

Tokenizing the Graph: Attentive Byte-pair Encoding for Knowledge Graph Embeddings

N'Dah Jean Kouagou¹, Arnab Sharma¹, Harshit Purohit¹, Caglar Demir¹ and Axel-Cyrille Ngonga Ngomo¹

¹Data Science Research Group, Heinz Nixdorf Institute, Paderborn University, Fürstenallee 11, 33102 Paderborn, Germany

Abstract

In recent years, knowledge graph embedding (KGE) models have been successfully applied to tackle various challenging tasks, including link prediction and query answering. However, due to the transductive nature of the embedding models, existing methods often face issues in handling real-world dynamic knowledge graphs involving heterogeneous information about the world. Following the recent success of *attention* mechanism in the natural language processing domain, in this work, we propose a novel architecture to compute knowledge graph embeddings. More specifically, we propose an attentive byte-pair encoding technique (BYTE) to bootstrap embeddings of entities and relation types from a sequence of their byte-pair encoded subword units. Given a triple (h, r, t) , BYTE acts as an encoder of h , r and t by combining the embeddings of their subword units. Thereafter, BYTE applies a KGE model's post-lookup-table layers as decoders which produce the likelihood of an input triple being true. Compared to the conventional setting, BYTE leads to massive feature reuse via weight tying, since it forces a knowledge graph embedding model to learn embeddings for subword units rather than directly for entities and relation types. Consequently, the sizes of embedding matrices are no longer bound to the unique number of entities and relation types of a knowledge graph. Experimental results show that BYTE generally improves the link prediction performance of knowledge graph embedding models. Additionally, under lexical perturbations of entity and relation representations, BYTE outperforms standard embedding-based models.

Keywords

Knowledge graph embedding, Link prediction, Byte-pair encoding, Tokenization, Transformer

1. Introduction

The field of natural language processing (NLP) has reached an unprecedented level with the advent of large language models (LLMs) [1, 2]. Such models have demonstrated significant capabilities in understanding and generating natural language text. Behind the success of LLMs, tokenizers play a fundamental role [3]. Tokenizers allow the transformation of plain text into smaller pieces (tokens) which serve as the building blocks for text understanding and generation [4, 5]. Through tokenization, LLMs can not only handle language variability and ambiguity (e.g., syntactic errors) but also process long sequences more efficiently. Current knowledge graph embedding (KGE) models are developed to work in a transductive setting¹, with a few partially supporting the inductive setting². These models initialize a fixed-size vocabulary from an input knowledge graph (KG) and map each entity and relation type to a unique element of the vocabulary before projecting them into a d -dimensional vector space.

Research in this field has mainly focused on designing models tailored towards the transductive link prediction task [6, 7, 8, 9, 10, 11]. This task is often formulated as the problem of learning a parameterized scoring function $\phi_{\Theta} : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$ such that $\phi_{\Theta}(h, r, t)$ ideally signals the likelihood of (h, r, t) being true [8]. However, most existing approaches assume that only entities can be novel, while all relation types must be observed during training. Moreover, there might be "new" entities

Posters, Demos, Blue Sky, and Tutorials at SEMANTiCS 2026, Sep 2026, Ghent, Belgium

✉ ndah.jean.kouagou@upb.de (N. J. Kouagou); arnab.sharma@uni-paderborn.de (A. Sharma); hpurohit@mail.uni-paderborn.de (H. Purohit); caglar.demir@upb.de (C. Demir); axel.ngonga@upb.de (A. N. Ngomo)
id 0000-0002-4217-897X (N. J. Kouagou); 0009-0007-8515-5253 (A. Sharma); 0009-0008-2944-9473 (H. Purohit); 0000-0001-8970-3850 (C. Demir); 0000-0001-7112-3516 (A. N. Ngomo)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹Entities and relation types involved in the inference phase are also seen during training

²In the inductive setting, KGE approaches are tasked to perform inference over unseen entities or relation types by leveraging learned patterns

or relation types arising from syntactic variations, for instance, “U.S.A” instead of “USA”, “authored by” changed to “written by”, etc. Such variations can lead to out-of-vocabulary entities and relation types that differ only linguistically, but not semantically. KGE models that rely on fixed vocabularies or learned identifiers often face challenges in performing link prediction when such situations arise. To deal with such scenarios, a natural solution would be to consider inductive models [12, 13, 14, 15, 16]. Yet, most existing methods assume that only entities can be new, and all relation types should be observed during training.

