Entity Recognition and Linkage for Reference Data

Nagaraj Bahubali Asundi , **Sriram Aralappa** , **Adarsh Anand** ,
Soniya Manchenahalli Gnanendra Prasad and **Abinaya Thulsi Chandrasekaran**
University of Koblenz-Landau
{nagarajbahubali, sriramaralappa, adarshanand, soniya, abinayathulsi}@uni-koblenz.de

Abstract

Entity mentions such as author names in research paper citations are inherently ambiguous due to name variations. Such name ambiguity together with incorrect author names can cause erroneous document retrieval from the bibliographic networks and eventually lead to improper attribution to authors. To overcome this issue, we propose a novel supervised deep learning model - Reference2Auth. Our model captures the relational context of an author by extracting his co-author patterns and semantic features of other citation attributes such as journal and title to perform collective disambiguation. We have used around 2K citation strings from the DBLP dataset to train our model. We were able to link each author name present in the reference strings to the respective original author entity in DBLP network with the test accuracy of around 99%.

1 Introduction

The increase in growth of scientific publications has led to widespread of digital libraries (DLs) such as DBLP[1], CiteSeerX[2], Google Scholar[3], Microsoft Academic[4], etc. Such libraries may contain millions of 'citation' records and serve as a critical source of information for academic communities. The term 'citation' refers to the collection of bibliographic information such as author name, title, journal, or year of publication that are pertinent to a particular scientific paper. As DLs are usually built having a target community of users with specific interests [21], it is often necessary to maintain the *quality* of their bibliographic content [22]. However, there are challenges to maintain the quality of such content due to discrepancies originating from two different sources: (1) Citing authors (2) DL systems.

In the first type, discrepancies arise when an author's publication or the paper is cited by several other authors inconsistently in the reference section of their paper. These inconsistencies occur due to the usage of different citation styles such as ASA[5], APA[6], IEEE[7], etc., by different authors. i.e., when citing a paper, the cited author's name may appear in various forms such as its full name, a partial name with initials, abbreviations, etc. For instance, the author 'Wei Shen' has been referred to in multiple surface forms like 'Shen W' or 'W Shen' in different publications [26, 27]. There is also a certain degree of human error, such as typos in the names of cited authors. Typos can happen when the citations are manually included in the reference section of the paper.

In the second type, the challenges to have quality bibliographic content arises due to data entry errors, ambiguous author names, abbreviations of journal names and disparate citation formats [23]. Among these challenges, author name ambiguity is one of the prominent issues that has gained attention from the research community [24, 25, 6]. To be specific, author name ambiguity exists when the same author name may appear in different name variations (synonyms), and distinct authors may share the same name (homonyms) [12]. For example, 'Zeyd Boukhers' and 'Z Boukhers' both refer to the same author 'Dr Zeyd Boukhers' from the University of Koblenz and Landau (synonyms). On the contrary, 'Hao Chen' can either refer to Prof. 'Hao Chen' from Boise State University, USA or Prof. 'Hao Chen' from Hunan University, China (homonyms).

Among these issues, as mentioned above, we are limiting our focus mainly on author name ambiguity and incorrect author names as a result of typos or spelling errors. To illustrate the problem, Table 1 shows a set of reference strings covering each issue. Here each author is given an identifier ranging from a_1 to a_7. In the first reference string, author a_2 can be uniquely identified, whereas a_1 can refer to any of the four 'Bing Li's available in the DBLP network due to homonymy. Subsequently, authors a_2 and a_5 refer to the same author 'Weihua Xiong' but appear in different name variations due to different citation styles (synonymy). Finally, in the third reference string, author a_6 refers to 'Christophe Pon-

[1] https://dblp.org/
[2] http://citeseerx.ist.psu.edu/
[3] https://scholar.google.com/
[4] https://academic.microsoft.com/home
[5] https://www.citethisforme.com/citation-generator/asa
[6] https://libguides.murdoch.edu.au/APA
[7] https://libguides.murdoch.edu.au/IEEE/entries

ID	Issue type	Citation styles	Citation or reference strings
1	Homonymy	ASA	Bing Li(a_1), Weihua Xiong(a_2). 2012. Visual saliency map from tensor analysis. In Proceedings of the 26th AAAI Conference on Artificial Intelligence, pages 1585-1591
2	Synonymy	IEEE	H. Peng(a_3), B. Li(a_4),W. Xiong(a_5), Predicting image memorability by multi-view adaptive regression, Proc. 23rd ACM Int. Conf. Multimedia, pp. 1147-1150, 2015.
3	Incorrect author name	APA	Pinsard, C.(a_6), Ramdoyal, R.(a_7) : An ocr-enabled digital comic books viewer. In: Computers Helping People with Special Needs, pp. 471–478. Springer, (2012)

Table 1. An illustrative example of author name ambiguity and incorrect author names

sard', but the last name is misspelt as Pinsard due to human error.

