RESC: Relation Extraction by Sequence Compression

A. Kuzmenko, V. Kireev

The extraction of entities and their interrelationships constitutes a fundamental technique for the compressed representation of knowledge in graph form. This paper introduces an architecture designed for the extraction of relational triples through sequence transformation and compression. The proposed model is constructed upon a BERT-like encoder and incorporates two specialized decision heads. By employing a novel sequence compression algorithm, each head is tasked with a simplified classification problem. The RESC architecture operates without autoregression and is not susceptible to issues arising from overlapping relations, a limitation common in many prior approaches. Consequently, the method achieves high performance, including an F1-score exceeding 0.88 on the NYT11 benchmark, a primary standard for relational triple extraction.

Keywords—relational triple, neural network, natural language processing, transformers, knowledge graphs

I. INTRODUCTION

The extraction of structured information from natural language constitutes a fundamental objective within natural language processing. Decision-making systems reliant on textual data typically require input to be organized in a predefined, structured format. The knowledge graph has emerged as a highly expressive structure for representing knowledge derived from textual documents and is widely utilized in applications such as the fine-tuning of Large Language Models.

The primary technology for constructing knowledge graphs is relational triple extraction. A relational triple is a structured representation of a text fragment, typically a sentence, composed of a subject, an object, and a predicate defining the relationship between them. Subjects and objects are entities such as persons, organizations, or locations while the predicate denotes a specific relationship type (e.g., person-affiliation or organization-location).

Existing methodologies for the joint extraction of entities and relations are generally categorized into four paradigms: (1) pipeline methods, (2) tabular methods, (3) sequence labeling-based methods, and (4) sequence transformation-based methods.

This paper proposes a modified pipeline approach, implemented using a single pre-trained BERT model equipped with two specialized decision heads. A key contribution of our work is a novel sequence compression algorithm. This algorithm enables the relational triple extraction task to be decomposed into and addressed through a series of straightforward classification problems. We demonstrate the efficacy of our proposed model, achieving competitive performance on the established NYT11 benchmark dataset.

V. Kireev, PhD, National Research Nuclear University MEPHI, Moscow, Russia, vskireev@mephi.ru, https://orcid.org/0000-0001-6315-163X

A. Kuzmenko, Postgraduate student, National Research Nuclear University MEPHI, Moscow, Russia, andrey_kuzmenko2907@mail.ru, https://orcid.org/0009-0007-5900-6676

II. PROBLEM DEFENITION

The problem of relational triple extraction or joint entity and relationship extraction can be formulated as finding a set of triple sets $Y = \{(s_1, r_1, o_1), (s_2, r_2, o_2), \dots, (s_k, r_k, o_k)\}$ extracted from a text document $X = \{x_1, x_2, \dots, x_m\}$, as a result of solving the equation:

$$p(Y|X, \theta) = p(k|X) \prod_{i=1}^{K} p\left(Y_i | X, Y_{i \neq j}, \theta\right) \quad (1)$$

where $s_i = [x_{s_i}^{start}, \dots, x_{s_i}^{end}]$ – as entity defined as a subject; $o_i = [x_{o_i}^{start}, \dots, x_{o_i}^{end}]$ – as entity defined as a object; r_i – relationship between entities from a fixed set $r = \{r_1, r_2, \dots, r_n\}$; $p(k|x)$ – models the size of a set of relational triples; $p\left(Y_i | X, Y_{i \neq j}, \theta\right)$ – models a triple Y_i, provided that it is connected not only with the sentence, but also with other triples $Y_{i \neq j}$; θ – model parameters.

III. RELATED WORK

Existing approaches to relational triple extraction are typically categorized into four primary paradigms [1, 2].

Pipeline Methods. This group of approaches strictly decomposes the task into two sequential stages: first, the extraction of all candidate entities, followed by the identification of relations among the discovered entities. Representative works include [3-5]. The principal advantages of this scheme are the flexibility in selecting disparate components for each subtask and the capacity for modular debugging and error analysis, which is crucial for system development. However, decoupling the system introduces significant drawbacks. Foremost is the multiplicative propagation of errors: entity recognition mistakes in the first stage inevitably lead to relation identification errors. Furthermore, employing two distinct models increases computational overhead, slows inference, and complicates domain adaptation, as each component requires separate retraining.

