Knowledge Graph Entity Alignment with Graph Convolutional Networks: Lessons Learned

Max Berrendorf[1(✉)], Evgeniy Faerman[1], Valentyn Melnychuk[2], Volker Tresp[1,3], and Thomas Seidl[1]

[1] Ludwig-Maximilians-Universität München, Munich, Germany
{berrendorf,faerman,seidl}@dbs.ifi.lmu.de
[2] Fraunhofer Institute for Integrated Circuits IIS, Erlangen, Germany
v.melnychuk@campus.lmu.de
[3] Siemens AG, Munich, Germany
volker.tresp@siemens.com

Abstract. In this work, we focus on the problem of entity alignment in Knowledge Graphs (KG) and we report on our experiences when applying a Graph Convolutional Network (GCN) based model for this task. Variants of GCN are used in multiple state-of-the-art approaches and therefore it is important to understand the specifics and limitations of GCN-based models. Despite serious efforts, we were not able to fully reproduce the results from the original paper and after a thorough audit of the code provided by authors, we concluded, that their implementation is different from the architecture described in the paper. In addition, several tricks are required to make the model work and some of them are not very intuitive.We provide an extensive ablation study to quantify the effects these tricks and changes of architecture have on final performance. Furthermore, we examine current evaluation approaches and systematize available benchmark datasets.We believe that people interested in KG matching might profit from our work, as well as novices entering the field. (Code: https://github.com/Valentyn1997/kg-alignment-lessons-learned).

1 Introduction

The success of information retrieval in a given task critically depends on the quality of the underlying data. Another issue is that in many domains knowledge bases are spread across various data sources [14] and it is crucial to be able to combine information from different sources. In this work, we focus on knowledge bases in the form of Knowledge Graphs (KGs), which are particularly suited for information retrieval [17]. Joining information from different KGs is non-trivial, as there is no unified schema or vocabulary. The goal of the entity alignment task is to overcome this problem by *learning* a matching between entities in different KGs. In the typical setting some of the alignments are known in advance (seed alignments) and the task is therefore supervised. More formally, we are given

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 3–11, 2020.
https://doi.org/10.1007/978-3-030-45442-5_1

graphs $G_L = (V_L, E_L)$ and $G_R = (V_R, E_R)$ with a seed alignment $A = (l_i, r_i)_i \subseteq V_L \times V_R$. It is commonly assumed that an entity $v \in V_L$ can match at most one entity $v' \in V_R$. Thus the goal is to infer alignments for the remaining nodes only.

Graph Convolutional Networks (GCN) [7,9], which have been recently become increasingly popular, are at the core of state-of-the-art methods for entity alignments in KGs [3,6,22,24,27]. In this paper, we thoroughly analyze one of the first GCN-based entity alignment methods, GCN-Align [22]. Since the other methods we are studying can be considered as extensions of this first paper and have a similar architecture, our goal is to understand the importance of its individual components and architecture choices.In summary, our contribution is as follows:

1. We investigate the reproducibility of the published results of a recent GCN-based method for entity alignment and uncover differences between the method's description in the paper and the authors' implementation.
2. We perform an ablation study to demonstrate the individual components' contribution.
3. We apply the method to numerous additional datasets of different sizes to investigate the consistency of results across datasets.

2 Related Work

In this section, we review previous work on entity alignment for Knowledge Graphs and revisit the current evaluation process. We believe that this is useful for practitioners, since we discover some pitfalls, especially when implementing evaluation scores and selecting datasets for comparison. An overview of methods, datasets and metrics is provided in Table 1.

Table 1. Overview of related work in the field of entity alignment for knowledge graphs with their used datasets and metrics.

Method	Datasets	Metrics	Code
MTransE [5]	WK3l-15K, WK3l-120K, CN3l	H@10(, MR)	yes
IPTransE [26]	DFB-{1,2,3}	H@{1,10}, MR	yes
JAPE [18]	DBP15K(JAPE)	H@{1,10,50}, MR	yes
KDCoE [4]	WK3l-60K	H@{1,10}, MR	yes
BootEA [19]	DBP15K(JAPE), DWY100K	H@{1,10}, MRR	yes
SEA [15]	WK3l-15K, WK3l-120K	H@{1,5,10}, MRR	yes
MultiKE [25]	DWY100K	H@{1,10}, MR, MRR	yes
AttrE [20]	DBP-LGD,DBP-GEO,DBP-YAGO	H@{1,10}, MR	yes
RSN [8]	custom DBP15K, DWY100K	H@{1,10}, MRR	yes
GCN-Align [22]	DBP15K(JAPE)	H@{1,10,50}	yes
CL-GNN [24]	DBP15K(JAPE)	H@{1,10}	yes
MuGNN [3]	DBP15K(JAPE), DWY100K	H@{1,10}, MRR	yes
NAEA [27]	DBP15K(JAPE), DWY100K	H@{1,10}, MRR	no

