Exploring metadata matching for reference linking

Anonymous NAACL submission

Abstract

Reference linking, or the identification of the paper in a database that is cited by a given reference, is an important part of academic publishing. In this work, we explored reference linking based on the lexical and semantic similarity in the metadata of references and candidate papers. Our experiments affirmed the strong accuracy of Jaccard similarity reported by prior work (lowest percentage error of 0.95%) but also highlighted its poor inference speed (0.88–1.89 s per query reference, depending on the amount of metadata used). In contrast, semantic similarity-based linking achieves about twice the error rate (1.90%) while being 94 times faster (0.02 s per query reference). We recommend that future reference linking efforts employ a mixed approach of first using the coarser but faster semantic similarity-based linking, and then, only if no candidate achieves a high semantic similarity score, resorting to the slower but more accurate Jaccard-based lexical linking.

1 Introduction

Reference lists are critical components of academic writing that inform readers of relevant and influential works (Tkaczyk, 2018a). However, linking each reference with the exact paper being cited (see Figure 1) is a non-trivial task, with Liang et al. (2021) finding that the percentage of S2ORC (Lo et al., 2020) papers for which all references to PubMed are correctly linked is only 4–7%. Incorrect reference linking causes more than just inconvenience to readers; it also impairs our ability to reliably compute quantitative measures of research value and importance, such as the h-index and journal impact factor, and thus academic career prospects and perception of journal prestige (Aksnes, 2006; McKiernan et al., 2019). The exponential expansion of the literature (Fortunato et al., 2018) is likely to only make linking references correctly and quickly even more challenging.

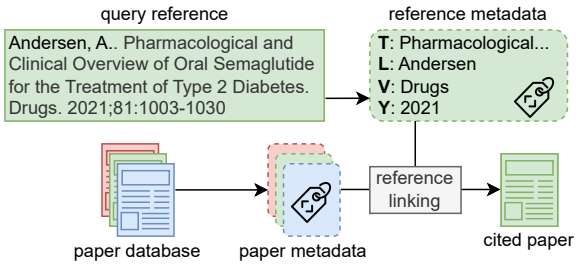

Figure 1: Illustration of reference linking. The metadata fields listed are cited paper **T**itle, author **L**ast names, publication **V**enue name, and publication **Y**ear.

In this work, we examine the task of linking references with the papers being cited. We explore various linking approaches that use lexical and semantic similarity of reference and paper metadata, and compare them on linking accuracy and inference speed. Our experiments show that semantic linking makes for a strong first-pass approach because of its speed and decent accuracy, whereas lexical linking is more suitable as a fine-grained fallback because it requires more time.

2 Dataset

The datasets made available by the prior works involve either very few references (up to 2K) or very few papers (less than 20K) (Tkaczyk, 2018a,b, 2019; Ghavimi et al., 2019; Lo et al., 2020). Therefore, for more meaningful performance evaluations, we curated a custom dataset.

2.1 Dataset construction

All data for this work was derived from the PubMed Central Open Access (PMCOA) subset, one of the largest repositories for publications in the biomedical and life science domains. We downloaded all 5.38 million PMCOA papers as XML files in bulk[1] and used a custom parser[2] to represent the content

[1] From ncbi.nlm.nih.gov/pmc/tools/ftp on 2023-06-18.
[2] github.com/titipata/pubmed_parser (MIT license).

of each paper in a structured JSON format. We then removed papers that had very short titles or that were not in English[3]; for details, see Appendix A. The 5.27 million remaining papers formed our set $\mathcal{P}$ of reference linking candidates.

Next, we looked within the papers in $\mathcal{P}$ for references x to PMCOA papers. Many references lacked the PMC identifier (PMCID) of the ground-truth cited paper P_x, but had the DOI or PubMed identifier (PMID). In such cases, we deduced the PMCID of P_x by constructing the mapping between DOI, PMID, and PMCID. Finally, to avoid overlap with the training data for the semantic similarity-based approaches (see Appendix B), we ignored all references x which cited papers P_x published before 2022. This led to a test set of 326 thousand references.

