A glimpse into Open Information Extraction with Large Language Models

Amrita H Nair [1] [2]

[1]Universität des Saarlandes, Saarbrücken, Germany [2]Univerzita Karlova, Prague, Czech Republic

Introduction

- This poster is based/inspired on the work by Wang et al., [2]
- Large Language Models(LLM's) store linguistic and relational information.
- The knowledge gained by LLM's can be exploited and extracted to perform partially unsupervised open information extraction.
- Extracting n-ary representations from text in an unsupervised manner helps in many tasks, such as building knowledge bases and knowledge graph construction.

Large Language Models

Large Language Models(LLM's) are models that are trained on large amounts of un-annotated and unlabelled data in an unsupervised fashion. Training is performed using different objectives like cloze task (masked language models), next sentence prediction, replaced token detection, sentence-order prediction task and so on. Due to the nature of the training and the amount of data provided, the model is able to perform many downstream tasks with relative ease. This prompted the rise of different methods to leverage this capability, fine-tuning being one of the most popular ones. Many different language models are freely available in the open-source domain.

BERT(Bidirectional Encoder Representations from Transformers) as the name suggests, "passes" over each sentence twice, in the forward and backward direction. The objectives used while training the model are masked word prediction where one word in the sentence is masked and the model is expected to predict the word from the context that it learns by viewing the sentence from both directions. The other task that BERT is trained is NSP(next sentence prediction) where the model is primarily a classification task where the model is expected to predict whether a pair of sentences follow one another.

Open Information Extraction

Open Information Extraction(OIE) is a task in NLP which involves generating a structured n-ary representation of text to facilitate various functions like knowledge graph construction. OIE facilitates the retrieval of relevant entities without the need for pre-defined relations. The relations extracted are sentence spans, the entities generally being noun-phrases. An example would be: `Sheela was born in Chennai in the year 1990`. The triples extracted would be "Sheela" - "born in" - "1990" and "Sheela" - "born in" - "Chennai". Intuitively, a simple sentence like the example above results in 2 triples. However, complex sentences may contain multiple triples.

Using LLM's for OIE attempts to leverage the relational information contained within the model.

DeepEx: Zero-Shot Open Information Extraction with LLM's

Wang et al., [2] proposed an interesting approach to extract relation phrases between any two entities in a sentence. Here, language models are treated as **zero-shot information extractors.**

Since LLM's seem to store relational information, the inspiration for this paper was to leverage this relational knowledge. The authors utilize the power of the attention mechanism. It deals with enhancing the model's ability to selectively focus on relevant parts of the input sequence when generating an output.