Methods. While the problem of entity alignments in Knowledge Graphs has been tackled historically by researching vocabularies which are as broad as possible, and establish them as a standard, recent approaches take a more data-driven view. Early methods use classical knowledge graph link prediction models such as TransE [2] to embed the entities of the individual knowledge graphs using an intra-KG link prediction loss, and differ in what they do with the aligned entities. For instance, MTransE [5] learns a linear transformation between the embedding spaces of the individual graphs using an L_2-loss. BootEA [19] adopts a bootstrapping approach and iteratively labels the most likely alignments to utilize them for further training. In addition to the alignment loss, embeddings of aligned entities are swapped regularly to calibrate embedding spaces against each other. SEA [15] learns a mapping between embedding spaces in both directions and additionally adds a cycle-consistency loss. Thereby, the distance between the original embedding of an entity, and the result of translating this embedding to the opposite space and back again, is penalized. IPTransE [26] embeds both KGs into the same embedding space and uses a margin-based loss to enforce the embeddings of aligned entities to become similar. RSN [8] generates sequences using different types of random walks which can move between graphs when visiting aligned entities. The generated sequences are feed to an adapted recurrent model. JAPE [18], KDCoE [4], MultiKE [25] and AttrE [20] utilize attributes available for some entities and additional information like the names of entities and relationships. Graph Convolutional Network (GCN) based models [3,6,22,24,27][1] have in common that they use GCN to create node representations by aggregating node representations together with representations of their neighbors. Most of GCN approaches do not distinguish between different relations and either consider all neighbors equally [6,22,24] or use attention [3] to weight the representations of the neighbors for the aggregation.

Datasets. The datasets used by entity alignments methods are generally based on large-scale open-source data sources such as DBPedia [1], YAGO [13], or Wikidata [23]. While there is the DWY-100K dataset, which comprises 100 K aligned entities across the three aforementioned individual knowledge graphs, most of the datasets, such as DBP15K, or WK3l are generated from a single multi-lingual database. There, subsets are formed according to a specific language, and entities which are linked across languages are used as alignments. A detailed description of most-used datasets can be found in Table 2.

As an interesting observation we found out that all papers which evaluate on DBP15, do not evaluate on the full DBP15K dataset[2] (which we refer to as *DBP15K (full)*), but rather use a smaller subset provided by the authors of JAPE [18] in their GitHub repository[3], which we call *DBP15K (JAPE)*. The smaller subsets were created by selecting a portion of entities (around 20K of 100K) which are popular,

[1] While [27] does not state explicitly that they use GCNs, their model is very similar to [21].

[2] Available at http://ws.nju.edu.cn/jape/.

[3] https://github.com/nju-websoft/JAPE/blob/master/data/dbp15k.tar.gz.

i.e. appear in many triples as head or tail. The number of aligned entities stays the same (15K). As [18] only reports the dataset statistics of the larger dataset, and does not mention the reduction of the dataset, subsequent papers also report the statistics of the larger dataset, although experiments use the smaller variant [3,18,19,22,26]. As the metrics rely on absolute ranks, the numbers are better than on the full dataset (cf. Table 3).

Scores. It is common practice to only consider the entities being part of the test alignment as potential matching candidates. Although we argue that ignoring entities exclusive to a single graph as potential candidates does not reflect well

Table 2. Overview of used datasets with their sizes in the number of triples (edges), entities (nodes), relations (different edge types) and alignments. For WK3l, the alignment is provided as a directed mapping on a entity level. However, there are additional triple alignments. Following a common practice as e.g. [15] we can assume that an alignment should be symmetric and that we can extract entity alignments from the triple alignments. Thereby, we obtain the number of alignments given in brackets.

