designed for the *node* objective. They consist of a set of consecutive layers that transform node representations. The number of neurons in the last layer is equal to the number of characters in the novel. The cross-entropy loss is used for the purpose of training. Architecture (a) transforms node representations using conventional fully connected layers of neurons (*Linear* in Figure 1). It depends solely on the vector representations of nodes without taking into consideration the topology of the mentions graph. We confront this architecture with (b) where fully connected layers are replaced with dedicated graph convolutional network layers (Kipf and Welling, 2017) that utilise the edge index. This allows us to verify if there is any advantage in using structural information for the task. For the *edge* objective we develop architectures that transform node representations of (a) and (b) into representations of edges. For this purpose the *AdjacentSum* layer is introduced which returns for every edge in the mentions graph the concatenation of the corresponding node representations and their dot product, i.e.

$$e_{ij} = Concat(r_i, r_j, r_i \cdot r_j)$$

where $(i, j) \in edge\_index$ and r_i is the vector representation of node i. Furthermore, (c) and (d) are supplied with additional edge related features denoted by *edge_attr* in Figure 1. We use this vector to store textual distance between edges and to encode link types.[2] As in the case of (a) and (b) the only difference between (c) and (d) is the introduction of graph convolutional layers. However, one should notice that even though (c) does not utilise convolutions it still takes into account network topology

[1] For this task we select the most frequent character identifier in the connected component if any is present and fallback to the most frequent lemma otherwise.

[2] There are five one-hot encoded link types that correspond to the respective rules of connecting mentions defined in Section 2.

due to the formulation of the *edge* objective and the definition of *AdjacentSum*. Both *edge* objective architectures are trained with binary cross entropy loss to detect edges that connect mentions of the same character.

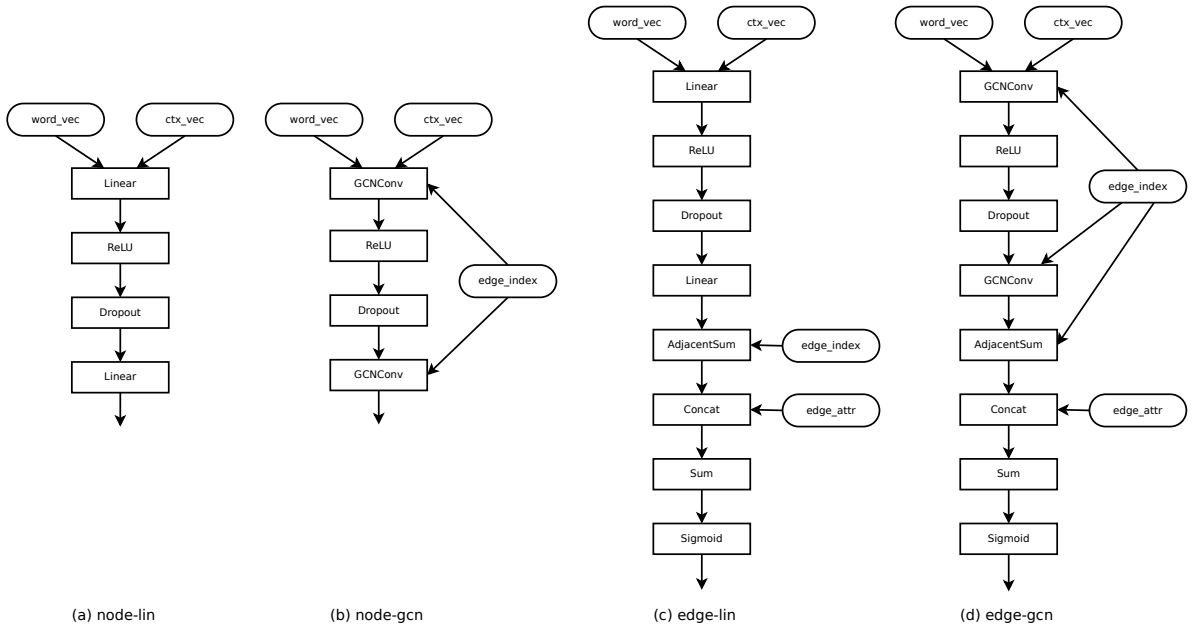

Figure 1: Neural network architectures

4 Experiments

For the purpose of evaluation we confront the neural architectures discussed in Section 3 with two baseline systems:

- *base* – a deterministic algorithm that memorizes all surface forms of character names that appear in the training set together with the corresponding character identifiers and annotates the spans of text that match the memorized surface forms with the stored character identifiers during the testing phase.[3]

- *seq* – a sequence to sequence model that learns a direct mapping from character names to their identifiers using the state-of-the-art technique of contextual string embeddings for sequence labeling (Akbik et al., 2018).

