Broadening BERT vocabulary for Knowledge Graph Construction using Wikipedia2Vec

Debanjali Biswas[1,0], Stephan Linzbach[1,0], Dimitar Dimitrov[1], Hajira Jabeen[1] and Stefan Dietze[1,2]

[1] *GESIS – Leibniz Institute for the Social Sciences, Cologne, Germany*

[2] *Heinrich-Heine-University Düsseldorf, Germany*

[0] *Equal Contribution*

Abstract

Recent advancements in natural language processing (NLP) have been driven by the utilization of large language models like BERT. These models, pre-trained on extensive textual data, capture linguistic nuances and relational information. Cloze-style prompts, which involve filling in missing words in a context, effectively tap into this stored knowledge for various NLP tasks. The "Knowledge Base Construction from Pre-trained Language Models (LM-KBC 2023)" challenge aims to harness language models' potential for knowledge graph construction through prompts, reducing the need for expensive data annotation. Our proposed approach in Track 1 focuses on expanding BERT's vocabulary with a task-specific vocabulary using Wikipedia2Vec embeddings, and fine-tuning the language model using OPTIPROMPT. These contributions aim to improve the knowledge graph population by leveraging the strengths of language models and external embeddings.

1. Introduction

In recent times, significant progress has been made in enhancing downstream NLP tasks by leveraging expansive language models like BERT [1], which undergo pretraining on extensive textual datasets. During their training, these models not only grasp linguistic nuances but also preserve relational information. To extract this stored knowledge, cloze-style prompts are employed. These prompts involve incomplete sentences or missing words in a given context, prompting the model to fill in the gaps. This approach has proven to be a valuable way of harnessing the latent potential of these language models for a variety of NLP tasks.

In this regard, the challenge of Knowledge Base Construction from Pre-trained Language Models (LM-KBC 2023)[1] explores the feasibility of leveraging the potential of language models in the downstream task of knowledge graph construction. Knowledge graphs have proven to be an efficient source for retrieving gold-standard relational information. However, populating knowledge graphs demands the extraction of relational information from various sources, including textual data. This task entails complex Natural Language Processing (NLP) pipelines

KBC-LM'23: Knowledge Base Construction from Pre-trained Language Models workshop at ISWC 2023

debanjali.biswas@gesis.org (D. Biswas); stephan.linzbach@gesis.org (S. Linzbach); dimitar.dimitrov@gesis.org (D. Dimitrov); hajira.jabeen@gesis.org (H. Jabeen); stefan.dietze@gesis.org (S. Dietze)

CEUR Workshop Proceedings (CEUR-WS.org)

[1] https://lm-kbc.github.io/challenge2023/

that encompass entity extraction, co-reference resolution, entity linking, and relation extraction. These components in turn rely heavily on expensive manual or automatically generated supervised data. To mitigate the dependence on expensive annotated data, the various knowledge stored in the self-supervised language models can be utilized to populate KGs. This is achieved by extracting relational knowledge using prompts, thus circumventing the need for resource-intensive data annotation processes.

1.1. Task Description

The goal of the knowledge graph construction task involves predicting the object entities associated with a given subject entity-relation pair. For instance, if presented with the subject entity-relation pair <*Cologne - CityLocatedAtRiver*>, the model's task is to predict the object entity <*Rhine*>. In this challenge, language model probing is utilized to extract the object entities, which in turn would populate the KGs. Using the instance provided above, a prompt such as "*Cologne is a city situated along the [MASK] river.*" could be employed to utilize a language model for the extraction of the object entity *Rhine* to fill the [MASK] token.

In a formal sense, when provided with an input subject-entity (s) and a relation (r), the objective is to employ LM probing to forecast the complete set of accurate object-entities $(o_1, o_2, ..., o_k)$.

The challenge comprises two tracks:

- Track 1: A small-model track with low computational requirements (<1 billion parameters).
- Track 2: An open track, allowing participants to select any Language Model (LM) they prefer.

1.2. Contributions

Our proposed approach falls under Track 1 of the challenge, and the following are our contributions:

1: Expanding BERT's vocabulary by incorporating a larger and more task-specific lexicon through integration with the Wikipedia2Vec [2] vocabulary.

