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 and relational knowledge. Therefore, cloze-style prompts, which involve filling in missing words in a sentence, can be used to solve knowledge-intensive NLP tasks with the help of a language model. 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. In particular, contestants are challenged to infer the correct Wikidata ID of objects, given a prompt used to link subject, relation, and object. Automatically inferring the correct objects would help in reducing the need for an expensive manual graph population. Our proposed approach in Track 1 focuses on expanding BERT's vocabulary with a task-specific one (i.e., Wikipedia2Vec) and facilitating its usage through prompt tuning with OPTIPROMPT.

1. Introduction

Recently, significant progress has been made in enhancing downstream NLP tasks by leveraging pre-trained language models like BERT [1], which infer their language understanding on extensive textual datasets. The fact that pre-trained LMs not only understand language but also capture relational information from these datasets makes them a valuable knowledge source for automatic knowledge graph construction. To extract this stored knowledge, cloze-style sentences (e.g., '*Dante was born in [MASK].*') are used to prompt the model to truthfully complete relation facts. Such relational facts consist of knowledge triples in the form of subject, relation, object.

In this regard, the challenge of Knowledge Base Construction from Pre-trained Language Models (LM-KBC 2023)[1] [2] explores the capability of language models to solve the downstream task of knowledge graph construction. Knowledge graphs have proven to be an efficient source for retrieving relational information. However, populating knowledge graphs demands the extraction of relational information from various sources, including textual data. This task

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/

entails complex Natural Language Processing (NLP) pipelines that encompass components for entity extraction, co-reference resolution, entity linking, and relation extraction. These components in turn rely heavily on expensive manually or automatically generated supervised data. To mitigate the dependence on expensive data annotations, the relational knowledge stored in the self-supervised language models can be utilized to populate KGs. 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 and offers 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.

With the approach proposed in this work, we addressed Track 1 of the challenge and ranked second. The code for our suggested approach can be accessed on our GitHub repository[2].

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 task is to predict the object entity <*Rhine*>. In this challenge, when provided with an input subject-entity (s) and a relation (r), the objective is to employ language model probing to retrieve the complete set of accurate object-entities ($o_1, o_2, ..., o_k$) to solve the knowledge graph construction tasks. Language model probing means asking the model to fill the masked token in a cloze-style prompt such as "*Cologne is a city situated along the [MASK] river.*" with the correct list of object entities, in this case <*Rhine*>. Furthermore, the task requires linking this list of object entities to the correct Wikidata [3] entity IDs, in this example <*Q584*>.

1.2. Dataset

The LM-KBC challenge provides a dataset with training and validation splits, along with the unseen test set. This data contains 21 distinct relations, for each of the relations one prompt template is given. Additionally, the dataset provides up to 100 training and validation examples per relation, each example containing a subject entity, the Wikidata ID of the subject entity, the relation, and a list of object entities with their respective Wikidata IDs. Four out of the 21 relations contain zero cases, where none of the object entities are correct. The dataset can be found at a Github[3] repository, which also provides an in-depth dataset statistic.

[2] https://github.com/debanjali05/LM-KBC2023-GESIS
[3] https://github.com/lm-kbc/dataset2023

1.3. Task Evaluation

For each example in the test set, Wikidata ID predictions are evaluated by calculating precision and recall against ground-truth values. The final macro-averaged F1-score is used to rank the participating systems submitted to the challenge.

1.4. Task Challenges

The primary challenge with the task of using language models for knowledge graph construction is that the language model is required to predict multi-tokens (e.g., <*Pharrell Williams*>) and multi-label object entities (i.e., having 0 to N object entities for a given subject-relation pair). While encoder-only language models like BERT [1] perform well at predicting single-token objects (e.g., <*Rhine*>), they struggle when tasked with predicting objects comprised of multiple tokens (e.g., <*Pharrell Williams*>). While, sequence-to-sequence language models such as T5 [4] offer a potential solution for handling multi-token answers, they fail to address the multi-label object entities. For multiple object entities, T5 can only solve the multiple labels task by taking into account the order of the labels. For instance, when dealing with a subject entity <*Hexadecane*> and relation <*CompoundHasParts*>, the object entities may be [<*Carbon*>, <*Hydrogen*>]. When fine-tuning T5 on this list of object entities, the model would also learn to reproduce the ordering of the elements seen during training, although an ordering is not indicated by the data. The second challenge is to predict the zero cases, where one could add a zero-case object entity like 'no objects' to the label set. However, this would not resemble a correct English sentence. (e.g., '*The parent organization of Turkcell is no objects.*') The third challenge is to link the predicted tokens, representing the object entity for the given subject-relations pair, to the correct Wikidata ID. A naive solution would involve using the Wikidata API [4] to disambiguate the tokens and link them to Wikidata entities. However, this approach neglects to take into account the contextual information provided by the subject-relation-pair.