Since the graph induction procedure depends on a proper demarcation of mention boundaries we decided to conduct two types of experiments for every model. First, we use name boundaries detected by the named entity recognizer learned from the mentions that appear in the training set.[4] This is a realistic setup that we use to measure the end-to-end performance of the complete name linking system trained solely from the provided dataset without access to any additional resources. Second, we repeat the same experiment with name boundaries obtained from the test set. In this scenario the performance of the name linking model can be measured independently from the named entity recognition technique used to identify mention boundaries in the complete system. Results are reported separately for the entire set of

[3] Any ambiguities are resolving by selecting the most frequent character identifier.

[4] For named entity recognition we use a procedure that closely resembles the *base* algorithm. First, the surface forms of character names that appear in the training set are memorized. Then, the spans of text that match the memorized forms are annotated in accordance with the standard IOB scheme during the testing phase.

characters that appear in the test set and for the characters that where already *observed* in the training set. Micro F-score is the main metric used for the purpose of evaluation. Macro F-score which is a metric that penalizes the system for bad performance with respect to the rare characters is also reported.

Let us start the analysis of the results collected in Table 1 with the more demanding scenario of the character name linker preceded by the named entity recognizer. In this case the *node-lin* model presents similar performance to *seq* which is not unexpected taking into account that *node-lin* does not utilise topology of the mentions graph in any manner. However, if we restrict our attention to the characters that where observed in the training part of the corpus the differences between *seq* and *node-lin* become more apparent with the former being more performant with respect to the micro f-score and the latter with respect to the macro f-score. Both *node-lin* and *seq* do not surpass the second baseline which is a far simpler solution to the problem. The *node-gcn* model outperforms both baselines which proves that there is a clear benefit in incorporating structural information into the model. Furthermore, the *edge-lin* and *edge-gcn* models which incorporate the structure of the graph in the training objective perform better than *node-gcn*, especially with respect to the macro f-score. One should also notice than there is no observable advantage of using graph convolutional layers instead of linear layers in case of models designed for the *edge* objective. In fact, the connected components extracted from the relations predicted by *edge-lin* and *edge-gcn* are so similar that the overall scores do not change. Although the *edge* models manifest better performance over the entire set of characters, if we focus on the characters that are present in both parts of the corpus the *node-gcn* model is a clear winner. These observations remain valid for the second evaluation scenario which utilises the mention boundaries obtained from the test set instead of the boundaries detected by the named entity recognizer.

			Micro			Macro		
Model	**Boundaries**	**Characters**	**Prec.**	**Recall**	**F-score**	**Prec.**	**Recall**	**F-score**
base	ner	all	0.9384	0.7628	0.8415	0.2293	0.2174	0.2168
seq	ner	all	0.8064	0.7644	0.7848	0.1498	0.1790	0.1520
node-lin	ner	all	0.9246	0.6790	0.7830	0.1952	0.1463	0.1572
node-gcn	ner	all	0.9298	0.7849	0.8512	0.2446	0.2506	0.2454
edge-lin	ner	all	0.8751	0.8714	**0.8732**	0.3914	0.3680	**0.3729**
edge-gcn	ner	all	0.8751	0.8714	**0.8732**	0.3914	0.3680	**0.3729**
base	ner	observed	0.9388	0.9416	0.9402	0.6531	0.6191	0.6175
seq	ner	observed	0.8072	0.9377	0.8676	0.3074	0.3796	0.3160
node-lin	ner	observed	0.8521	0.8193	0.8354	0.5609	0.4188	0.4481
node-gcn	ner	observed	0.9392	0.9680	**0.9534**	0.7208	0.7341	**0.7213**
edge-lin	ner	observed	0.7766	0.9542	0.8563	0.1522	0.1516	0.1494
edge-gcn	ner	observed	0.7766	0.9542	0.8563	0.1522	0.1516	0.1494
base	test set	all	0.9768	0.7533	0.8506	0.2415	0.2193	0.2253
seq	test set	all	0.8064	0.7644	0.7848	0.1498	0.1790	0.1520
node-lin	test set	all	0.7011	0.7011	0.7011	0.1882	0.1602	0.1565
node-gcn	test set	all	0.7961	0.7961	0.7961	0.2446	0.2679	0.2437
edge-lin	test set	all	0.9226	0.9226	**0.9226**	0.6256	0.6297	**0.6195**
edge-gcn	test set	all	0.9226	0.9226	**0.9226**	0.6256	0.6297	**0.6195**
base	test set	observed	0.9981	0.9300	0.9628	0.7886	0.6849	0.7189
seq	test set	observed	0.8072	0.9377	0.8676	0.3074	0.3796	0.3160
node-lin	test set	observed	0.8634	0.8634	0.8634	0.6162	0.4930	0.5201
node-gcn	test set	observed	0.9823	0.9823	**0.9823**	0.8770	0.8921	**0.8638**
edge-lin	test set	observed	0.9758	0.9758	0.9758	0.6498	0.6272	0.6263
edge-gcn	test set	observed	0.9758	0.9758	0.9758	0.6498	0.6272	0.6263