2.2 Metadata-based representative texts

For each sample x (resp. candidate P), we constructed a representative text r_x (resp. r_P), exploring two different variants:

Raw. We formed the raw r_x by copy-pasting raw references, i.e. by concatenating (with a single space) all the metadata of x in the corresponding original XML file. For each candidate P, we designed r_P to be very similar to the typical reference entry by concatenating (with a single space) the authors' **L**ast names, the **T**itle, the publication **Y**ear, and the publication **V**enue name.

Modes. To contrast against **Raw**, we also explored using only selected metadata fields. As in Lo et al. (2020), we considered **T**itle information an essential part of all representative texts. We supplemented **T**itles with the four most common metadata fields: publication **V**enue name, publication **Y**ear, author **L**ast names[4], and **A**bstract (see Table 1). All unavailable metadata fields were represented by an empty string. For examples, see Appendix C.

We defined the mode in terms of the initial letters of the used metadata field(s) – for instance, mode **TV** indicates that **T**itle and **V**enue were used.

3 Methods

Based on the representative texts r_x and r_P we computed a similarity score $s(r_x, r_P)$. We then performed a nearest neighbour search and linked x to the candidate paper with the highest similarity score. The next subsections introduce the two forms of similarity we explored, namely, *lexical* similarity and *semantic* similarity.

[3] github.com/pemistahl/lingua-py (Apache 2.0 license).

[4] All detected last names were concatenated with a single space to form a single string.

	T	**V**	**Y**	**L**	**A**
Candidates	100.0	100.0	100.0	96.5	87.8
Samples	99.2	99.9	99.3	90.4	0.0

Table 1: Percentage availability of **T**itle, publication **V**enue name, publication **Y**ear information, author **L**ast names, and **A**bstract among candidates and samples. As might be expected, no references had an abstract.

3.1 Linking with lexical similarity

We considered two forms of lexical similarity:

TitleMatch. We looked for exact matches in pre-processed titles, and, in cases of multiple matches, we randomly selected the paper for linking.

Jaccard. This refers to the Jaccard index-based lexical similarity used by Lo et al. (2020) when constructing S2ORC, a massive corpus of scientific papers. The similarity score $s(r_x, r_P)$ is computed as the harmonic mean of the Jaccard index J and a containment index C:

$$s(r_x, r_P) = \frac{2 \times J \times C}{J + C}.$$

The indexes J and C are based on the representative texts r_x and r_P:

$$J = \frac{|\mathcal{N}_x \cap \mathcal{N}_P|}{|\mathcal{N}_x \cup \mathcal{N}_P|} \quad \text{and} \quad C = \frac{|\mathcal{N}_x \cap \mathcal{N}_P|}{\min\left(|\mathcal{N}_x|, |\mathcal{N}_P|\right)},$$

where $\mathcal{N}_x$ and $\mathcal{N}_P$ respectively denote the sets of trigrams on the character level extracted from the representative texts r_x and r_P. Unlike Lo et al. (2020), to increase recall, we chose to not use any threshold for linking.

We were unable to test the lexical-based methods explored by Tkaczyk (2018a,b, 2019) because they made use of CrossRef's APIs and were thus applicable only within CrossRef's databases, not within our custom dataset.

3.2 Linking with semantic similarity

This approach involved using pre-trained text encoders to compute latent embeddings v_x and v_P from the representative texts r_x and r_P, normalising all embeddings with the L2 norm, then performing a nearest neighbour search with cosine similarity as a proxy for semantic similarity.