Table-based Methods. This paradigm, first introduced in [6], formulates the task as populating a table where cells represent potential entity pairs and their labels denote

relations. Methods within this group are predominantly autoregressive [7-8], which constitutes a bottleneck for inference speed. An additional limitation is their frequent inability to handle multiple, overlapping relations within a single sentence.

Methods Based on Sequence Relabeling. The third paradigm transforms the triple extraction task into a sequence labeling problem via specialized tagging schemas. The seminal work is [9], with subsequent developments in [10-11]. These methods generally achieve superior accuracy compared to other paradigms under equivalent conditions and maintain relatively high inference speed.

Sequence Transformation Methods. The final paradigm encompasses methods that transform the input sequence directly into a set of output triples [12], including adaptations framed as set generation problems [13-14]. Recent interest has focused on leveraging Large Language Models (LLMs) for this purpose [15-19], due to their demonstrated few-shot and zero-shot capabilities. However, the quality of these generative approaches typically still lags behind that of dedicated sequence relabeling methods.

IV. METHOD

We propose a non-autoregressive approach for extracting relational triples from natural language texts. It includes three conventional stages: formation of candidates – stage, formation of pairs of entities; selection of candidates – filtering of deliberately incorrect candidates; classification relations – identification relations between entities. The scheme of the proposed method is illustrated in Figure 1.

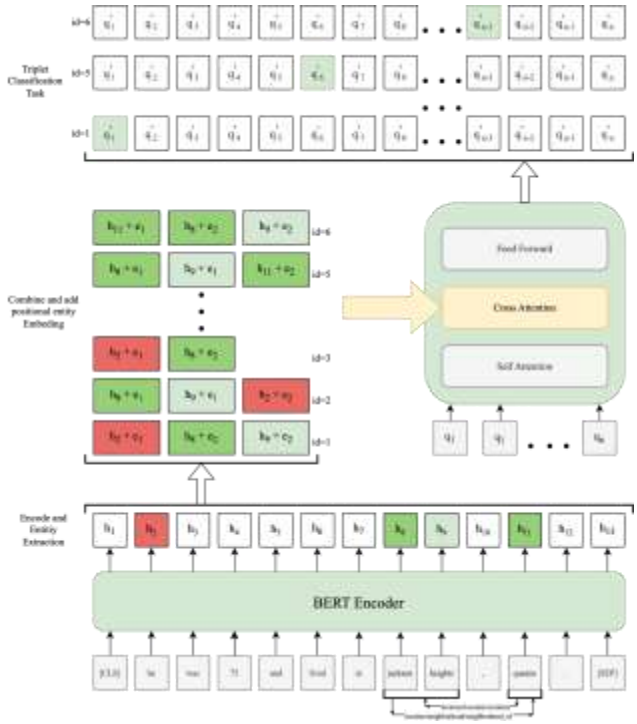

Fig. 1. Proposed RESC-architecture

Figure 1 illustrates the process of extracting two triplets containing the common entity "Jackson Heights". As demonstrated by the example, the application of the proposed approach helps to avoid the problem of overlapping relations. A detailed description of the mechanisms ensuring this advantage is provided in the subsequent sections of this paper.

A. Formation candidates

A sequence of text tokens $X = \{x_1, x_2, \dots, x_m\}$ is encoded by a pre-trained BERT-like model into a sequence of vector representations $H = \{h_1, h_2, \dots, h_m\}$. All possible substrings, candidate entities, are formed from the vector representations. Each entity is a potential subject $H^{sub} = \{h^{sub}_{start}, \dots, h^{sub}_{end}\}$ or object $H^{obj} = \{h^{obj}_{start}, \dots, h^{obj}_{end}\}$ of the relation. Each candidate triplet is a concatenation of a candidate subject and a candidate object $\{h^{sub}_{start}, \dots, h^{sub}_{end}, h^{obj}_{start}, \dots, h^{obj}_{end}\}$. We call this operation sequence compression. It allows us to extract the relation between these entities directly by solving the simplest classification problem.