Our main research question in this work is, *how can we retain the advantage of the existing knowledge graph embedding models while reducing their dependence on fixed vocabularies and improving their robustness to lexical variations in entities and relation types?* We propose an attentive byte-pair encoding layer BYTE to improve existing KGE models on knowledge graphs where entity and relation type names carry semantic information. BYTE represents each entity and relation type as a sequence of byte-pair encoded subword units (tokens). More specifically, during training, given a triple (h, r, t) , a KGE model predicts its likelihood by combining embeddings of subword units representing (h, r, t) . Hence, a KGE model does not plainly retrieve embedding vectors of entities and relation types but constructs them on the fly using an attention mechanism [17] on subword units. Our extensive experiments with 4 state-of-the-art KGE models over 8 benchmark knowledge graphs suggest that BYTE improves the link prediction performance of KGE models on knowledge graphs where semantic information on syntactic representations of triples are visible. Furthermore, we evaluate BYTE under different ratios of lexical perturbations, where entities and relation types are rewritten using alternative surface forms. These experiments show that BYTE is more robust than conventional embedding-based models under such representational variation. We provide an open-source implementation of BYTE in the DICE-Embeddings framework [18]³.

2. Background and Related Works

2.1. Knowledge Graphs

A knowledge graph (KG) is often formally defined as a set of triples $\mathcal{G} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where each triple $(h, r, t) \in \mathcal{G}$ contains two entities/nodes $h, t \in \mathcal{E}$ and a relation/edge $r \in \mathcal{R}$. Therein, $\mathcal{E}$ and $\mathcal{R}$ denote a finite set of entities/nodes and relations/edges. These facts have been used in a wide range of applications, including web search, question answering, and recommender systems [19, 20]. Despite their wide application domains, most KGs on the web are incomplete [19].

2.2. Link Prediction and Knowledge Graph Embeddings

Link prediction on KGs refers to the task of identifying entities that share some (hidden) relationships. This task is often formulated as the problem of learning a parameterized scoring function $\phi_{\Theta} : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$ such that $\phi_{\Theta}(h, r, t)$ signals the likelihood of (h, r, t) is true [8]. For instance, given the triples $(western_europe, locatedin, europe)$ and $(germany, locatedin, western_europe) \in \mathcal{G}$, a good scoring function is expected to return high scores for $(germany, locatedin, europe)$, while returning a lower score for $(europe, locatedin, germany)$.

Most KGE models learn continuous vector representations of entities and relation types tailored towards predicting missing triples. Three training strategies are commonly used: negative sampling [21], 1vsAll [22], and KvsAll [8, 23]. In KvsAll, for a given pair (h, r) , predicted scores for all entities are computed and normalized via the logistic sigmoid; the cross-entropy loss over binary label vectors often yields state-of-the-art performance [9, 24].

³Code: <https://github.com/dice-group/dice-embeddings>

2.3. Transductive and Inductive Approaches

A plethora of KGE models have been developed over the last decade [25, 26, 27]. Most KGE models map entities $e \in \mathcal{E}$ and relation types $r \in \mathcal{R}$ found in a KG $\mathcal{G} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ to a d -dimensional vector space [20]. Although recent works continue improving predictive performance by adding more complex neural architectures, the resulting models inherit fundamental limitations: the embedding layer grows linearly w.r.t. the number of entities, and unseen entities and relation types cannot be handled at inference time. Inductive approaches handle unseen entities via rule mining (e.g., AnyBURL [14]), graph neural networks (e.g., NodePiece [12] and ULTRA [28]), or transformer-based models such as LP-BERT [16]. In contrast, BYTE is model-agnostic and can be integrated with any existing transductive KGE model as a universal encoder, while remaining within the efficient embedding-based link prediction paradigm.

3. Methodology

BYTE augments any transductive KGE model with three steps (Figure 1): **(i)** byte-pair tokenization of each triple component, **(ii)** embedding lookup for subword units, and **(iii)** attention-augmented linear mapping back to $\mathbb{R}^d$. Given $(h, r, t) \in \mathcal{G}$, these steps are composed as $\text{BYTE} = \text{LinAttn} \circ \text{Emb} \circ \text{Tok}$.