2: Jointly training a mapping from the BERT embedding space to the Wikipedia2Vec embedding space and vice versa.

3: Fine-tuning our language model using OPTIPROMPT [3] in a retrieval based set up.

2. Methodology

Our proposed approach involved the enrichment of BERT's vocabulary with task-relevant terms, this is achieved by integrating entities from the Wikipedia2Vec [2] embeddings space. This augmentation results in a more refined representation of entities within BERT, making it capable of handling multi-token entities. The underlining assumption is that the entities in the Wikipedia2Vec space closely align with those required for the task. To put it simply, we're integrating additional knowledge from Wikipedia2Vec into BERT's vocabulary, allowing us to depict entities in a more comprehensive manner and increasing the model's understanding of the relationships between the LM vocabulary and the prompt vocabulary.

2.1. Wikipedia2Vec

Wikipedia2Vec [2] is a valuable tool for extracting embeddings of both words and entities (specifically, concepts linked to Wikipedia pages) from the Wikipedia corpus. This tool facilitates the simultaneous learning of embeddings for words and entities, positioning semantically related words and entities in proximity within a continuous vector space. This tool employs the traditional skip-gram model for learning word embeddings and an extension proposed in [4] to acquire embeddings specifically tailored for entities.

2.2. Linear Mapping

To broaden the scope of BERT vocabulary, it's necessary to establish a connection between BERT and Wikipedia2Vec. This is achieved by jointly training a bidirectional linear transformation which can transform a vector representation from the BERT space to the Wikipedia2Vec space and vice versa. To facilitate this training, an initial dataset is essential. In this context, we identify all the words within BERT that match the Wikipedia2Vec vocabulary. Subsequently, the focus is directed towards prioritizing the use of the entity embedding over the word embeddings from Wikipedia2Vec in conjunction with the BERT embedding.

Once the dataset is generated, the embeddings from both BERT and Wikipedia2Vec within the dataset are subjected to transformation and normalization before being input into the linear transformation process. For loss computation, we measure the Euclidean distance between the transformed embeddings and the embeddings of the neighboring context surrounding the target embedding. Additionally, we view the transformation as a classification task involving the nearest neighbors and thus employ the cross-entropy loss to predict the accurate target. The Faiss index [5] search is utilized to retrieve the nearby neighborhoods of the target embeddings for this particular task. Furthermore, the loss computation process is performed in both directions: from BERT to Wikipedia2Vec and vice versa. Following this, we consolidate the losses from each direction of the transformation.

2.3. Fine-tuning

Following the completion of the mapping between BERT and Wikipedia2Vec, we further refine our language model through fine-tuning using factual prompts via OPTIPROMPT [3], a continuous prompt optimization technique. In contrast to the conventional method of exploring a discrete token space, OPTIPROMPT directly aims to identify optimal prompts by crafting prompts through vectors within the embedding space. This approach builds upon the concept initially introduced by AUTOPROMPT [6].

Considering the enlarged vocabulary of BERT, we make use of a linear transformation that has been trained before to find the embedding of the subject entity within the Wikipedia2Vec space and then bring it back to the BERT space. This enriched embedding provides BERT with additional entity-related information. We then leverage this embedded representation of the