2. Methodology

As previously discussed in Section 1.4, using sequence-to-sequence models is not a viable option for this task. We resort to using BERT since a preliminary experiment showed promising results for predicting single token entities in particular. However, the main limitations of BERT are the rather small and single-token-only vocabulary. To address this limitation, our approach involves mapping BERT representations to a significantly larger and task-specific representation space containing multi-token entities. One strategy for achieving this is by utilizing word embeddings like Glove embeddings, comprising a 400K vocabulary size and n-grams representation. However, GloVe embeddings [5] only cover about 50% of the relevant tokens in our case, entities. A task-specific target representation space, where BERT could be mapped, is Wikipedia2Vec, which covers around 70% of the relevant tokens when using the freely available pre-trained version [5].

[4] https://www.wikidata.org/w/api.php
[5] https://wikipedia2vec.github.io/wikipedia2vec/

Wikipedia2Vec [6] is a joint embedding of both words and entities (i.e., concepts linked to Wikipedia pages) calculated for a Wikipedia dump. This representation positions words and entities that are semantically related in their respective proximity. Wikipedia2Vec optimizes the traditional skip-gram model for learning word embeddings and an extension proposed in Yamada et al. [7] to acquire embeddings specifically tailored for entities. Using Wikipedia2Vec as a task-specific representation space to map BERT provides three significant advantages: (i) a substantially larger vocabulary compared to BERT (i.e., 4M embeddings, with 2M word embeddings and 2M Wikipedia article embeddings), (ii) many of the Wikipedia entities (here, they represent articles) are multi-token (e.g., *ENTITY/Barack_Obama*), and (iii) Wikipedia articles have a direct link to Wikidata IDs (e.g., *ENTITY/Barack_Obama* is linked to the Wikidata ID *Q76* via the entity's Wikipedia page.). These three advantages address the challenges of multi-token entities, multi-label entities, and linking to Wikidata IDs, as outlined in Section 1.4 except for the zero cases, which are handled later in Section 2.2. Our method consist of three steps: (i) pre-training BERT transformation to Wikipedia2Vec representation space (cf. Section 2.1), (ii) fine-tuning mapping to contextualized BERT embeddings (cf. Section 2.2), and (iii) inference of correct entities for subject-relation-pair stated in a prompt (cf. Section 2.3).

2.1. Pre-training BERT to Wikipedia2Vec Mapping

In order to expand the vocabulary of BERT, we are required to map BERT word embeddings to the Wikipedia2Vec embedding space. We accomplish this by pre-training a neural network (a single-layer feed-forward linear mapping) denoted as $f : D_{BERT} \rightarrow D_{Wikipedia2Vec}$, which transforms BERT embeddings to a corresponding vector representation in the Wikipeida2Vec embedding space. To facilitate the training of this mapping as a first step, a dataset was systematically generated. The generation process entailed string matching the token in BERT vocabulary to their Wikipedia2Vec correspondents, while prioritizing entity embeddings over word embeddings from Wikipedia2Vec. The resulting dataset denoted as $D = \{(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)\}$ consists of pairs (x_i, y_i) where x_i is a token embedding from BERT (e.g., the embedding of the token '*France*' or '*hello*') and y_i is the target embedding in Wikipedia2Vec space (e.g., the embedding of the entity '*ENTITY/France*' or the word '*hello*', respectively). It's important to note that this simple string matching provides us with only positive examples. To generate hard negative examples $\hat{Y}$, we use a Faiss index [8] to retrieve the closest entities/words representations from the Wikipedia2Vec embedding space to the target embedding y_i. Since Wikipedia2Vec contains more than 4 mio embeddings, we use the Faiss Index to efficiently query the embedding space Zhan et al. [9]. The Faiss index allows combining multiple indexes to achieve better search performance and balance the trade-offs between accuracy and efficiency.The next step is to optimize InfoNCE [10] loss function between a single x_i and y_i with N many negative examples $\hat{Y}$:

$$L(x_i, y_i) = \log \frac{\exp^{-d(f(x_i), y_i)}}{\exp^{-d(f(x_i), y_i)} + \sum_{\hat{y_i} \in \hat{Y}} \exp^{-d(f(x_i), \hat{y_i})}} \tag{1}$$

Where d is the Euclidean distance, $\hat{Y}$ are N many nearest neighbors of y_i in the Wikipedia2Vec embedding space and $N = 1000$. This loss function incorporates principles from informa-

tion theory, it not only separates positive and negative pairs but also maximizes the mutual information between $f(x_i)$ and the positive sample y_i.

