Recent Advances in Entity Resolution

Bing Li[1], Yaoshu Wang[2], and Wei Wang[3]

[1] UNSW, Sydney, Australia
[2] Shenzhen Institute of Computing Sciences, Shenzhen, China
yaoshuw@sics.ac.cn
[3] Hong Kong University of Science and Technology (Guangzhou), China
weiwcs@ust.hk

1 Tutorial's subject and relevance to ER

Entity resolution (ER) (a.k.a., entity matching, record linkage, and duplicate record detection) aims at identifying records that refer to the same real-world entity from different data sources. As a fundamental task for data cleaning and data integration [1], entity resolution has been widely applied in knowledge graph construction [2], e-Commerce [3], etc. Since its inception, it has been extensively studied by means of various methodologies such as declarative rules [4], crowdsourcing [5], and machine learning [6]. Over the past few years, deep learning (DL) has fuelled fast-paced advances in many established fields such as CV, NLP, as well as for data management. This trend also brings new opportunities and challenges to the ER problem. Many DL-based ER models [7, 8] have emerged to tackle this long-standing problem.

This tutorial aims at providing a focused and multi-faceted review of recent advances for entity resolution, especially the DL-based ER solutions. As a typical AI for DB problem, the studies of DL-based ER solution attracted widely attentions from both AI and DB communities. The core of the interplay is how to model the table schema and training pipeline. In this sense, it is not only interesting to see how DL techniques are applied to a typical DB problem, but also how it in turn changes the way DB, as its origin, dealing with the problem. We first discuss the importance of entity resolution in data cleaning and data integration, and then review the recent DL-based entity matching models regarding different schema modeling, *i.e.*, schema-agnostic, hard-schema, and soft-schema. We analyze their strengths and weaknesses. Moreover, we show two types of ER pipeline, blocker and matcher pipleline and joint-learning paradigm. We wish this tutorial could be an impetus towards more AI for DB applications.

2 Target audience, prerequisite knowledge, and learning goals

Target audience: Anyone who is interested in recent advances in data or schema integration, and data science.

Prerequisite knowledge: The audience is expected to have some basic understanding of data management and machine/deep learning. Nevertheless, we

will cover the necessary technical details in the Background and Preliminaries section of the tutorial.

Learning goals: The audience will be updated with recent progress of entity resolution techniques from both DB and AI communities.

3 Tutorial contents and intended structure

The contents mainly consists of five parts. The first part introduces the necessary background and preliminaries. The second to fourth parts delve into schema-agnostic, hard-schema, and soft-schema solutions, respectively. The fifth part introduces ER pipeline under current DL fashion. The sixth part concludes the tutorial and discusses potential future directions.

3.1 Background and Preliminaries

In the introductory part of the tutorial, we first introduce the concept of entity resolution and explain its importance in data clearing and integration. Then we introduce its basics: (1) the definition of schema, tuple, and entity resolution; (2) a typical ER pipeline: a blocker and a matcher; (3) a brief introduction of deep learning techniques, including attention mechanism, LSTM, Transformer, BERT, etc. (4) a summary of the solutions that will be elaborated in the rest of the tutorial.

3.2 Schema-agnostic Solutions

Schema-agnostic solutions do not use schema information or presume the schema is agnostic. In this setting, each tuple is treated as a plain text sequence, which is often known as the unstructured text sequence matching problem that has been extensively studied, especially in the NLP community. Thus, most schema-agnostic solutions are directly borrowed from text matching and QA matching models of the NLP community, with major differences in the chosen of specific neural networks, *e.g.*, CNN [9] and Transformer [10]. These models often produce inferior results outside textual tables as being discarded useful schema information.

3.3 Hard-schema Solutions

Hard-schema ER solutions explicitly exploit schema information in *both* representation and comparison phases. In representation learning, the tuple embeddings are schematically generated and organized, *i.e.*, aggregate tokens' embeddings to attributes', and aggregate attributes' embeddings to records'. There are a verity of aggregation functions, *e.g.*, simple averaging [11] , RNN [11] , and LSTM [11, 7]. In the comparison phase, compare each tuple pair schematically (*e.g.*, in an attribute-wise manner) using similarity functions such as cosine [11] or attention-based similarity [7]. The comparison results are then fed into a

classifier (*e.g.*, SVM [11] or MLP [7]) for the final ER decision. Hard-schema representation and comparison are too rigorous to flexibly integrate information scattered across different attributes (*e.g.*, misplaced attributes problem). Further, it requires tables' schema to be identical (*i.e.*, mediate schema), which has to incur an error-prone and cumbersome schema-mapping step, which hinders their applications on datasets manifesting high schema heterogeneity.