Dataset	Subset	Graph	Triples	Entities	Relations	Alignments
DBP15K (full)	fr-en	fr	192,191	66,858	1,379	15,000
		en	278,590	105,889	2,209	
	ja-en	ja	164,373	65,744	2,043	15,000
		en	233,319	95,680	2,096	
	zh-en	zh	153,929	66,469	2,830	15,000
		en	237,674	98,125	2,317	
DBP15K (JAPE)	fr-en	fr	105,998	19,661	903	15,000
		en	115,722	19,993	1,208	
	ja-en	ja	77,214	19,814	1,299	15,000
		en	93,484	19,780	1,153	
	zh-en	zh	70,414	19,388	1,701	15,000
		en	95,142	19,572	1,323	
WK3l-15K	en-de	en	209,041	15,127	1,841	1,289 (10,383)
		de	144,244	14,603	596	1,140 (10,383)
	en-fr	en	203,356	15,170	2,228	2,498 (8,024)
		fr	169,329	15,393	2,422	3,812 (8,024)
WK3l-120K	en-de	en	624,659	67,650	2,393	6,173 (50,280)
		de	389,554	61,942	861	4,820 (50,280)
	en-fr	en	1,375,406	119,749	3,109	36,749 (87,836)
		fr	760,497	118,592	2,336	36,013 (87,836)
DWY-100K	dbp-wd	dbp	463,294	100,000	330	100,000
		wd	448,774	100,000	220	
	dbp-yg	dbp	428,952	100,000	302	100,000
		yg	502,563	100,000	31	

the use-case situation[4], we follow this evaluation scheme for our experiments to maintain comparability.

3 Method

GCN-Align [22] is a GCN-based approach to embed all entities from both graphs into a common embedding space. Each entity i is associated with *structural* features $h_i \in \mathbb{R}^d$, which are initialized randomly and updated during training. The features of all entities in a single graph are combined to the feature matrix H. Subsequently, a two-layer GCN is applied. A single GCN layer is described by $H^{(i+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(i)}W^{(i)}\right)$ with $\hat{A} = A + I$, where A is the adjacency matrix, and $\hat{D}_{ii} = \sum_{j=1}^{n} \hat{A}_{ij}$ is the diagonal node degree matrix. The input of the first layer is set to $H^{(0)} = H$, and σ is non-linear activation function, chosen as ReLU. The output of the last layer is considered as the structural representation, denoted by $s_i = H_i^{(2)} \in \mathbb{R}^d$. Both graphs are equipped with their own node features, but the convolution weights $W^{(i)}$ are shared across the graphs.

The adjacency matrix is derived from the knowledge graph by first computing a score, called *functionality*, for each relation as the ratio between the number of different entities which occur as head, and the number of triples in which the relation occurs α_r. Analogously, the *inverse functionality* α'_r is obtained by replacing the nominator by the number of different tail entities. The final adjacency matrix is obtained as $A_{ij} = \sum_{(e_i,r,e_j)} \alpha'_r + \sum_{(e_j,r,e_j)} \alpha_r$. Note, that analogously to structural features GCN-Align is able to process the attributes and integrate them in final representation. However, since attributes have little effect on final score, and to be consistent with other GNN models, here we focus only on structural representations.

Implementation Specifics. The code[5] provided by the authors differs in a few aspects from the method described in the paper. First, when computing the adjacency matrix, $fun(r)$ and $ifun(f)$ are set to at least 0.3. S, the node embeddings are initialized with values drawn from a normal distribution with variance $n^{-1/2}$, where n is the number of nodes[6]. Additionally, the node features are always normalised to unit Euclidean length before passing them into the network. Finally, there are no convolution weights. This means that the whole GCN does not contain a single parameter, but is just a fixed function on the learned node embeddings.

[4] In the typical scenario it is not known in advance, which entities have matching and which not. Therefore the resulting score is too optimistic. We advocate to investigate this shortcoming further in future work.

[5] https://github.com/1049451037/GCN-Align.

[6] We could not find any justification for that in literature.

4 Experiments

In initial experiments we were able to reproduce the results reported in the paper using the implementation provided by the authors. Moreover, we are able to reproduce the results using our own implementation, and settings adjusted to the authors' code. In addition, we replaced the adjacency matrix based on functionality and inverse functionality by a simpler version, where $a_{ij} = \{(h, r, t) \in T \mid h = e_i, t = e_j\}$. We additionally use $\hat{D}^{-1}\hat{A}$ instead of the symmetric normalization. In total, we see no difference in performance between our simplified adjacency matrix, and the authors' one. We identified two aspects which affect the model's performance: Not using convolutional weights, and normalizing the variance when initializing node embeddings. We provide empirical evidence for this finding across numerous datasets. Our results regarding Hits@1 (H@1) are summarised in Table 3.