An important part of sequence compression is the positional encoding of entities, since such reconstruction does not take into account the position of the vector sequence. We propose to take into account the location of entities using two trained auxiliary vectors e_1, e_2 transforming the compressed sequence $\{h^{sub}_{start} + e_1, \dots, h^{sub}_{end} + e_1, h^{obj}_{start} + e_2, \dots, h^{obj}_{end} + e_2\}$. Thus, for the subject "he" and the object "Jackson heights", the sequence of vector representations looks like this (Fig. 1) $\{h_7 + e_1, h_8 + e_2, h_9 + e_2$.

B. Selection candidates

A critical issue of greedy enumeration is issue with compound entities. We call an entity «compound» if it includes two or more tokens. In this case, the number of candidate entities will have complexity $O(SN)$, where S is the length of the text in tokens, and N is the maximum length of a substring. In this case, the number of combinations of candidate triplets will be calculated as the number of placements $n/(n-2)!$, where n is the number of found entities. It is easy to see that for $S \to N$ we get complexity $O(N^4)$. Such complexity motivates us to use an auxiliary entity identification module. It solves a simple problem, which is almost a complete analogue of the well-known problem of named entity extraction. We transform each vector h_i using a linear projection and solve the classical classification problem into three classes: 0 – is not part of the entity, 1 – the beginning of the entity, 2 – is part of the entity, but not the first element. As a result of solving this problem, we transform $O(N^4)$ into $O(N^2)$. In Figure 1 this operation is illustrated by highlighting the h_i vectors. Bright green (0) is the head of a correctly found entity, light green (2) is the continuation of the entity, bright red (1) is an erroneously found entity, no highlighting (0) is not an entity.

C. Classification of relations

To determine the type of relations, we use a technique inspired by works devoted to multimodality [20-21]. We initialize a set of trainable query vectors $Q = \{q_1, q_2, \dots, q_n\}$, where n is the number of relations. We pass these vectors to the transformer block, which is

connected by the cross-attention mechanism with the candidate triplets. Under the action of the cross-attention mechanism, the query vectors are transformed into signal vectors $Q' = \{q'_1, q'_2, \dots, q'_n\}$. For each signal vector, a binary classification problem is solved. We propose to use a single affine transformation for all types of relations. This approach eliminates the need for the autoregressiveness of the model and does not suffer from the problem of multiple relations, when several types of relations exist between one pair of entities.

We experiment with the classical pretrained encoder model [22], which contains almost the entire mass of trainable parameters. The sizes of the model and its components are given in Table 1.

TABLE I. DISTRIBUTION OF RESC PARAMETERS

Part	*Count*
All weigths	117M
BERT encoder	109M
Entity extraction module	2k
Relation extraction module	8M

V. TRAINING

We train all the model weights in one run, optimizing a combined loss function of the following form:

$$\mathcal{L} = \mathcal{L}_E + \lambda \mathcal{L}_R \quad (2)$$

The $\mathcal{L}_E$ component is responsible for solving the problem of identifying candidate entities. Which is a classic cross-entropy into three classes: 0 – not an entity, 1 – the beginning of an entity, 2 – the continuation of an entity.

$$\mathcal{L}_E = -\frac{1}{B_e * S} \sum_i^{B_e} \sum_j^{S} \log \frac{\exp(h_i^e)}{\sum_{k=0}^{2} \exp(h_k^e)} \cdot$$

$$\cdot 1[y_i^e \neq ignore\_index] * 1[y_i^e \neq pad\_index] \quad (3)$$

where $1[y_i^e \neq ignore\_index]$ – ignoring results on non-target labels; $1[y_i^e \neq pad\_index]$ – ignoring results on padded sequences; B_e – batch size; S – length of the maximum sequence in the batch.