Author name ambiguity and inaccuracy can affect the quality of scientific data gathering and consequently leads to incorrect identification and credit attribution to authors. Hence it is vital to map the author names in the citations to the correct author in the DLs.

Author name mapping can be regarded as one of the cases of the broad notion of entity linking. Entity linking, also known as Named Entity Disambiguation (NED), is a process of mapping an entity mention with the help of the surrounding contextual information to its corresponding unique entity in the Knowledge Base (KB). In our case, the process of entity linking can be defined more formally as : Let R = r_1,r_2,...,r_m be a set of reference (citation) strings, A = a_1,a_2,...,a_n be a set of author name mentions and E = e_1,e_2,...,e_o be a set of unique author entities in DBLP. The objective is to map each entity mention (author name) $a_j \in A$ in the reference $r_i \in R$ to corresponding unique entity $e_k \in E$ (correct author) in KB (DBLP database). Figure 1 illustrates this task of author name mapping.

To implement the task of entity linking in citations, we propose Reference2Auth, a supervised deep learning model that maps each author names in the reference string to the corresponding unique author in the DBLP network. The task is achieved with the assistance from the contextual citation attributes such as co-author names, journal name and title of the paper. We transform each of these contextual attributes to the corresponding embeddings depending on their type. Further, these embeddings are collated and supplied as an input to the model. Finally, the model takes this input and links each author name in the reference string to the respective author entity present in the DBLP network.

2 Related Work

Named Entity Linking (NEL), also known as Named Entity Disambiguation (NED), is considered as an essential sub-task to link entity mentions in the text to their corresponding entities in KB [1]. A typical entity linking system consists of three modules, namely Candidate Entity Generation, Candidate Entity Ranking, and Unlinkable Mention Prediction [2]. With the help of Candidate Entity Generation module, the NEL aims to retrieve the target entities by filtering the irrelevant entities in KB. In Candidate Entity Ranking, the entities are ranked to find the most suitable entity for the mention. Later the Unlinkable Mention Prediction module will evaluate if the top-ranked entity is the targeted entity.

Cucerzan in [4] looks into computing semantic relatedness of documents which is Explicit Semantic Analysis (ESA). Here, the goal is to find a pair of articles that contain similar words and compute their relatedness score from the word-based similarity of the articles. Therefore, using this approach is advantageous because they are simple and easy to implement. However, their performance is low compared to supervised methods. Much recent work in this domain has focused on the cosine similarity measure or ranking techniques between the entities. As [4] focuses only on contextual similarities between the entities, Xianpei et al. [5] aims to improve this traditional entity linking method [4]. Hence, they propose a graph-based collective entity linking approach that can model and exploit the global interdependence, i.e., a mutual dependence between the entities.

Due to name abbreviations, misspellings and similar names in publications, an author can be represented using multiple names, and various authors may share the same name. Han et al. [6] presents two supervised learning approaches to disambiguate authors in the citations. Firstly, using the naive Bayes probability model, a generative statistical model to capture all writing patterns in the author citations. Therefore, Among the candidate entities, one that has the highest probability is matched to the corresponding entity in the knowledge base. Secondly, using Support Vector Machines (SVMs) [7], a discriminative model. This paper proposes the idea of a canonical name. Such a word may have more than just the name of the individual for representation. Momeni et al. in [12] presents a similar approach that tackles the author name homonym problem, i.e., when different authors share identical names. [12] uses author grouping method to disambiguate author names by categorizing all the publications belonging to the same ambiguous author into one block. Ferreira et al.[13] employs a similar approach that uses clustering on the citation records of the authors. Author grouping methods, in general, apply a similarity function along with some clustering technique to