Table 3. Ablation study on using convolution weights and different embedding initialisation.We fix using convolution weights and the variance for the normal distribution from which the embedding vectors are initialized and optimize the other hyperparameters according to validation H@1 (80/20% train-validation split) on *DBP15K (JAPE) zh-en* in a large-scale hyperparameter search, comprising 1,440 experiments, with the following grid: optim. $\in$ {Adam, SGD}, lr $\in \{0.1, 0.5, 1, 10, 20\}$, #layers $\in \{1, 2, 3\}$, #neg. samples $\in \{5, 50, 100\}$, #epochs $\in \{10, 500, 2000, 3000\}$. Hence, we obtain four sets of hyperparameters. For each dataset, we perform a smaller hyperparameter search to fine-tune LR, #epochs & #layers for each dataset (again 80/20 split) and evaluate the best models on the official test set with standard deviation computed across 5 runs.

Weights		No		Yes	
Variance. Emb. Init.		1	$n^{-1/2}$	1	$n^{-1/2}$
DBP15K (full)	fr-en	**31.51 ± 0.16**	27.64 ± 0.22	21.82 ± 0.39	16.73 ± 0.59
	ja-en	**33.26 ± 0.10**	29.06 ± 0.23	26.21 ± 0.33	20.78 ± 0.16
	zh-en	**31.15 ± 0.15**	22.55 ± 0.27	24.96 ± 0.71	18.85 ± 0.99
DBP15K (JAPE)	fr-en	**45.37 ± 0.13**	41.03 ± 0.13	35.36 ± 0.33	30.50 ± 0.38
	ja-en	**45.53 ± 0.18**	40.29 ± 0.09	35.81 ± 0.53	31.46 ± 0.15
	zh-en	**43.30 ± 0.12**	39.37 ± 0.20	33.61 ± 0.49	29.94 ± 0.35
DWY100K	wd	**58.50 ± 0.05**	54.07 ± 0.05	50.13 ± 0.11	38.85 ± 0.31
	yg	**72.82 ± 0.06**	67.06 ± 0.03	67.36 ± 0.10	60.67 ± 0.30
WK3l-120K	en-de	**10.10 ± 0.03**	9.17 ± 0.05	9.02 ± 0.17	6.75 ± 0.12
	en-fr	**8.28 ± 0.03**	7.38 ± 0.03	7.26 ± 0.11	5.07 ± 0.16
WK3l-15K	en-de	16.57 ± 0.12	14.41 ± 0.23	**17.43 ± 0.38**	12.66 ± 0.30
	en-fr	**17.07 ± 0.15**	16.16 ± 0.16	15.98 ± 0.16	12.41 ± 0.18

Node Embedding Initialization. Comparing the columns of Table 3 we can observe the influence of the node embedding initialization. Using the settings from the authors' code, i.e. not using weights, a choosing a variance of $n^{-1/2}$ actually results in inferior performance in terms of H@1, as compared to use a standard normal distribution. These findings are consistent across datasets.

Convolution Weights. The first column of Table 3 corresponds to the weight usage and initialization settings used in the code for GCN-Align. We achieve slightly better results than published in [22], which we attribute to a more exhaustive parameter search. Interestingly, all best configurations use Adam optimizer instead of SGD. Adding convolution weights degrades the performance across all datasets and subsets thereof but one as witnessed by comparing the first two columns with the last two columns.

5 Conclusion

In this work, we reported our experiences when implementing the Knowledge Graph alignment method GCN-Align. We pointed at important differences between the model described in the paper and the actual implementation and quantified their effects in the ablation study. For future work, we plan to include other methods for entity alignments in our framework.