The $\mathcal{L}_R$ component is responsible for solving the problem of determining relational triples from the available candidates. This is a binary multilabel classification problem, which can be written as follows:

$$\mathcal{L}_R = -\sum_i^{B_r} \sum_j^{N_i} \sum_k^{N_r} \{y_k^r \log \hat{y}_k^r + (1 - y_k^r) \cdot$$

$$\cdot \log(1 - \hat{y}_k^r)\}, \quad (4)$$

where B_r is the size of the text batch; N_i – is the number of placements, which can be calculated as $k/(k-2)!$, where k is the number of entities defined in the text; N_r – is the number of relations; y_k^r – is the label of class k: 0 is not a relation, 1 is a relation; $\hat{y}_k^r$ – is the estimate of the obtained values of the probability of belonging to class k.

We train the relation classification task only on combinations of gold entity labels. We assume that in the case of inference, an uncovered area of incorrect entities may be detected, but we assume that such an impact will be insignificant.

In our experiments, we use the NYT [23] public dataset [24] adapted for the task of extracting relational triples. The dataset contains 24 types of unique relations. It is divided into training, validation, and test parts, the descriptive characteristics of which are shown in Table 2.

TABLE II. NYT11 CHARACTERISTICS

Part	*Samples*	*Entities*	*Relations*
Train	56 196	177 461	121 450
Validation	5 000	15 923	10 848
Test	5 000	15 861	10 836

We train RESC in two stages. In the first stage, we train the weights of each subtask separately, freezing the overall encoder weights. We hypothesize that this allows the model to obtain a stable state in solving the entity retrieval and relation identification tasks without erasing the information about the text structure obtained in the pretraining stage.

The first stage training was performed until the f-measure of each task reached a plateau. On the NYT dataset, this was achieved in less than 10 epochs with a learning rate of 1e-4. Table 3 shows the values of the extraction metrics at the end of the first stage.

TABLE III. FIRST STAGE RESILTS

Part	*recall*	*precision*	*F*
Entity extraction	75.0	87.8	81.0
Relation identification	96.0	96.0	96.0
Joint entity and relationship extraction	72.0	80.0	76.0

In the second stage, we unfreeze the model encoder. We train all model weights for 10 epochs with a learning rate of 1e-4, then reduce the learning rate to 1e-5 for another 15 epochs. This training strategy allowed us to increase the f measure from 87.6 to 88.4.

Table 4 shows the results of the second stage training. Table 5 shows the results of the experiment of training all model weigths in one stage.

TABLE IV. SECOND STAGE RESULTS

Part	*recall*	*precision*	*F*
Entity extraction	93.6	96.0	94.8
Relation identification	96.5	97.3	96.9
Joint entity and relationship extraction	86.9	89.9	88.4

TABLE V. RESULTS OF TRAINING ALL MODEL WEIGTHS IN ONE STAGE

Part	*recall*	*precision*	*F*
Entity extraction	93.9	96.0	94.9
Relation identification	95.0	96.8	95.9
Joint entity and relationship extraction	86.0	89.3	87.6

We observed that the task of identifying relations of the reconstructed sequence is extremely simple, and the difference from entity extraction is especially noticeable at the end of the first stage. From Table 1 it is clear that the number of parameters of the head identifying relations is significantly greater than the number of parameters of the head extracting the entity. We experimented with increasing the parameters for the task of extracting entities to close values of the second head, but obtained little different results.

At all stages of training, we use the AdamW optimizer [25], which has proven itself well in practice. We show it and the other training hyperparameters in Table 6.

TABLE VI. TRAINING HYPERPARAMETERS

Hyperparameter	*Value*
$\overline{B_e}$	32
$\overline{B_r}$	32
$\overline{\gamma}$	1
Learning rate (first stage)	1e-4
Learning rate (second stage)	1e-4, 1e-5
B1, B2	0.9; 0.999
Weight decay	1e-2

We experimented with different initializations of query embeddings using context information, developing a context that could add information about the relation type. The context was encoded by the BERT encoder, and we used the embedding of a special token [CLS] as an initialization. However, as our experiment showed, we were unable to find a context that could statistically significantly increase the convergence of the head responsible for identifying the type of relations. We attribute this to the nature of very close relations in the NYT and the fairly high level of solving the problem of identifying relations from a compressed sequence.