We experimented with the following four pre-trained text encoders:

Sent2vec. Sent2vec (Pagliardini et al., 2018) was trained with a simple, unsupervised objective to produce distributed representations for general domain texts.[5]

SBERT. SBERT (Reimers and Gurevych, 2019) was developed by using siamese and triplet networks with BERT (Devlin et al., 2019) to produce semantically meaningful embeddings for general domain text.[6]

HAtten. HAtten (Gu et al., 2022) is an encoder for scientific texts that was trained on local citation recommendation, i.e. for finding appropriate papers to cite in a given sentence.[7]

SciNCL. SciNCL (Ostendorff et al., 2022) is similar to HAtten, but was trained with a more nuanced citation graph embedding-based contrastive learning objective.[8]

4 Experiments

We conducted all experiments under the inference setting (i.e. linking one sample at a time) on a single RTX 3090 GPU and over two random test subsets (seeds 1, 2), each containing 20 thousand samples. To reduce the over-representation of highly-cited papers, we ensured that each ground-truth cited paper appeared at most once per test subset.

For all modes and all approaches, the metadata texts were truncated (see Appendix D), lowercased, then stripped of excess white spaces and all punctuation. For lexical approaches only, we replaced special characters with a single space.

Because certain experiments were very time-consuming (see Section 5.2), we selected the modes to use with each model greedily. This entailed combining the best two modes involving two metadata fields to form a three-field mode, then the next best two-field mode to form a four-field mode, and so on, until accuracy stopped improving. For simplicity, we also always used the same mode for samples and candidates.

Below, we outline how the selected metadata texts were provided as input for each approach:

- **Jaccard, Sent2vec**: Concatenation with " ".
- **SBERT**: Concatenation with `[SEP]`.
- **HAtten, SciNCL**: Concatenation with `[SEP]` for mode **TA** and with " " for all other modes.

Note that we were unable to run Jaccard with mode **TA** due to memory constraints.

We were unable to find the code used by Lo et al. (2020) for reference linking with Jaccard similarity, so we implemented it on our own and optimised it with GPU acceleration (see Appendix E). For the nearest neighbour search required by the semantic linking approaches, we used the algorithm designed by Gu et al. (2022) because we found it to be faster than FAISS (Johnson et al., 2019).

5 Results and discussion

We assessed performance on two fronts. The first is *percentage error*, i.e. the percentage of samples for which the linked candidate was not the ground-truth cited paper. The second is *inference time*, i.e. the time required per sample x to construct the representative text r_x, compute the similarity score against all candidates, and find the top-scoring candidate. The time taken to process candidates was excluded because it could be performed in advance.

5.1 Jaccard linking is more accurate

TitleMatch performed notably poorly (27.08% error) in our tests (see Table 2). We expected the errors to be caused by the randomness involved with resolving multiple matches; however, only 0.22% of the wrongly-linked samples even had matching titles with the ground-truth cited papers. This suggested that the lexical differences between titles in references and of cited papers were often so major that they could not be overcome with our preprocessing, justifying developing more flexible methods for reference linking.

Jaccard-based lexical linking achieved the lowest error of just 0.95% with **Raw** representatives, whereas the lowest percentage error with semantic linking was 1.90%, achieved by SBERT with mode **TVL**. The superior performance of Jaccard linking may be due to representative texts being concatenations of individual pieces of text, which makes them less coherent and less like the naturally-occurring text corpora used to pre-train semantic encoders. Note that the semantic methods were

[5] `sent2vec_wiki_unigrams` (BSD license).

[6] huggingface.co/sentence-transformers/all-MiniLM-L6-v2 (Apache 2.0 license).

[7] We used HAtten trained on arXiv (MIT license).

[8] huggingface.co/malteos/scincl (MIT license).