Acknowledgements. This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A and by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology through the Center for Analytics-Data-Applications (ADA-Center) within the framework of "BAYERN DIGITAL II". The authors of this work take full responsibilities for its content.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
2. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a Meeting, Lake Tahoe, Nevada, United States, 5–8 December 2013. pp. 2787–2795 (2013). http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data
3. Cao, Y., Liu, Z., Li, C., Liu, Z., Li, J., Chua, T.: Multi-channel graph neural network for entity alignment. In: Korhonen et al. [10], pp. 1452–1461. https://www.aclweb.org/anthology/P19-1140/

4. Chen, M., Tian, Y., Chang, K., Skiena, S., Zaniolo, C.: Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In: Lang [12], pp. 3998–4004. https://doi.org/10.24963/ijcai.2018/556
5. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: Sierra [16], pp. 1511–1517. https://doi.org/10.24963/ijcai.2017/209
6. Fey, M., Lenssen, J.E., Morris, C., Masci, J., Kriege, N.M.: Deep graph matching consensus. In: International Conference on Learning Representations (2020). https://openreview.net/forum?id=HyeJf1HKvS
7. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 1263–1272. JMLR. org (2017)
8. Guo, L., Sun, Z., Hu, W.: Learning to exploit long-term relational dependencies in knowledge graphs. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning (ICML 2019), Long Beach, California, USA, 9–15 June 2019. Proceedings of Machine Learning Research, vol. 97, pp. 2505–2514. PMLR (2019). http://proceedings.mlr.press/v97/guo19c.html
9. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
10. Korhonen, A., Traum, D.R., Màrquez, L. (eds.): Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL 2019), Florence, Italy, 28 July – 2 August 2019, Volume 1: Long Papers. Association for Computational Linguistics (2019). https://www.aclweb.org/anthology/volumes/P19-1/
11. Kraus, S. (ed.): Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019), Macao, China, 10–16 August 2019. ijcai.org (2019). https://doi.org/10.24963/ijcai.2019
12. Lang, J. (ed.): Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI 2018), Stockholm, Sweden, 13–19 July 2018. ijcai.org (2018). http://www.ijcai.org/proceedings/2018/
13. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: a knowledge base from multilingual wikipedias. In: Seventh Biennial Conference on Innovative Data Systems Research (CIDR 2015), Asilomar, CA, USA, 4–7 January 2015, Online Proceedings. www.cidrdb.org (2015). http://cidrdb.org/cidr2015/Papers/CIDR15_Paper1.pdf
14. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proc. IEEE **104**(1), 11–33 (2015)
15. Pei, S., Yu, L., Hoehndorf, R., Zhang, X.: Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In: Liu, L., et al. (eds.) The World Wide Web Conference (WWW 2019), San Francisco, CA, USA, 13–17 May 2019, pp. 3130–3136. ACM (2019). https://doi.org/10.1145/3308558.3313646
16. Sierra, C. (ed.): Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI 2017), Melbourne, Australia, 19–25 August 2017. ijcai.org (2017). http://www.ijcai.org/Proceedings/2017/
17. Singhal, A.: Introducing the knowledge graph: things, not strings. Official Google Blog **5**, (2012)
18. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10587, pp. 628–644. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68288-4_37

19. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: Lang [12], pp. 4396–4402. https://doi.org/10.24963/ijcai.2018/611
20. Trisedya, B.D., Qi, J., Zhang, R.: Entity alignment between knowledge graphs using attribute embeddings. In: The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019), The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI 2019), Honolulu, Hawaii, USA, 27 January – 1 February 2019, pp. 297–304. AAAI Press (2019). https://aaai.org/ojs/index.php/AAAI/article/view/3798
21. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
22. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October – 4 November 2018, pp. 349–357. Association for Computational Linguistics (2018). https://www.aclweb.org/anthology/D18-1032/
23. Wikidata. https://www.wikidata.org/
24. Xu, K., et al.: Cross-lingual knowledge graph alignment via graph matching neural network. In: Korhonen et al. [10], pp. 3156–3161 (2019). https://www.aclweb.org/anthology/P19-1304/. arXiv preprint arXiv:1905.11605
25. Zhang, Q., Sun, Z., Hu, W., Chen, M., Guo, L., Qu, Y.: Multi-view knowledge graph embedding for entity alignment. In: Kraus [11], pp. 5429–5435. https://doi.org/10.24963/ijcai.2019/754
26. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via joint knowledge embeddings. In: Sierra [17], pp. 4258–4264. https://doi.org/10.24963/ijcai.2017/595
27. Zhu, Q., Zhou, X., Wu, J., Tan, J., Guo, L.: Neighborhood-aware attentional representation for multilingual knowledge graphs. In: Kraus [11], pp. 1943–1949. https://doi.org/10.24963/ijcai.2019/269