VI. COMPARISON SYSTEMS

This study presents a comparative analysis of existing methodologies for relational triple extraction, evaluated on the publicly available NYT dataset. The proposed RESC architecture achieves state-of-the-art performance, demonstrating accuracy metrics on par with contemporary baseline models. However, unlike BERT-based systems, which are traditionally employed for their high extraction quality, RESC offers a distinct computational advantage characterized by a complexity of $O(K^2)$ compared to $O(N^2)$, where K denotes the number of candidate entities and N represents the input sequence length. Given that, in practical applications, K is substantially smaller than N, this translates to a marked reduction in computational overhead.

Moreover, a key architectural innovation of RESC lies in its efficient utilization of the encoder. The model performs the computationally expensive encoding operation only once to jointly address both entity recognition and relation identification subtasks. In contrast, reference BERT-based solutions typically require two separate encoder passes for these steps, leading to a proportional increase in both temporal and hardware resource consumption. By minimizing the number of forward passes, RESC significantly diminishes the overall computational burden without degrading extraction performance.

TABLE VII. COMPARISON SYSTEMS

	recall	*precision*	*F*
CopyRE multi decoder [12]	56.6	61.0	58.7
GraphRel 2p [26]	60.0	63.9	61.9
OrderCopyRE [27]	67.2	77.9	72.1
CasRel$_{Random}$ [11]	75.7	81.5	78.5
CasRel$_{BERT}$ [11]	89.5	89.7	89.6
TPLinker$_{BERT}$ [28]	92.5	91.3	91.9
PRGC$_{Random}$ [29]	82.3	89.6	85.8
PRGC$_{BERT}$ [29]	91.9	93.3	92.6
ReLiK$_{BERT}$ [30]	92.9	93.2	93.1
RESC	86.9	89.9	88.4

The empirical results, detailed in the accompanying table, confirm that the RESC architecture delivers accuracy comparable to existing approaches while exhibiting substantially higher computational efficiency. This efficiency is particularly critical in resource-constrained environments, for real-time processing of large-scale data, or when deploying models on edge devices with limited computational capacity. Consequently, the proposed solution establishes an optimal equilibrium between high model accuracy and practical deployability.

VII. CONCLUSIONS

In the article, we present a new pipeline architecture for extracting relational triples from texts in the natural language of RESC, based on the sequence reconstruction mechanism proposed in the same work, which allows you to extract relational triples directly. The proposed system is implemented on the basis of a single BERT-like encoder, which minimizes the main disadvantages of pipeline systems, such as error multiplicativity and performance. The

experimental results showed a quality comparable to state-of-the-art methods on the main benchmark NYT.

REFERENCES

[1] Kuzmenko A.V., Kireev. V. S. Classification of methods for extracting relational triples from natural language texts. Proceedings of the XXV international scientific and technical conference Neuroinformatics-2023, 2023, pp. 302-311. (in Russian)

[2] Shang, et al. Relational Triple Extraction: One Step is Enough. Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22). 2022, pp. 4360-4366.

[3] Zenelnko, D., Aone, C., Richardella, A., et al. Kernel methods for relation extraction. Journal of Machine Learning Research. 2003. Vol. 3, pp 1083-1106.

[4] Chan, S., Y., and Roth, D. Exploiting syntactico-semantic structures for relation extraction. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011. pp 551-560.

[5] Zhong, Z., and Chen, D., A. A frustratingly easy approach for entity and relation extraction. Proceedings of the 2021 Conference of the North American Chaper of the Association for Computational Linguistics: Human Language Technologies. 2021. DOI: 10.18653/v1/2021.naacl-main-5.

[6] Miwa, M., and Sasaki, Y. Modeling Joint Entity and Relation Extraction with Table Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014, pp 1858-1869.

[7] Gupta, P., Schutze, H. and Andrassy, B. A Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction. Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. 2016, pp. 2537-2547.

[8] Zhang, M., Zhang, Y., Fu, G. End-to-End Neural Relation Extraction with Global Optimization. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. 2017, pp. 1730-1740.

[9] Zheng, S., et al. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. 2017, vol. 1: Long Papers, pp. 1227-1236.

[10] Dai, D., et al. Joint Extraction of Entities and Overlapping Relations Using Position-Attentive Sequence Labeling. Proceedings of the AAAI Conference on Artificial Intelligence. 2019, pp. 6300-6308.