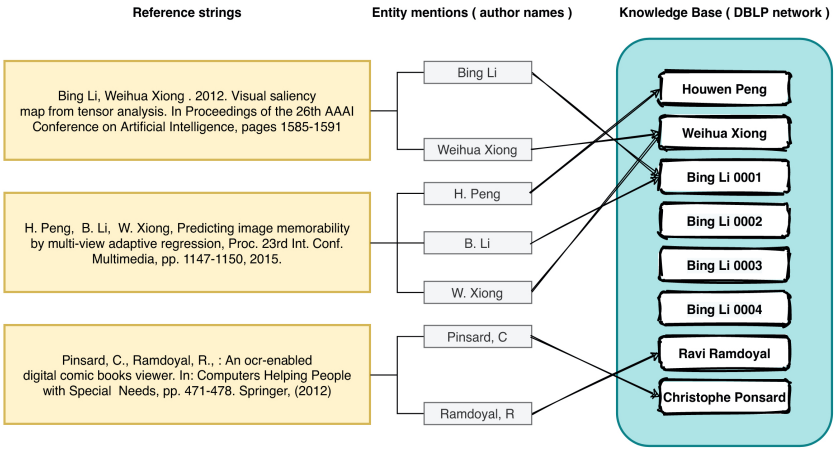

Fig. 1. An illustration for the task of linking an entity mention in the reference string with the DBLP bibliographic network

group references that belong to the same author to maximize intra-cluster similarities and minimize inter-cluster similarity.

In recent years, the area of deep learning is heavily explored by several scholars, often coming up with models that outperform the more traditional feature-based approaches for Entity Recognition and Linkage. A broad description of the most recent neural entity linking systems is discussed in the survey conducted by Sevgili et al. [14].

Hourrane et al.[15] proposes a corpus-based approach that uses deep learning word embeddings to compute the citation similarities. Muhammad Ebraheem et al. [16] proposed a novel Entity Resolution system called the DEEPER. It uses a combination of bi-directional recurrent neural networks along with Long Short Term Memory (LSTM) as the hidden units to generate a distributed representation (vector) for each tuple to capture the similarities between the tuples. Ganguly et al. in [17] proposes Author2Vec, a model that uses deep learning to overcome the link sparsity problem faced by models like DeepWalk [18]. Author2Vec uses dedicated models for content (textual data) and link information to be used in parallel. By combining these two models, Author2Vec intends to learn the author embeddings for a bibliographic network in an unsupervised manner.

Hoffart et al.[19] presents a robust method for collective disambiguation of author names. It harnesses the context from a knowledge base and uses a new form of coherence graph. The system generates a weighted graph of the candidate entities and mentions to compute a dense subgraph that approximates the best entity-mention mapping. Jonathan Raiman in [20] proposes DeepType, a multilingual Entity Linking by Neural Type system Evolution. It has been applied on the datasets WikiDisamb30, CoNLL (YAGO), TAC KBP 2010) with highly encouraging results. The algorithm involves two steps: The first step involves performing a heuristic search or stochastic optimization over the discrete variables. It is then followed by gradient descent to fit the classifier parameters.

3 Data Preparation

3.1 Data Collection

We have used the DBLP bibliographic repository[8] as a source of data to train our model. We have taken the dump of this repository in the form of an XML document. As of July 2020, this document contains 4.4 million records (references), such as conference papers, articles, thesis, etc., from various fields of research. Each reference illustrates metadata information of a paper with one or more authors, title, journal, year of publication and a few other attributes. Also the authors in DBLP who share the same name have a suffix number to differentiate them. For instance, the authors with the same name 'Bing Li' are given suffixes such as 'Bing Li 0001', 'Bing Li 0002'.

The number of attributes appearing in each reference string is not fixed and is varying in nature from one reference to the other. We had to store this massive number of references onto a database which makes it relatively easy to query and obtain the top authors based on the number of papers published. However, it is hard to store these references onto a relational database due to their diversified nature. Therefore we have used Mongo[9] database to handle the schemaless references. We identified the top 40 authors and queried the Mongo database to fetch around 2k references consisting of these authors.