2.2. Prompt and fine-tuning BERT to Wikipedia2Vec Mapping

Using the available prompt templates[6], we generated prompts for all subject-relation pairs in the training set, provided by the organizers. For each prompt, we fetch the masked token representation from BERT, denoted as h_B, which is then transformed using the pre-trained linear mapping f to a representation in the Wikipedia2Vec space, as $h_W = f(h_B)$ (cf. Section 2.1). Subsequently, we search for the top 200 nearest neighbors of h_W in the Wikipedia2Vec space using a Faiss index, which will represent the negative examples $\hat{Y} = \{\hat{y}_0, \hat{y}_1, ..., \hat{y}_{199}\}$ in fine-tuning. To obtain positive examples $Y = \{y_0, y_1, ..., y_n\}$ (where n is the number of correct object entity IDs for each subject-relation pair), we utilize the Wikidata object IDs corresponding to each subject-relation pair to retrieve the respective Wikipedia articles and subsequently their associated Wikipedia2Vec representations. For the zero object entity cases, we incorporated the zero vector as the correct target. The primary objective of the fine-tuning is to increase the Euclidean distance d between the transformed masked token representations h_W and negative examples $\hat{Y}$, while reducing the distance to positive examples Y. To achieve this, the supervised contrastive loss (in Equation 2) as proposed by Khosla et al. [11] is utilized. This loss formulation is applicable to multi-label classification and, unlike binary cross-entropy, it is not fragile to long-tail distributions. Furthermore, we can interpret the resulting output as a probability distribution over a set of candidate labels. Conceptually, one can think about this loss as the average of several independent cross-entropy losses. The exponent of the negative Euclidean distance d is used as a custom similarity function (in Equation 3) analog to the linear mapping training.

$$\mathcal{CL} = -\frac{1}{|Y|} \sum_{y_i \in Y} \log \frac{\exp^{sim(h_W, y_i)}}{\sum_{y_j \in Y} \exp^{sim(h_W, y_j)} + \sum_{\hat{y}_i \in \hat{Y}} \exp^{sim(h_W, \hat{y}_i)}} \tag{2}$$

$$sim = \exp(-d(h_W, Y)) \tag{3}$$

To allow BERT to effectively adapt to the task, we perform prompt tuning using OPTIPROMPT [12], a continuous prompt optimization technique. OPTIPROMPT aims to identify optimal prompts by crafting prompts through vectors within the embedding space. OPTIPROMPT optimization and the fine-tuning of the linear mapping between BERT and Wikipedia2Vec are performed jointly.

2.3. Inference using BERT to Wikipedia2Vec Mapping

As depicted in Figure 1, for a given subject-relation pair, we first generate a prompt using the provided template and replace the relation tokens in the prompt with the optimal prompt tokens generated using OPTIPROMPT during the prompt and fine-tuning phase, as explained in Section 2.2 (cf. Step 1 in Figure 1). Following this, the modified prompt is employed to generate

[6] https://github.com/lm-kbc/dataset2023/blob/main/prompts.csv

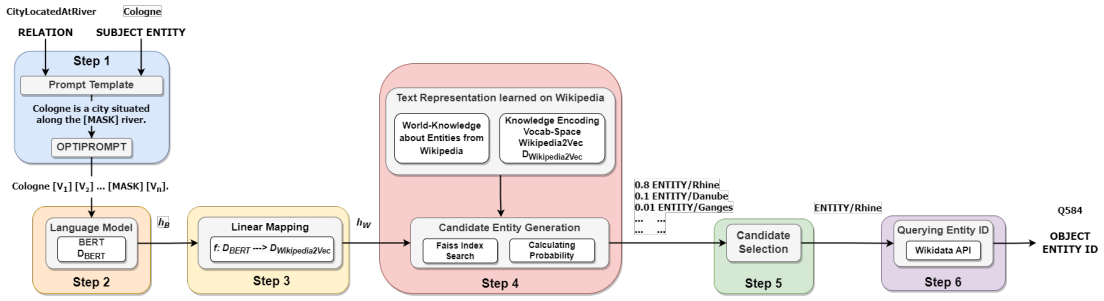

Figure 1: Inference pipeline showing the different steps of our approach.