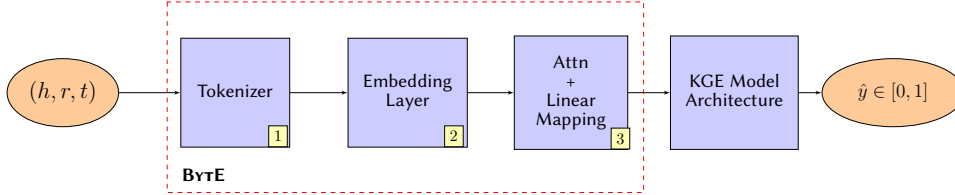


Figure 1: Workflow of a KGE model using our byte-pair encoding approach.

$\text{Tok}(\cdot)$ maps each entity and relation type string to at most m subword ids, padding shorter sequences and truncating longer ones. We use either the pre-trained GPT2 tokenizer or a KG-trained tokenizer (T2) with minimum merge frequency 2. $\text{Emb}(\cdot)$ retrieves a trainable d -dimensional vector for each subword unit, yielding three matrices in $\mathbb{R}^{m \times d}$. $\text{LinAttn}(\cdot)$ first applies *intra-component attention* over subword units within each triple component, then flattens and linearly projects each matrix to a vector in $\mathbb{R}^d$:

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{E}\mathbf{W}_Q(\mathbf{E}\mathbf{W}_K)^\top}{\sqrt{d}}\right), \quad \mathbf{E}' = \mathbf{A}(\mathbf{E}\mathbf{W}_V), \quad \mathbf{z} = \mathbf{W} \text{vec}(\mathbf{E}') + \mathbf{b}, \quad (1)$$

where $\mathbf{E} \in \mathbb{R}^{m \times d}$ is a subword embedding matrix and $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V, \mathbf{W}, \mathbf{b}$ are shared trainable parameters. The resulting head, relation, and tail vectors are passed to any downstream KGE model's post-lookup-table layers, which output $\hat{y} \in [0, 1]$ as the predicted triple likelihood.

4. Experimental Setup

We evaluate BYTE on Countries-S1/S2/S3, UMLS, WN18RR, FB15K-237, and NELL-995 (h25, h75, h100) using DistMult, ComplEx, QMult, and Keci as backbones. We follow the KvsAll setup with cross-entropy loss, Adam optimizer, and a grid search over the learning rate $\{0.1, 0.01, 0.001\}$ and embedding dimension $d \in \{32, 64\}$ for 500 epochs [24]. We report MRR and compare base models, GPT2, T2, Attn+GPT2, Attn+T2, and LP-BERT [16] as a text-based baseline.

Table 1

Link prediction MRR. **GPT2/T2**: pretrained vs. KG-trained BPE (min. frequency 2); **Attn+***: attention after embedding; **LP-Bert**: BERT baseline (“-” if not applicable).

	Countries-S1				Countries-S2				Countries-S3			
	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci
Without BYTE	0.457	0.389	0.360	0.489	0.346	0.383	0.404	0.220	0.166	0.132	0.210	0.092
GPT2	0.545	0.497	0.463	0.653	0.490	0.510	0.503	0.416	0.256	0.245	0.196	0.247
T2	0.493	0.542	0.550	0.581	0.443	0.300	0.434	0.434	0.324	0.214	0.214	0.258
Attn+GPT2	0.296	0.296	0.342	0.499	0.252	0.282	0.292	0.429	0.216	0.226	0.217	0.191
Attn+T2	0.795	0.898	0.533	0.273	0.607	0.272	0.483	0.699	0.231	0.226	0.232	0.371
LP-Bert	0.019	-	-	-	0.015	-	-	-	0.022	-	-	-
FB15K-237 / WN18RR / UMLS												
	FB15K-237				WN18RR				UMLS			
	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci
Without BYTE	0.261	0.258	0.277	0.258	0.369	0.334	0.163	0.359	0.687	0.823	0.812	0.811
GPT2-Tokenizer	0.155	0.158	0.168	0.169	0.296	0.293	0.239	0.301	0.706	0.811	0.883	0.783
T2	0.277	0.268	0.169	0.168	0.366	0.348	0.331	0.354	0.703	0.846	0.861	0.781
Attn+GPT2	0.176	0.049	0.063	0.178	0.178	0.190	0.194	0.178	0.833	0.828	0.841	0.830
Attn+T2	0.296	0.292	0.288	0.293	0.223	0.357	0.383	0.374	0.656	0.731	0.799	0.842
LP-Bert	0.403	-	-	-	0.484	-	-	-	0.966	-	-	-
NELL-995 (h100, h75, h25)												
	h100				h75				h25			
	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci
Without BYTE	0.228	0.216	0.218	0.232	0.222	0.208	0.224	0.228	0.229	0.228	0.258	0.233
GPT2-Tokenizer	0.288	0.289	0.292	0.305	0.293	0.298	0.280	0.308	0.266	0.248	0.259	0.277
T2	0.271	0.265	0.284	0.279	0.274	0.272	0.277	0.287	0.289	0.286	0.259	0.281
Attention layer+GPT2	0.285	0.271	0.269	0.292	0.295	0.293	0.271	0.296	0.240	0.000	0.193	0.238
Attention layer+T2	0.305	0.301	0.309	0.301	0.303	0.301	0.297	0.303	0.260	0.241	0.233	0.254
LP-Bert	0.253	-	-	-	0.272	-	-	-	-	-	-	-