3.2 Feature Generation

As each reference string may have one or more authors, it is not possible to feed input samples of variable length to our model. To overcome this issue, we restrict the number of authors in a sample to two. One acts as an author and the other as co-author. Thus, we duplicate each reference string into multiple samples by supplying all possible author and co-author name combinations but keeping the title and journal

[8] https://dblp.uni-trier.de/xml/
[9] https://www.mongodb.com/

same. To account possible citation styles for author names, we further enrich the input sample by including all name variations for author and co-author. Therefore, each input sample is a tuple consisting of an author name variant, a co-author name variant, a title and a journal. Label for each such sample is the original author from DBLP corresponding to the author name variant.

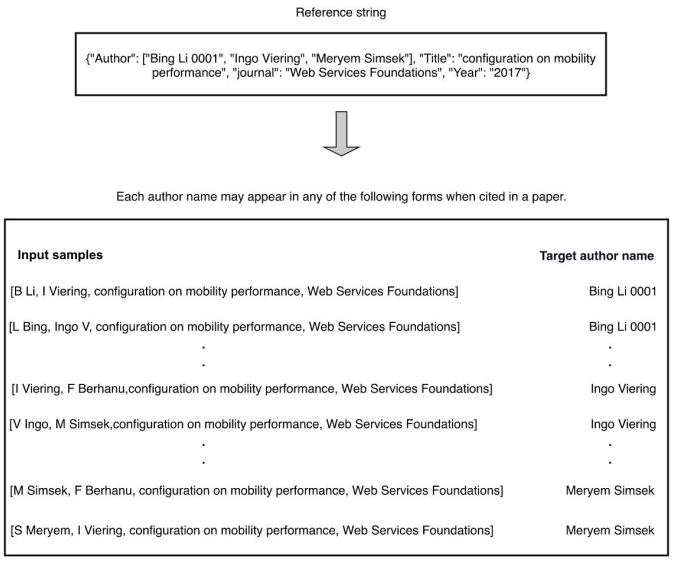

Fig. 2. Generation of input samples for Reference2Auth

Figure 2 illustrates the generation of multiple samples from a reference string to capture possible name variants of authors. With this strategy, we generated 133K synthetic references from the original dataset of 2K references to train our model.

As we cannot supply the reference strings directly in text format to the model, there is a need for converting these references to some form of numeric representations. Generating word embeddings is the widely used approach in most of the NLP tasks to process the text. Here, each word is converted to its numeric vector form before feeding to any ML model. Few of the prominent language models to generate such embeddings are Chars2vec[10], Word2vec[11], ELMo[12], GloVe[13], etc.

Each of the models mentioned above has its own set of advantages and limitations depending upon the characteristics of the text. Hence, the selection of such models has to be done cautiously. For example, the citation attributes such as author name and title belong to distinct categories of text. Author name merely represents a person name and does not carry any semantic characteristics to it. In comparison, a title can be attributed to semantics. Therefore we cannot employ the same embedding strategy for both.

During the phase of feature generation, we transform citation attributes such as author, title, journal in a reference string into vector representations or embeddings based on their specific nature. We further collate the embeddings of individual citation attribute to form a unified reference embedding. Eventually, the collated reference embedding is utilized for training the model. Label for each such collated reference embedding is the sparse categorical value of the target author (i.e., index of the target author in the unique list of original authors)

Author name embedding

As stated before, author names do not hold any specific semantic nature to them since they are simply a specific sequence of characters. Besides language models such as Word2vec or GloVe essentially work on a finite set of vocabulary and fail to interpret words that are out of vocabulary. We require a model that can encode the words based on the order and distribution of characters such that authors with similar name spellings are encoded similarly.

Chars2vec is a language model that is preferred when the text consists of abbreviations, typos, etc. It captures the non - vocabulary words and places words with similar spelling closer in the vector space. This model uses a fixed list of characters for word vectorization, where a one-hot encoding represents each character.

A pair of similar or dissimilar words are passed to this model, where the sequence of one-hot vectors of each letter in words are passed through two layers of LSTM and to an extended neural network. This model strives to place similar words nearer in vector space by reducing the distance between the words in similar pairs. Whereas, it tries to increase the distance between words in dissimilar pairs and place them far from each other. In order to obtain the embedding of author names, we used Chars2vec model, which is implemented with the help of Keras based on TensorFlow. When citing a paper, we noticed that the representation of an author name could vary due to different citation styles. So, we manually generated the standard citation styles for each author name. Our aim here is to place the different citation styles of an author close to each other in the vector space. For this, we trained the Chars2vec model with positive (similar pair) and negative (dissimilar pair) samples. A positive sample is a pair of different citation styles of an author with a label 0. In contrast, negative samples are generated as a pair of different citation styles of different authors with a label 1. For example, (Umur Karabulut, U Karabulut) forms a positive pair as they represent the same person whereas (Umur Karabulut, Ingo Viering) forms a negative pair. After training the model, we get enriched embeddings of dimension 200 for author names.