3.4 Soft-schema Solutions

The soft-schema solutions integrate schema information into representation learning, besides that, the schema is not entailed in the subsequent steps. Typically, the schema information are softly encoded into tuple embeddings using various techniques, *e.g.*, GraphER [12] uses GCN to encode record-attribute-token hierarchy. In order to be compatible with the current pre-trained LMs fashion (*e.g.*, BERT, RoBERTa, ALBERT), BERT-ER [13] adds extra table and attribute encodings on the embedding layer. Ditto [14] and EMBER [15] use special tokens (*e.g.*, [Att: Num]) as add-ons to feed into BERT. Soft-scheme solutions provide more flexibilities that could be further utilized by the powerful deep pre-trained LMs, these solutions empirically achieve significantly higher effectiveness than the aforementioned ones.

3.5 ER Pipeline

In order to reduce the quadratic searching space of entity matching, an ER pipeline often contains a blocking module. Blocking is a predominant speed-up techniques to group potentially co-referent tuples into blocks such that the fine-grained matching are exclusively performed within blocks.

Blocker-matcher Pipeline Blocker-matcher pipeline is the most traditional ER pipeline, it complies with the *filter-and-refine* paradigm. Blocking and matching are regarded as two isolated processes and there is no interaction between them. As being amenable to the distributed representations (*i.e.*, embedding), hashing-based solutions and nearest neighbor (NN) solutions becomes the de-facto standard for DL-based ER solutions. Typical hashing-based solutions include locality sensitive hashing (LSH) [11, 16], and learning to hash [13]. Hashing approaches pursue a high efficiency. To meet the need for a high recall, deep nearest neighbor blocker are proposed, *e.g.*, Ditto [14] pick k-NN candidates with a BERT-based similarity.

Joint-learning Paradigm BERT-ER [13] proposes a joint-learning paradigm that jointly learns blocker and matcher in a multi-task learning framework. In this way, the blocker could be aware of the matching features from a shared BERT encoder, and the model is able to utilize the results of matching to further rectify the blocking-incurred error. This solution is high effective in facilitating the performances of both blocking and learning tasks.

3.6 Future Opportunities

We highlight some potential directions for future research: (1) it is interesting to build pre-trained models on relational table data using self-supervision. (2) Exploit the information in dirty and erroneous schema or semi-schema would be of high practical meaning. (3) Another direction is to explore efficient entity resolution techniques on massive table collections.

Bibliography

[1] Xin Luna Dong and Divesh Srivastava. Big data integration. In *2013 IEEE 29th ICDE*, pages 1245–1248. IEEE, 2013.

[2] Yun-Nung Chen, William Yang Wang, Anatole Gershman, and Alexander Rudnicky. Matrix factorization with knowledge graph propagation for unsupervised spoken language understanding. In *Proceedings of the 53rd Annual Meeting of the ACL*, pages 483–494, 2015.

[3] Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. Corleone: hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 601–612, 2014.

[4] Mauricio A Hernández and Salvatore J Stolfo. The merge/purge problem for large databases. In *ACM Sigmod Record*, volume 24, pages 127–138. ACM, 1995.

[5] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB*, 5(11):1483–1494, 2012.

[6] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, 2015.

[7] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. In *Proceedings of the 2018 ACM SIGMOD*, pages 19–34. ACM, 2018.

[8] Cheng Fu, Xianpei Han, Le Sun, Bo Chen, Wei Zhang, Suhui Wu, and Hao Kong. End-to-end multi-perspective matching for entity resolution. In *Proceedings of the 2019 IJCAI*, pages 4961–4967. AAAI Press, 2019.

[9] Shuohang Wang and Jing Jiang. A compare-aggregate model for matching text sequences. In *ICLR*, 2017.

[10] Ursin Brunner and Kurt Stockinger. Entity matching with transformer architectures-a step forward in data integration. In *International Conference on Extending Database Technology, Copenhagen, 30 March-2 April 2020*. OpenProceedings, 2020.

[11] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. Distributed representations of tuples for entity resolution. *Proceedings of the VLDB*, 11(11):1454–1467, 2018.

[12] Bing Li, Wei Wang, Yifang Sun, Linhan Zhang, Muhammad Asif Ali, and Yi Wang. Grapher: Token-centric entity resolution with graph convolutional neural networks. In *AAAI*, pages 8172–8179, 2020.

[13] Bing Li, Yukai Miao, Yaoshu Wang, Yifang Sun, and Wei Wang. Improving the efficiency and effectiveness for bert-based entity resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13226–13233, 2021.

[14] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment*, 14(1):50–60, 2020.

[15] Sahaana Suri, Ihab F Ilyas, Christopher Ré, and Theodoros Rekatsinas. Ember: No-code context enrichment via similarity-based keyless joins. *arXiv preprint arXiv:2106.01501*, 2021.

[16] Wei Zhang, Hao Wei, Bunyamin Sisman, Xin Luna Dong, Christos Faloutsos, and Davd Page. Autoblock: A hands-off blocking framework for entity matching. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 744–752, 2020.