5. Results

Table 1 reports the link prediction results on the Countries, FB15K-237, WN18RR, UMLS, and NELL-995 benchmark datasets. Overall, results suggest that BYTE often improves the link prediction results on knowledge graphs where syntactic representations of triples are semantically meaningful.

Introducing textual tokenization via the GPT2 and T2 variants consistently improves link prediction performance across datasets. On the Countries datasets, semantic information on syntactic representations of triples is visible, e.g., (*western_europe*, *locatedin*, *europe*), (*germany*, *locatedin*, *western_europe*). With BYTE, KGE models learn embeddings for subword units, e.g., “western”, “_”, and “europe”, which are composed at test time (Section 3) rather than retrieved as fixed entity vectors. The GPT2 variant benefits from pretrained subword embeddings that encode rich linguistic regularities, yielding clear gains particularly on the Countries and WN18RR benchmarks. The T2 variant, trained directly on KG textual labels, often achieves better results on domain-specific datasets such as UMLS and NELL-995, where the local vocabulary differs from general-purpose corpora.

The addition of an attention mechanism, i.e., Attn+GPT2 and Attn+T2, further enhances model expressivity by allowing the model to dynamically weight contextual subwords. In particular, Attn+T2 achieves the best MRR in several cases, e.g., on Countries-S1. On FB15K-237, where entity identifiers are less linguistically meaningful, improvements are smaller and base models remain competitive.

Robustness under lexical perturbations. Table 2 reports results when 10% of entity and relation names are rewritten with lexical variants. BYTE improves 23 of 24 model-dataset combinations, with the largest gains on Countries-S1 and UMLS. On WN18RR, gains are milder and ComplEx remains

Table 2

Link prediction results with 10% perturbation on entities and relation types.

	UMLS				Countries-S1				WN18RR			
	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci
Without BYTE	0.1987	0.2039	0.2176	0.2088	0.4354	0.3674	0.3412	0.4662	0.3208	0.2947	0.0977	0.3084
With BYTE	0.2466	0.2543	0.2439	0.2422	0.6899	0.5101	0.6291	0.6280	0.3247	0.2640	0.2347	0.3185
LP-Bert	–	–	–	–	–	–	–	–	–	–	–	–
FB15k-237 / NELL-995-H100 / NELL-995-H70												
	FB15k-237				NELL-995-H100				NELL-995-H70			
	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci	DistMult	ComplEx	QMult	Keci
Without BYTE	0.2432	0.2407	0.2579	0.2403	0.1695	0.1525	0.1662	0.1602	0.1534	0.1661	0.1589	0.1552
With BYTE	0.2685	0.2693	0.2693	0.2702	0.2394	0.2460	0.2400	0.2600	0.2330	0.2398	0.2489	0.2515
LP-Bert	0.2508	–	–	–	0.2009	–	–	–	0.2110	–	–	–

slightly below the base model.

6. Conclusion and Future Work

We proposed BYTE to integrate subword tokenization and attention into transductive KGE models. On eight benchmarks it improves link prediction when surface forms are meaningful, and is more robust under lexical perturbation. Future work includes inductive link prediction and LLM-KGE hybrids.

Declaration on Generative AI

The authors used generative AI to paraphrase parts of this paper, then reviewed and edited the content and take full responsibility for it.

Acknowledgments

Supported by MKW NRW (WHALE, LFN 1-04) and BMFTR (KI-Akademie OWL, grant 16IS24057B).