Table 1: Character name linking performance broken down by the source of mention boundaries and the set of characters.

5 Conclusion

In this study we investigated the impact of using graph convolutional networks and the edge based training objective on the performance of the character name linking system built upon a character mentions graph. The conducted experiments show that the structural information encoded in the mentions graph improves the performance of the character name linking system significantly. Furthermore, there are several interesting observations that can be drawn from the achieved results. First, the usage of graph convolutional layers is essential in case of the *node* models in order to be able to outperform the baseline models. Second, taking into consideration that *node-gcn* is the best performing model with respect to the observed characters, it should be used in all situations where a training set that references all the characters that are important for the given study is provided. Third, there is no observable performance drop when graph convolutional layers are replaced by linear layers in the models developed for the *edge* objective. However, taking into account that the model which utilizes graph convolutional layers contains fewer parameters, it should be preferred.

Although the results presented in this preliminary study are promising, there are several issues that require further investigation. The choice of graph convolutional networks among many geometric deep learning models was an arbitrary one. Thus, the impact of other graphical neural network architectures on the performance of the name linking system should be measured in the future. The dataset used for the study, although comparable in size to the dataset used by Choi and Chen (2018), consists of only one novel. Hence, the question of how well the obtained results generalize to novels that come from different authors and periods is left open. Finally, taking into consideration the top performance of the *node-gcn* model with respect to the observed characters, a method of indicating unobserved characters for the *node* objective should be developed.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

David Bamman, Ted Underwood, and Noah A. Smith. 2014. A Bayesian Mixed Effects Model of Literary Character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 370–379, Baltimore, Maryland, June. Association for Computational Linguistics.

M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. 2017. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Yu-Hsin Chen and Jinho D. Choi. 2016. Character Identification on Multiparty Conversation: Identifying Mentions of Characters in TV Shows. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 90–100, Los Angeles, September. Association for Computational Linguistics.

Jinho D. Choi and Henry Y. Chen. 2018. SemEval 2018 Task 4: Character Identification on Multiparty Dialogues. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 57–64, New Orleans, Louisiana, June. Association for Computational Linguistics.

Kevin Clark and Christopher D. Manning. 2016. Deep Reinforcement Learning for Mention-Ranking Coreference Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262, Austin, Texas, November. Association for Computational Linguistics.

Mariona Coll Ardanuy and Caroline Sporleder. 2014. Structure-based Clustering of Novels. In *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*, pages 31–39, Gothenburg, Sweden, April. Association for Computational Linguistics.

David K. Elson and Kathleen R. McKeown. 2010. Automatic Attribution of Quoted Speech in Literary Narrative. In Maria Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press.

Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Markus Krug. 2020. *Techniques for the Automatic Extraction of Character Networks in German Historic Novels*. Ph.D. thesis, Universität Würzburg.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark, September. Association for Computational Linguistics.

Hardik Vala, David Jurgens, Andrew Piper, and Derek Ruths. 2015. Mr. Bennet, his coachman, and the Archbishop walk into a bar but only one of them gets recognized: On The Difficulty of Detecting Characters in Literary Texts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 769–774, Lisbon, Portugal, September. Association for Computational Linguistics.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. 2016. Learning Global Features for Coreference Resolution. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004, San Diego, California, June. Association for Computational Linguistics.