	Raw	T	TV	TY	TL	TA	TLY	TVL	TVY	TVLY
TitleMatch	–	**27.07**	–	–	–	–	–	–	–	–
Jaccard	**0.95**	1.43	1.24	1.40	1.08	–	–	–	1.25	–
Sent2Vec	7.41	2.81	3.40	**2.27**	4.95	54.02	–	–	–	–
SBERT	2.16	2.62	2.25	2.46	2.27	35.12	–	**1.90**	–	1.92
HAtten	37.82	**4.44**	19.89	4.75	8.74	79.28	–	–	–	–
SciNCL	5.32	3.08	3.29	2.86	2.78	46.77	**2.71**	–	–	–

Table 2: Average percentage error (lower is better) on two test subsets. The best score per row is in bold and the best overall is underlined.

evaluated zero-shot (i.e. without fine-tuning to reference linking specifically), which explains their poorer performance.

For all approaches, accuracy (100-percentage error) did not always improve with the number of metadata fields used. This was particularly evident in **Raw** references leading to worse accuracies for most approaches than when only partial metadata was used. This justifies the importance of parsing metadata when reference linking.

All semantic linking approaches also performed very poorly with mode **TA**. This was unsurprising because the **A**bstract component for all samples was an empty string whereas almost all candidates had **A**bstracts (see Table 1), resulting in the representative texts for samples and for candidates being extremely dissimilar.

5.2 Semantic linking is much faster

TitleMatch was the fastest of all approaches, performing each inference almost instantaneously, but we disregarded it as a viable approach because of its poor accuracy.

The four semantic reference linking approaches required very little time per sample regardless of the mode, with SBERT needing just 0.02 seconds. In contrast, the Jaccard lexical approach took much longer, ranging from 0.52 seconds with mode **T** to 1.89 seconds with mode **Raw**. It follows that SBERT with mode **TVL** achieves a 94 times higher throughput than Jaccard with **Raw**. Because papers can have many references to be linked – the average paper in our dataset had 41 references – we find semantic linking to be much more feasible in application than Jaccard-based linking.

5.3 Suggested framework

Based on our experiments, we recommend primarily using semantic-based reference linking because of its high accuracy and inference speed, and resorting to Jaccard-based lexical linking for references whose semantic link achieved a similarity score lower than some threshold t. An appropriate threshold value can be determined by using a precision-recall curve; for SBERT with mode **TVL**, we recommend $t = 0.7959$ (see Appendix F).

6 Related works

Reference linking originated from the different task of *citation matching* (Hitchcock et al., 1997; McCallum et al., 2000; Pasula et al., 2002), which aims to group references that cite the same paper but does not require identifying the paper being cited. Reference linking is also similar to the task of *local citation recommendation* (Gu et al., 2022), in which appropriate papers to cite are recommended based on a query sentence, except references are not natural language sentences.

Most prior works on reference linking have relied on *lexical* similarity in metadata as measured by term frequency-based metrics (Lawrence et al., 1999; Foufoulas et al., 2017; Lo et al., 2020) and edit distances (Tkaczyk, 2018b, 2019), with some works defining different lexical similarity measures for each metadata field (Fedoryszak et al., 2013) and training support vector machines to classify candidate links (Ghavimi et al., 2019). In contrast, our work explored the viability of *semantic* similarity under the zero-shot setting and without any handcrafted heuristics.

7 Conclusion

Reference linking is a surprisingly non-trivial task where the straightforward approaches based on lexical similarity are either not accurate or not fast. In contrast, semantic similarity-based linking is a promising approach that balances speed and accuracy. We encourage people who perform reference linking to first do extensive exploration to understand which metadata fields are most present within their dataset before deciding on the mode and linking method to use.

8 Limitations

The primary limitations of our study include:

- We did not explore additional metadata fields, such as affiliation information, due to its absence in the raw papers. This aspect is left for future investigation.
- The optimal ordering of metadata in modes was not thoroughly examined, despite its potential significance in constructing representative texts.
- We decided against fine-tuning any model for reference linking because all models demonstrated very strong accuracy under the zero-shot setting.

In future, we will explore modelling the dependencies among metadata fields and expand our experiments to include papers from beyond the biomedical domain.