the mask token representation using BERT (cf. Step 2 in Figure 1). This representation is then transformed using the fine-tuned linear mapping (cf. Step 3 in Figure 1). We utilize the Faiss index to fetch the top 200 closest (in terms of L2 norm) candidate entities to the transformed representation of the masked token. Simultaneously, we also calculate the similarity between candidate entities and the transformed masked token representation using Equation 1, which yields a probability distribution over the candidates (cf. Step 4 in Figure 1). To obtain the final set of object entities, we select the top k answers where k is inferred by $k = 1//max(probability)$ (cf. Step 5 in Figure 1). Finally, we use the Wikidata API to query the correct Wikidata IDs for the set of selected object entities (cf. Step 6 in Figure 1).

3. Results and Discussion

Detailed results of our approach on the test dataset are shown in Table 1. These outcomes provide insights into the performance of our approach across different relations, which is discussed below. In Table 2, a comparison is drawn between the results of our approach, the baseline method using BERT, and the winning approach.

The overall results of our approach are only marginally above the BERT baseline provided by the challenge organizers (cf. Table 2). The performance of BERT decreased in the relations concerned with countries and languages. One reason for this could be that BERT has the required words in its vocabulary. Thus, is able to produce meaningful outputs. On the other hand, we were able to increase the models' performance for relations that depend on fine-tuning (i.e., <*CompoundHasParts*>, <*PersonPlaysInstrument*>, <*PersonHasNobelPrize*>). However, in this case, our model just learned to predict the most common object entities without much subject sensitivity. There are some relations that our model is now able to predict better, without a dataset-specific explanation. In particular, the increased performance for the <*PersonHasAutobiography*> and <*CompanyHasParentOrganisation*> show that we are indeed able to predict entities that were not originally in BERT vocabulary but learned from the Wikipedia2Vec embedding space.

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**

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

Method	Precision	Recall	F1 score
BERT - baseline	0.1418	0.1467	0.1399
Our Approach	0.1329	0.2319	0.1401
Winner	0.3950	0.3925	0.3226

4. Conclusion

In this work, a novel approach is introduced that aims to expand the BERT vocabulary by integrating entity-specific embeddings sourced from Wikipedia2Vec, in order to address the task of knowledge graph construction using language model probing. To summarize, our proposed approach comprises the following three steps: (i) Expanding BERT's vocabulary by incorporating a larger and more task-specific representation through integration with the Wikipedia2Vec vocabulary. (ii) Training a mapping from the BERT embedding to the Wikipedia2Vec embedding space. (iii) Simultaneously performing prompt tuning using OPTIPROMPT and fine-tuning the mapping between BERT embedding to the Wikipedia2Vec embedding space. In conclusion, we observed that our model lacked sensitivity when predicting objects for different subject entities.

We think that this behavior might be explained by three sources. Firstly, reducing the embedding size (768 (BERT) to 500 (Wikipedia2Vec)) thus losing parts of the information encoded in BERT. Secondly, Wikipedia2Vec knows more than 100 times the amount of words/entities than BERT. Therefore, we hypothesize that the embeddings might not carry enough information to distinguish between unknown representations. Lastly, it is not clear that BERT encodes the given subject texts in a meaningful way. Thus, we think that our pipeline would benefit from an end-to-end pre-training step, that increases both the sensitivity of the output representation and the knowledge available in the input representation. In the best case, retaining the contextual understanding of BERT while enabling the usage of a much larger and task-dependent vocabulary.

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] S. Singhania, J.-C. Kalo, S. Razniewski, J. Z. Pan, Lm-kbc: Knowledge base construction from pre-trained language models, semantic web challenge @ iswc, CEUR-WS (2023). URL: https://lm-kbc.github.io/challenge2023/.

[3] D. Vrandečić, M. Krötzsch, Wikidata: A free collaborative knowledgebase, Communications of the ACM 57 (2014) 78–85. doi:10.1145/2629489.

[4] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[5] J. Pennington, R. Socher, C. Manning, GloVe: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1532–1543. URL: https://aclanthology.org/D14-1162. doi:10.3115/v1/D14-1162.

[6] 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.

[7] 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.

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

[9] J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, S. Ma, Jointly optimizing query encoder and product quantization to improve retrieval performance, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 2487–2496.

[10] A. v. d. Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, arXiv preprint arXiv:1807.03748 (2018).

[11] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, Advances in neural information processing systems 33 (2020) 18661–18673.

[12] 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.