Journal embedding

Name of the journal can provide a hint about the area of research or domain to which published papers relate. Subsequently, this information can be used to enhance the encoding of an author to reflect his area of research. In the DBLP network, we noticed that a significant number of reference strings contain a journal name that is abbreviated. To ensure that the essence of journal names is captured, we fetched the expanded journal names using the DBLP venue API[14].

[10]https://github.com/IntuitionEngineeringTeam/chars2vec

[11]https://arxiv.org/pdf/1301.3781.pdf

[12]https://arxiv.org/pdf/1802.05365.pdf

[13]https://nlp.stanford.edu/pubs/glove.pdf

[14]https://dblp.org/search/venue/api

The venue API returns data in an HTML format upon which we used the beautiful soup[15] to retrieve the journal names. Currently, the total number of unique journals present in the DBLP network is 13,101. Instead of invoking the venue API on demand, we stored the complete set of expanded journal names with their abbreviations as keys into a dictionary data structure. This reduces the time complexity as multiple API calls can hinder the performance of our model. Once we have the journal names, the next task is to generate embeddings for the same. As the journal names appear as sentences, we performed sentence embedding twice. We once used the BERT[16] pre-trained model and another using Word2vec pre-trained model to generate two versions of embedding for the complete DBLP journal set.

Title embedding

Titles of references are always meaningful sentences and infer the domain of the scientific paper. This feature can be used to understand the research area of the author. Similar to journals, we used BERT and Word2vec pre-trained models to generate sentence embedding for the titles.

4 Methodology

4.1 Architecture

Our Reference2Auth model is a sequential model which is a two-layer, fully connected deep neural network with 128 neurons in each hidden layers. Size of the input layer is either 1936 or 1000 depending on the language model used to generate the embeddings of journal and title. Size of the output layer is 1523, which corresponds to the number of unique author entities. The model aims at linking the entity mentions (author names) in a given reference string to one of the target entities (original author) in the output layer. Figure 3 briefly illustrates the architecture of our Reference2Auth model, linking author name mention in the input reference to its unique entity (Hao Chen 0002).

4.2 Implementation

We have implemented two versions of the model. In the first version, BERT model is used to generate the embeddings for title and journal. In the second version, Word2vec is used to generate the embeddings for the same title and journal. However, in both the approaches, Chars2vec is used to generate the embeddings for author and co-author name variants.

Chars2vec model assigns embeddings of dimension 200 for the input author name. BERT model assigns embeddings of dimension 768 for the input title or journal. Likewise, Word2vec model assigns embeddings of dimension 768 for the input title or journal. So, in the first version with BERT, we are feeding the input sample of size 1936 [author name variant-200, co-author name variant-200, title-768, journal-768] to train our model whereas, in the second version with Word2vec, the size of the input sample is 1000 [author name variant-200, co-author name variant-200, title-300, journal-300]. The output layer of our model is the unique list of target authors. The index of the target author is assigned as a label for each input sample and sent for training.