References

Dag W Aksnes. 2006. Citation rates and perceptions of scientific contribution. *Journal of the American Society for Information Science and Technology*, 57(2):169–185.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. 2020. SPECTER: Document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2270–2282, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mateusz Fedoryszak, Dominika Tkaczyk, and Łukasz Bolikowski. 2013. Large scale citation matching using apache hadoop. In *Research and Advanced Technology for Digital Libraries*, pages 362–365, Berlin, Heidelberg. Springer Berlin Heidelberg.

Santo Fortunato, Carl T Bergstrom, Katy Börner, James A Evans, Dirk Helbing, Staša Milojević, Alexander M Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, et al. 2018. Science of science. *Science*, 359(6379):eaao0185.

Yannis Foufoulas, Lefteris Stamatogiannakis, Harry Dimitropoulos, and Yannis Ioannidis. 2017. High-pass text filtering for citation matching. In *Research and Advanced Technology for Digital Libraries: 21st International Conference on Theory and Practice of Digital Libraries, TPDL 2017, Thessaloniki, Greece, September 18-21, 2017, Proceedings 21*, pages 355–366. Springer.

Behnam Ghavimi, Wolfgang Otto, and Philipp Mayr. 2019. Exmatcher: Combining features based on reference strings and segments to enhance citation matching.

Nianlong Gu, Yingqiang Gao, and Richard H. R. Hahnloser. 2022. Local citation recommendation with hierarchical-attention text encoder and scibert-based reranking. In *Advances in Information Retrieval*, pages 274–288, Cham. Springer International Publishing.

Steve Hitchcock, Les Carr, Stephen Harris, JMN Hey, and Wendy Hall. 1997. Citation linking: Improving access to online journals. In *Proceedings of the second ACM international conference on Digital libraries*, pages 115–122.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. 2015. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 1–6.

Steve Lawrence, C Lee Giles, and Kurt D Bollacker. 1999. Autonomous citation matching. In *Proceedings of the third annual conference on Autonomous Agents*, pages 392–393.

Zhentao Liang, Jin Mao, Kun Lu, and Gang Li. 2021. Finding citations for pubmed: a large-scale comparison between five freely available bibliographic data sources. *Scientometrics*, 126:9519–9542.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Andrew McCallum, Kamal Nigam, and Lyle H Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178.

Erin C McKiernan, Lesley A Schimanski, Carol Muñoz Nieves, Lisa Matthias, Meredith T Niles, and Juan P Alperin. 2019. Use of the journal impact factor in academic review, promotion, and tenure evaluations. *Elife*, 8:e47338.

Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, and Georg Rehm. 2022. Neighborhood contrastive learning for scientific document representations with citation embeddings. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11670–11688, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart J Russell, and Ilya Shpitser. 2002. Identity uncertainty and citation matching. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press.

Bimal Paudel, Connor Pedersen, Yang Yen, and Shin-Yi Lee Marzano. 2022. Fusarium graminearum virus-1 strain fgv1-sd4 infection eliminates mycotoxin deoxynivalenol synthesis by fusarium graminearum in fhb. *Microorganisms*, 10(8):1484.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Dominika Tkaczyk. 2018a. Matchmaker, matchmaker, make me a match.

Dominika Tkaczyk. 2018b. Reference matching: for real this time.

Dominika Tkaczyk. 2019. What if i told you that bibliographic references can be structured?

Ziyi Wang, Achal Neupane, Jiuhuan Feng, Connor Pedersen, and Shin-Yi Lee Marzano. 2021. Direct metatranscriptomic survey of the sunflower microbiome and virome. *Viruses*, 13(9):1867.

A Paper validation

Because PMCOA is a multilingual corpus but our linking methods are designed for English text, we used `lingua`[9] to identify the language of each paper based on its title. `lingua` performs better with longer texts, so we ignored papers whose titles were shorter than 10 characters.

[9] See github.com/pemistahl/lingua-py (Apache 2.0 license).