Table 1
Detailed results of our proposed methodology on the test set

Relation	Precision	Recall	F1 score
BandHasMember	0.0000	0.0000	0.0000
CityLocatedAtRiver	0.0133	0.0300	0.0180
CompanyHasParentOrganisation	0.4933	0.5100	0.4950
CompoundHasParts	0.6768	0.6167	0.6341
CountryBordersCountry	0.3175	0.2804	0.2641
CountryHasOfficialLanguage	0.0256	0.0538	0.0338
CountryHasStates	0.0000	0.0000	0.0000
FootballerPlaysPosition	0.0933	0.2533	0.1343
PersonCauseOfDeath	0.1950	0.6800	0.1967
PersonHasAutobiography	0.0400	0.0350	0.0367
PersonHasEmployer	0.0167	0.0383	0.0223
PersonHasNoblePrize	0.2733	0.8900	0.3430
PersonHasNumberOfChildren	0.0000	0.0000	0.0000
PersonHasPlaceOfDeath	0.1233	0.5300	0.1300
PersonHasProfession	0.0500	0.0895	0.0604
PersonHasSpouse	0.0000	0.0000	0.0000
PersonPlaysInstrument	0.1133	0.1753	0.1313
PersonSpeaksLanguage	0.1467	0.2843	0.1848
RiverBasinsCountry	0.2000	0.3917	0.2462
SeriesHasNumberOfEpisodes	0.0000	0.0000	0.0000
StateBordersState	0.0133	0.0108	0.0113
Average	**0.1329**	**0.2319**	**0.1401**

subject entity to formulate prompts during the fine-tuning process. Additionally, our method entails setting up fine-tuning as a retrieval task employing a Softmax.

2.4. Fetching the Wikidata ID for the predicted object entities

In order to retrieve the accurate Wikidata ID of object entities, we utilize the Wikidata API. During the inference process, BERT generates a representation for the *[MASK]* token within the prompts. This representation corresponds to the predicted object token within the BERT space. We subsequently apply the trained mapping to convert this embedding back into the Wikipedia2Vec space, and then utilize the Wikidata API to fetch the correct Wikidata ID by employing the accurate entity name from Wikipedia.

3. Experiments and Results

In this section, we present the outcomes of our method for the task of knowledge graph construction via prompts. The code for our suggested approach can be accessed on our GitHub repository: https://github.com/debanjali05/LM-KBC2023-GESIS. Detailed results of our approach on the test dataset are furnished in Table 1. These outcomes offer us valuable insights

Table 2
Comparison of our proposed methodology with baseline in Track 1

Method	Precision	Recall	F1 score
`Baseline`	0.1418	0.1467	0.1399
`Our Approach`	0.1329	0.2319	0.1401

into the performance of our approach across different relations. In Table 2, a comparison is drawn between the results of our approach and the baseline method using BERT.

4. Conclusion

In this work, we introduce a novel strategy that aims to augment the BERT vocabulary by integrating entity-specific embeddings sourced from Wikipedia2Vec. This approach has been developed to address the task of knowledge graph construction, which involves the creation and enrichment of knowledge graphs. Our work aligns with Track 1 of the LM-KBC 2023 challenge, which specifically focuses on leveraging language models to enhance knowledge graph construction processes. By implementing this approach, we endeavor to enhance the ability of language models to capture and represent relational information in the context of knowledge graphs. This endeavor is particularly pertinent in the realm of natural language processing, as it contributes to the ongoing advancement of techniques that facilitate the extraction and organization of structured information from unstructured textual data through the use of language models.

References

[1] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, 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), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423. doi:10.18653/v1/N19-1423.

[2] I. Yamada, A. Asai, J. Sakuma, H. Shindo, H. Takeda, Y. Takefuji, Y. Matsumoto, Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 23–30. URL: https://aclanthology.org/2020.emnlp-demos.4. doi:10.18653/v1/2020.emnlp-demos.4.

[3] Z. Zhong, D. Friedman, D. Chen, Factual probing is [MASK]: Learning vs. learning to recall, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Online, 2021, pp. 5017–5033. URL: https://aclanthology.org/2021.naacl-main.398. doi:10.18653/v1/2021.naacl-main.398.

[4] I. Yamada, H. Shindo, H. Takeda, Y. Takefuji, Joint learning of the embedding of words and entities for named entity disambiguation, in: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Association for Computational Linguistics, Berlin, Germany, 2016, pp. 250–259. URL: https://aclanthology.org/K16-1025. doi:10.18653/v1/K16-1025.

[5] J. Johnson, M. Douze, H. Jégou, Billion-scale similarity search with GPUs, IEEE Transactions on Big Data 7 (2019) 535–547.

[6] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, S. Singh, AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online, 2020, pp. 4222–4235. URL: https://aclanthology.org/2020.emnlp-main.346. doi:10.18653/v1/2020.emnlp-main.346.