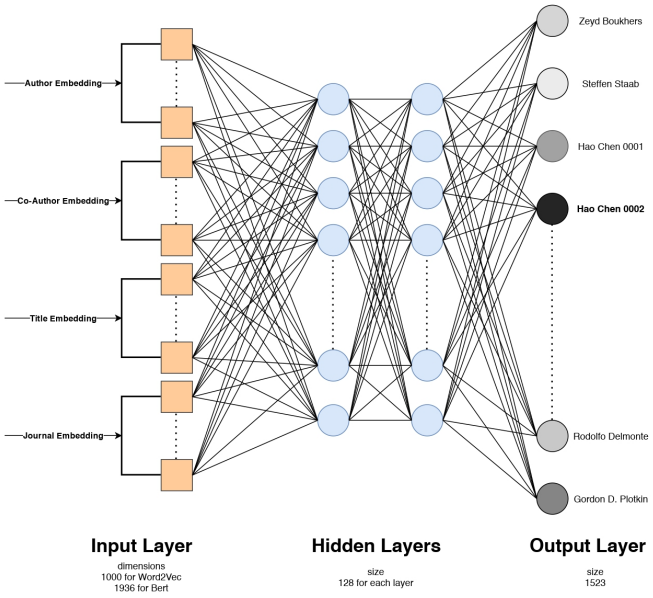

Fig. 3. Architecture of Reference2Auth model

4.3 Model Tuning

Early stopping

For each epoch, the Reference2Auth model fine-tunes the parameters to predict the appropriate target author. Performance of the model is considerably influenced by the number of epochs set to train. There are several concerns associated with hard-coding the epoch count within the model. A low epoch count may lead to underfitting. Whereas, a high epoch count may lead to overfitting. To avoid this, we enabled early stopping, which allows us to specify an arbitrary large number for the epoch. Keras supports early stopping of the training via a callback called *EarlyStopping*. This callback is configured with the help of the *monitor* argument. The monitor allows us to set the performance measure that has to be monitored. In our case, *monitor* is set to monitor the validation loss. With this setup, the model receives a trigger to halt the training when it observes no more improvement in the validation loss.

Often, the very first indication of no more improvement in the validation loss would not be the right epoch to stop training; because the model may start improving again after passing through a few more epochs. We overcome this issue by adding a delay to the trigger in terms of consecutive epochs count on which we can wait to observe no more improvement. A delay is added by setting the *patience* argument to an appropriate value. In our case, *patience* is set to 50, so that the model only halts when the validation loss stops getting better for the past 50 consecutive epochs.

Model checkpoint

Although we stop training our model when the achieved validation loss is minimum, the model obtained at the end of the training may not give the best accuracy on validation data. To account this, Keras provides an additional callback called

[15]https://www.crummy.com/software/BeautifulSoup/

[16]https://arxiv.org/pdf/1810.04805.pdf

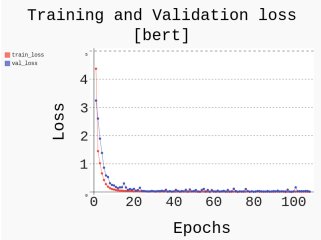

Fig. 4. Loss [BERT]

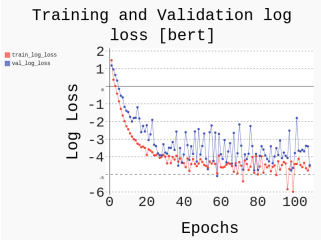

Fig. 5. Log loss [BERT]

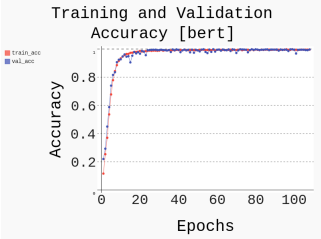

Fig. 6. Accuracy [BERT]

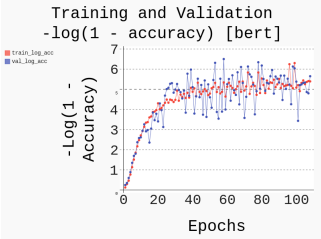

Fig. 7. Log accuracy [BERT]

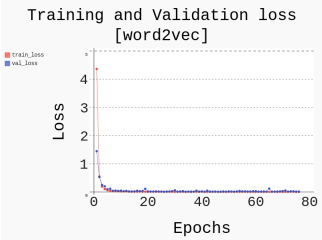

Fig. 8. Loss [Word2Vec]

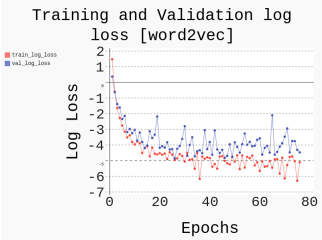

Fig. 9. Log loss [Word2Vec]

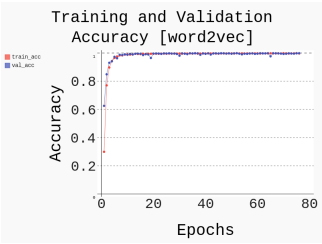

Fig. 10. Accuracy [Word2Vec]

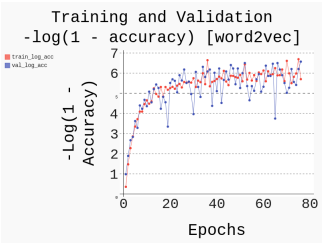

Fig. 11. Log accuracy [Word2Vec]

ModelCheckpoint. This callback is configured with the help of another *monitor* argument. In this case, we have set the *monitor* to monitor the validation accuracy. With this setup, the model updates the weights only when it observes better validation accuracy compared to earlier epochs. Eventually, we end up persisting the best state of the model with respect to the best validation accuracy.