B Test split

Many of the semantic encoders we considered have been trained on scientific text, so for fair comparisons, we needed to ensure our test set was disjoint from all model's train set. Sent2vec was released in 2018 and the SBERT pretrained weights we chose were released in 2021. The train set for HAtten consisted only of papers published up till 2019. As for SciNCL, the train set was constructed from SciDocs (Cohan et al., 2020) and S2ORC (Lo et al., 2020), both of which were released in 2020. With these release dates in mind, we felt confident that using only sample references that cite papers published in and after 2022 would be suitable.

C Example representative texts

Table 3 provides examples of representative texts for a sample reference.

Mode	Representative text
T	direct metatranscriptomics survey of the sunflower microbiome and virome
TL	direct metatranscriptomics survey of the sunflower microbiome and virome wang naupane feng pedersen marzano
TV	direct metatranscriptomics survey of the sunflower microbiome and virome viruses

Table 3: Example representative texts (concatenated with a single space) for the reference by Paudel et al. (2022) to cite Wang et al. (2021).

D Truncation limits

For each metadata field, we chose truncation limits (see Table 4) based on the 95-th percentile number of characters in that field across all papers.

	T	**L**	**V**	**Y**	**A**
Truncation limit	200	120	60	4	2480

Table 4: Truncation limits in terms of number of characters per metadata field.

E Jaccard implementation

The most straightforward method of finding the candidate P whose representative r_P has maximal Jaccard similarity against the representative r_x of a sample x has a time complexity of $\mathcal{O}(nm)$, where n is the number of candidates and m is the maximal length of r_x. We improved the efficiency to

$\mathcal{O}(n+m)$ by using a trigram inverted index, but inferences were still slow. Therefore, we decided to optimise Jaccard linking with the GPU.

We represented each character in each string by its ASCII code. Because special characters and punctuation were removed during preprocessing, all ASCII codes contained up to three digits, and we ensured that all codes had exactly three digits by prepending with zero wherever necessary. This allowed us to associate each trigram with a unique integer; for instance, "bat" is associated with (0)98097116. In turn, after identifying and alphabetically sorting the trigrams in each r_P (resp. r_x), we could associate each P (resp. x) with a vector of trigram integers w_P (resp. w_x).

We enforced a mandatory vector length ℓ on the trigram integer vectors so that we could exploit the GPU for parallel processing. Vectors longer than ℓ were truncated and vectors shorter than ℓ were padded at the back with zeros. The exact value of ℓ depended on truncation limits relevant to the mode being used; for instance, with mode **TV**, we let $\ell = 200 + 60 + 1 = 261$.

We used `numba` (Lam et al., 2015) to compute Jaccard similarities with CUDA GPU programming and used 16 threads per block.

F Precision-recall trade-off for SBERT

When following our suggestion of using both semantic and lexical linking, our advice is to check the precision-recall trade-off for the semantic linking approach to select an appropriate threshold t.

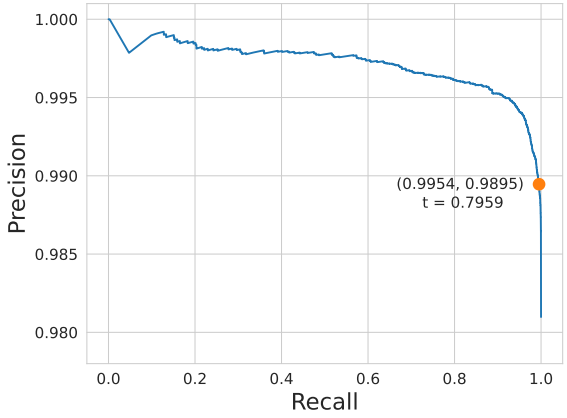

Figure 2: Precision-recall curve for links made on the two test subsets by SBERT with mode **TVL**. The position on the curve that is closest to the theoretical optimum (1.0, 1.0) is indicated with an orange dot.

In the case of SBERT with mode **TVL**, based on Figure F, we recommend using $t = 0.7959$.