[11] Wei, Z., et al. A Novel Cascade Binary Tagging Framework for Relational Triple Extraction. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020, pp. 1476-1488.

[12] Zeng D., Zhang, H. and Liu, Q. CopyMTL: Copy Mechanism for Joint Extraction of Entities and Relations with Multi-Task Learning. Proceedings of the AAAI Conference on Artificial Intelligence. 2020, pp. 9507-9514.

[13] Sui, D., Zeng, X., et al. Joint entity and relation extraction with set prediction networks. IEEE Transactions on Neural Networks and Learning Systems. 2023, DOI: 10.1109/TNNLS.2023.3264735.

[14] Wen, X., et al. End-to-end entity detection with proposer and regressor. Neural Proceedings Letters. 2023, doi: 10.1007/s111063-023-11201-8.

[15] Hu, Y., Ameer, I. and Zou, X. Zero-shot Clinical Entity Recognition using ChatGPT. 2024, DOI: arxiv-2303.16416.

[16] Yuan, C., Xie, Q. and Ananiadou, S. Zero-shot Temporal Relation Extraction with CharGPT. 2023, DOI: 10.48550/arxiv.2304.05454.

[17] Feng, P., Wu, H. and Yang, Z. Leveraging Prompt and Top-K Predictions with ChatGPT Data Augmentation for Improved Relation Extraction. Applied Sciences (ISSN 2076-3417). 2023, Vol 13(23).

[18] Dagdelen, J., Dunn, A., Lee, S., et al. Structured information extraction from scientific text with large language models. Nature communications. 2024. Vol. 15, No 1418.

[19] Liang Xu, Changxia Gao, Xuetao Tian. Domain-control prompt-driven zero-shot relational triplet extraction. Neurocomputing. 2024 Vol. 574. Doi: 10.1016/j.neucom.2024.127270.

[20] Junnan Li, Dongxu Li, Silvio Savarese, Steven Hoi. BLIP-2: bootstrapping language-image Pre-Training with frozen image encoders and large language models. Proceedings of the 40th International Conferece on Machine Learning, Honolulu, Hawaii, USA. PLMR 202, 2023.

[21] Hang Zhao, Yifei Xin, Zhesong Yu, et al. SLIT: boosting audio-text pre-training via multi-stage learning and instruction tunning. Arxiv:2402.07485v2.

[22] Jacob Delvin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: pre-training of deep bidirectional transformer for language understanding. Arxiv:1810.04805.

[23] Sebastian Riedal, Limin Yao, Andrew McCallum. Modeling relations and their mentions without labeled text. In Joint Euripean Conference on Machine Learning and Knowledge Discovery in Databases, pp. 148-163.

[24] Extracting relational facts by an end-to-end neural model with copy mechanism. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. vo1. 1, pp 506-514.

[25] Ilya Loshchilov, Frank Hutter. Decoupled weight decay regularization. Published as a conference paper at ICLR 2019. Arxiv:1711.05101v3.

[26] Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1409–1418, Florence, Italy. Association for Computational Linguistics.

[27] Xiangrong Zeng, Shizhu He, Daojian Zeng, Kang Liu, Jun Zhao. Learning the extraction order of multiple relational facts in a sentence with reinforcement learning. Proceedings of the 2019 Conference on Empirical Methods in Natural Lanugage Processing and the 9th International Joint Conference on Natural Language Processing, pp. 367-377.

[28] Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, , Limin Sun. TPLinker: Single-stage Joint Extraction of Entities and Relations Through Token Pair Linking. Proceedings of the 28th International Conference on Computational Linguistics, pages 1572–1582 Barcelona, Spain (Online), December 8-13, 2020.

[29] Hengyi Zheng, Rui Wen, Xi Chen et al. PRGC: potential relation and global correspondence based joint relational triple extraction. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pp. 6225–6235.

[30] Riccardo Orlando, Pere-Lluís Huguet Cabot, Edoardo Barba, Roberto Navigli. ReLiK: Retrieve and LinK, Fast and Accurate Entity Linking and Relation Extraction on an Academic Budget. Proceedings of the Associtation for Computational Linguistics ACL 2024. Arxiv:2408.00103.