5 Results

We evaluated our model on train, validation and test data with a split ratio of 70% : 15% : 15%. Training and validation loss is monitored continuously over the optimal number of epochs.

Figure 4 represents the training and validation loss of the model; when the BERT generated title and journal embeddings are used. Apart from the first epoch, we noticed that initially, the validation loss is significantly higher than the training loss. As the number of epochs increase, the validation loss gradually approaches 0, catching up with training loss in the process. The model stops at 108th epoch (due to early stopping) as the performance of the model does not improve any further over the past 50 consecutive epochs (patience). Figure 5 is a logarithmic projection of figure 4, which makes the loss curve more human-readable. As shown in figures 6 and 7, the model achieves an accuracy of 99.65% and 99.85% in validation and test phases, respectively.

We noticed a similar trend when the model is trained on title and journal embeddings produced from Word2vec. Figures 8 and 9 depicts the training and validation loss. Here, the loss steadily decreases and reaches its minimum at 76th epoch. With this setup, our model achieved an accuracy of 99.86% and 99.87% in validation and test phases, respectively. Table 2 summarises accuracy of the model in different phases using BERT and Word2vec.

Phase	**BERT**	**Word2vec**
Validation	99.65%	99.86%
Test	99.85%	99.87%

Table 2. Accuracy results

6 Conclusion

In this paper, we briefly outlined the challenges associated with entity linking for author names in citation strings. Further, we proposed a supervised deep learning model - Reference2Auth, which leverages additional citation attributes such as co-authors, title and journal to perform collective disambiguation. The experimental results have shown that Reference2Auth performs more accurate linking compared to the baselines.

Despite achieving high accuracy, there are a couple of avenues for improving and enhancing the Reference2Auth model. First, we extract the semantic features of the paper with the help of its title to understand the author's domain of research. However, in a real-world scenario, an author may not reuse a certain group of semantically similar title words. This makes the model harder to predict, when a title with different set of words is encountered. Therefore there is a need for developing an efficient method to determine the author's areas of research interest by mining domain-specific keywords from the entire paper instead of its title. Second, the model can perform the linking task over only 1523 authors since it is trained on a limited set of citation strings. Finally, the model does not address the dynamic nature of bibliographic repositories. Since these repositories continuously get updated, the learned model needs to be trained on the new set of data to deal with new incoming authors and publications.

References

[1] S.Singh, "Natural Language Processing for Information Extraction", in Macquarie University, Australia, 2018

[2] W. Shen,J. Wang, J. Han, "Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions",National Key Basic Research Program, China

[3] K. Humphreys, R. Gaizauskas, S. Azzam, C. Huyck, B. Mitchell, H. Cunningham, and Y. Wilks, "University of Sheffield: Description of the lasie-ii system as used for muc-7," in MUC-7, 1998

[4] Cucerzan, S., 2007. Large-scale named entity disambiguation based on Wikipedia data.In: Proceedings of EMNLP-CoNLL

[5] Han, X., Sun, L. and Zhao J. 2011. Collective Entity Linking in Web Text: A Graph-Based Method. In: Proceedings of 34th Annual ACM SIGIR Conference

[6] Han, H., Giles, L., Zha, H., Li, C. and Tsioutsiouliklis, K. Two supervised learning approaches for name disambiguation in author citations. In JCDL 2004.

[7] Vladimir N. Vapnik. 1995. The nature of statistical learning theory. Springer-Verlag, Berlin, Heidelberg.

[8] Biryukov, Maria. (2008). Co-author Network Analysis in DBLP: Classifying Personal Names. 14. 399-408.