References

- [1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (2019) 9.
- [2] L. Floridi, M. Chiriatti, GPT-3: its nature, scope, limits, and consequences, Minds Mach. 30 (2020) 681–694.
- [3] P. Rust, J. Pfeiffer, I. Vulić, S. Ruder, I. Gurevych, How good is your tokenizer? on the monolingual performance of multilingual language models, arXiv preprint arXiv:2012.15613 (2020).
- [4] T. Kudo, J. Richardson, Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, in: EMNLP (Demonstration), Association for Computational Linguistics, 2018, pp. 66–71.
- [5] S. Choo, W. Kim, A study on the evaluation of tokenizer performance in natural language processing, Applied Artificial Intelligence 37 (2023) 2175112. URL: <https://doi.org/10.1080/08839514.2023.2175112>. doi:10.1080/08839514.2023.2175112. arXiv:<https://doi.org/10.1080/08839514.2023.2175112>.
- [6] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data, in: Icml, 2011.
- [7] B. Yang, W.-t. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, arXiv preprint arXiv:1412.6575 (2014).

- [8] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [9] I. Balazevic, C. Allen, T. M. Hospedales, Tucker: Tensor factorization for knowledge graph completion, in: *EMNLP/IJCNLP (1)*, Association for Computational Linguistics, 2019, pp. 5184–5193.
- [10] S. Zhang, Y. Tay, L. Yao, Q. Liu, Quaternion knowledge graph embeddings, *Advances in neural information processing systems* 32 (2019).
- [11] D. Ruffinelli, S. Broscheit, R. Gemulla, You CAN teach an old dog new tricks! on training knowledge graph embeddings, in: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020. URL: <https://openreview.net/forum?id=BkxSmlBFvr>.
- [12] M. Galkin, E. G. Denis, J. Wu, W. L. Hamilton, Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs, in: *ICLR*, OpenReview.net, 2022.
- [13] W. L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: *NIPS*, 2017, pp. 1024–1034.
- [14] C. Meilicke, M. W. Chekol, D. Ruffinelli, H. Stuckenschmidt, Anytime bottom-up rule learning for knowledge graph completion, in: *IJCAI*, ijcai.org, 2019, pp. 3137–3143.
- [15] M. Galkin, X. Yuan, H. Mostafa, J. Tang, Z. Zhu, Towards foundation models for knowledge graph reasoning, *arXiv preprint arXiv:2310.04562* (2023).
- [16] D. Li, B. Zhu, S. Yang, K. Xu, M. Yi, Y. He, H. Wang, Multi-task pre-training language model for semantic network completion, *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 22 (2023). URL: <https://doi.org/10.1145/3627704>. doi:10.1145/3627704.
- [17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, *CoRR* abs/1706.03762 (2017). URL: <http://arxiv.org/abs/1706.03762>. arXiv:1706.03762.
- [18] C. Demir, A.-C. N. Ngomo, Hardware-agnostic computation for large-scale knowledge graph embeddings, *Software Impacts* 13 (2022) 100377. URL: <https://www.sciencedirect.com/science/article/pii/S2665963822000811>. doi:<https://doi.org/10.1016/j.simpa.2022.100377>.
- [19] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, *Proceedings of the IEEE* 104 (2015) 11–33.
- [20] A. Hogan, E. Blomqvist, M. Cochez, C. d’Amato, G. d. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge graphs, *ACM Computing Surveys (CSUR)* 54 (2021) 1–37.
- [21] A. Bordes, N. Usunier, A. García-Durán, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: *NIPS*, 2013, pp. 2787–2795.
- [22] T. Lacroix, N. Usunier, G. Obozinski, Canonical tensor decomposition for knowledge base completion, in: *International Conference on Machine Learning*, PMLR, 2018, pp. 2863–2872.
- [23] D. Ruffinelli, S. Broscheit, R. Gemulla, You CAN teach an old dog new tricks! on training knowledge graph embeddings, in: *ICLR*, OpenReview.net, 2020.
- [24] C. Demir, A.-C. Ngonga Ngomo, Clifford embeddings—a generalized approach for embedding in normed algebras, in: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, 2023, pp. 567–582.
- [25] M. Wang, L. Qiu, X. Wang, A survey on knowledge graph embeddings for link prediction, *Symmetry* 13 (2021) 485.
- [26] Y. Dai, S. Wang, N. N. Xiong, W. Guo, A survey on knowledge graph embedding: Approaches, applications and benchmarks, *Electronics* 9 (2020) 750.
- [27] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, *IEEE Transactions on Knowledge and Data Engineering* (2017).
- [28] M. Galkin, X. Yuan, H. Mostafa, J. Tang, Z. Zhu, Towards foundation models for knowledge graph reasoning, in: *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, OpenReview.net, 2024. URL: <https://openreview.net/forum?id=jVEoydFOl9>.