[9] Smeaton, A.F., Keogh, G., Gurrin, C., McDonald, K., Sødring, T.: Analysis of papers from twenty-five years of SIGIR conferences: what have we been doing for the last quarter of a century? SIGIR Forum37(1) (2003)

[10] Zhang, H., Qiu, B., Lee, C.G., Foley, H.C., Yen, J.: An LDA-based CommunityStructure Discovery Approach for Large-Scale Social Networks. In: IEEE InternationalConference on Intelligence and Security Informatics(2007)

[11] Zhang, H., Qiu, B., Lee, C.G., Foley, H.C., Yen, J.: Probabilistic Community Discovery Using Hierarchical Latent Gaussian Mixture Model. In: Proceedings of the Twenty-SecondAAAI Conference on Artificial Intelligence, pp. 663–668 (2007)

[12] Fakhri Momeni and Philipp Mayr. Evaluating Co-authorship Networks in Author Name Disambiguation for Common Names. In TPDL '16: 20th Int. Conf. on Theory and Practice of Digital Libraries, Hannover, Germany, volume 9819 of LNCS, Springer, 2016

[13] Ferreira, A. A., Veloso, A., Gon¸calves, M. A., Laender, A. H., 2010. Effective self-training author name disambiguation in scholarly digital libraries. In: Proceedings of the 10th annual joint conference on Digital libraries. ACM, pp. 39–48.

[14] Sevgili, O., Shelmanov, A., Arkhipov, M., Panchenko, A., Biemann, C.: Neural Entity Linking: A Survey of Models based on Deep Learning. arXiv:2006.00575 [cs] (May 2020), http://arxiv.org/abs/2006.00575, arXiv: 2006.00575.

[15] Hourrane, Oumaima Mifrah, Sara Benlahmar, EL Habib Bouhriz, Nadia Rachdi, Mohamed. (2018). Using Deep Learning Word Embeddings for Citations Similarity in Academic Papers.

[16] M. Ebraheem, S. Thirumuruganathan, S. R. Joty, M. Ouzzani, and N. Tang. Distributed representations of tuples for entity resolution. PVLDB, 11(11):1454–1467, 2018.

[17] Ganesh, J., Ganguly, S., Gupta, M., Varma, V., and Pudi, V. (2016). "Author2vec: Learning author representations by combining content and link information," in Proceedings of the 25th International Conference on World Wide Web (WWW 2016), eds J. Bourdeau, J. Hendler, R. Nkambou, I. Horrocks, and B. Y. Zhao (Montreal, QC: ACM), 49–50

[18] Nowell, D.L., Kleinberg, J.: The link-prediction problem for social networks. In: Journal of the American Society for Information Science and Technology. (2007) 1019-1031.

[19] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Furstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 782–792. Association for Computational Linguistics.

[20] Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In Association for the Advancement of Artificial Intelligence.

[21] Gonçalves, M.A., Fox, E.A., Watson, L.T. and Kipp, N.A. Streams, structures, spaces, scenarios, societies (5S): A formal model for digital libraries. ACM Trans. Inf. Syst. 22, 2 (2004): 270-312.

[22] A. H. F. Laender, M. A. Gonçalves, R. G. Cota, A. A. Ferreira, R. L. T. Santos, and A. J. C. Silva. Keeping a digital library clean: new solutions to old problems. In DocEng, pages 257–262, 2008.

[23] D. Lee, J. Kang, P. Mitra, C. L. Giles, and B.-W. On. Are your citations clean? Comm. ACM, 50(12):33–38, 2007.

[24] Anderson A. Ferreira1, Marcos André Gonçalves, Alberto H. F. Laender. A Brief Survey of Automatic Methods for Author Name Disambiguation Acm Sigmod Record,41(2), 15-26. 2012

[25] H. Han, W. Xu, H. Zha, and C. L. Giles. A hierarchical naive Bayes mixture model for name disambiguation in author citations. In SAC, pages 1065–1069, 2005.

[26] Liang W, Li X, He X, Liu X, Zhang X (2018) Supervised ranking framework for relationship prediction in heterogeneous information networks.Appl Intell 48(5):1111–1127

[27] Jialong Han, Aixin Sun, Gao Cong, Wayne Xin Zhao, Zongcheng Ji, and Minh C Phan. 2018. Linking Fine-Grained Locations in User Comments. IEEE Transactions on Knowledge and Data Engineering30, 1(2018), 59-72.]