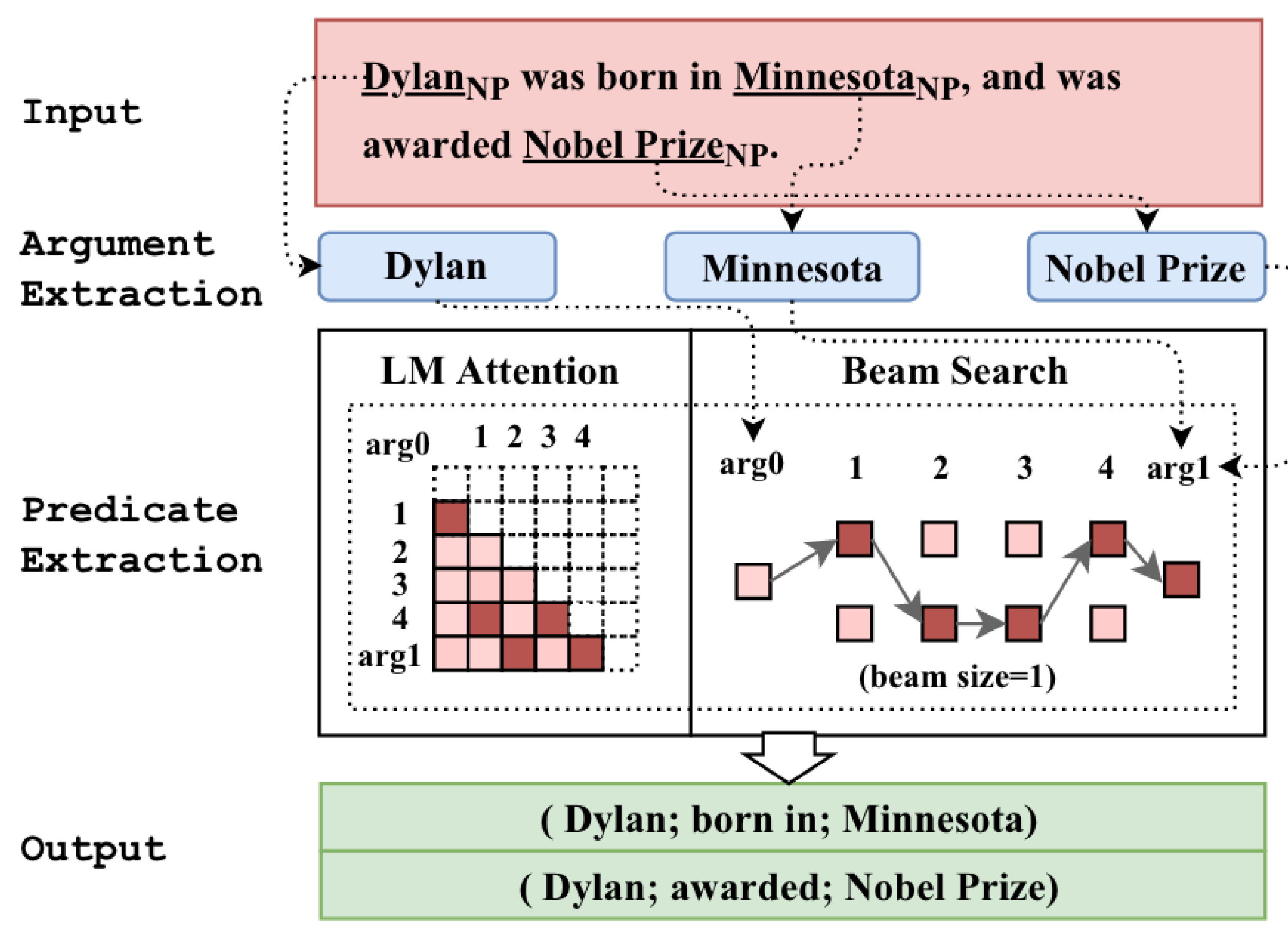
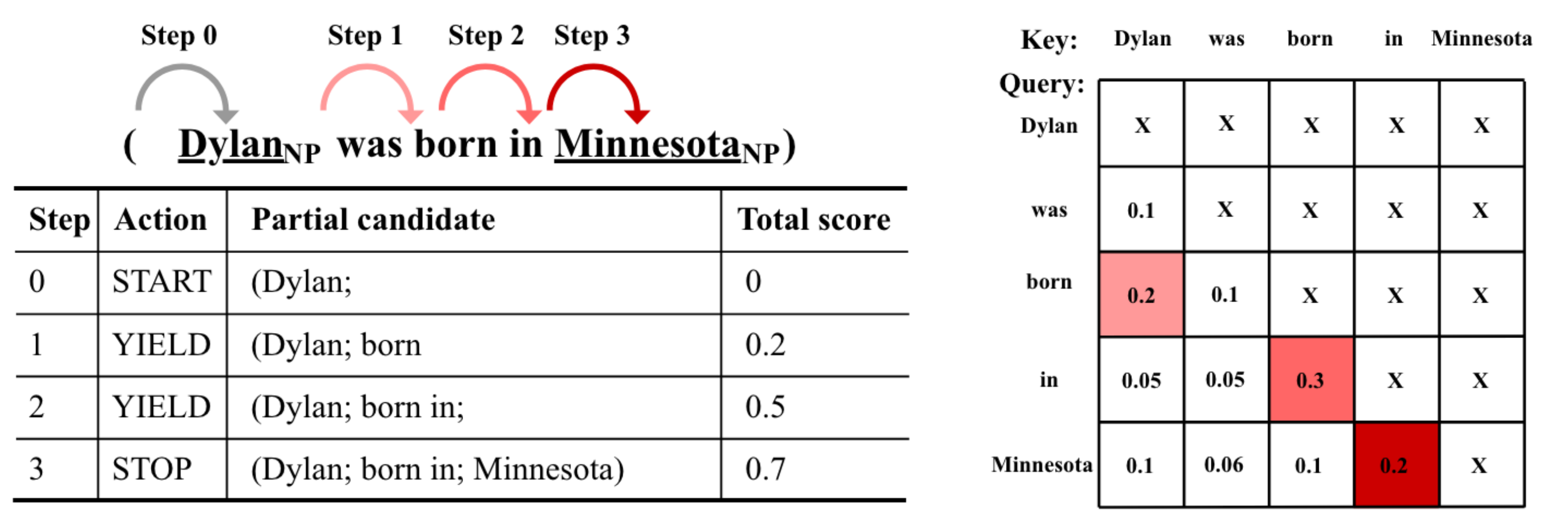


Figure 1. Wang et al., [3]

The architecture of this approach is shown in figure 1 above. Noun phrases are extracted using Spacy. Each pair of noun phrases is treated as an argument pair for which the predicate needs to be extracted. The pairs are considered only in one direction at a time, but the alogrithm is run in both directions, since some triples are often in reverse order. The predicate(relation phrase) is extracted using the **attention scores** for the combination of words between the noun phrases with the maximum score which gives the output.

An example is provided in figure 2. The example phrase `Dylan was born in Minnesota and was awarded Nobel Prize` is used where "Dylan", "Minnesota" and "Nobel Prize" are the noun phrases that have been extracted by Spacy. Consider the sub-phrase `Dylan was born in Minnesota`. Here "Dylan" and "Minnesota" are the noun phrases whoose predicate has to be extracted. The right part of figure 2 shows the attention matrix for this sub-phrase. Let "Dylan" be `ARG1` and "Minnesota" be `ARG2`. In the attention matrix, the word with the highest attention score while attending to `ARG1` is "born". This word is added to the predicate phrase and the next step is to look at the attention scores for the word "born". The next word is "in" with an attention score of 0.3. "in" is added to the predicate phrase and the total score for this computation is increased by 0.3. The algorithm can be followed using the table that is in the right in figure 2. The algorithm stops upon encountering `ARG2` at which point the total score is 0.7.

Computing a score for every potential sequence proves to be computationally expensive, especially when dealing with long sequences. Consequently, the exhaustive search approach becomes intractable due to its impracticality.



Step	Action	Partial candidate	Total score
0	START	(Dylan;	0
1	YIELD	(Dylan; born	0.2
2	YIELD	(Dylan; born in;	0.5
3	STOP	(Dylan; born in; Minnesota)	0.7

Key: / Query:	Dylan	was	born	in	Minnesota
Dylan	X	X	X	X	X
was	0.1	X	X	X	X
born	0.2	0.1	X	X	X
in	0.05	0.05	0.3	X	X
Minnesota	0.1	0.06	0.1	0.2	X

Figure 2. Wang et al., [3]

To tackle this issue, the authors adopted beam search as an approximate strategy to efficiently explore the search space. Beam search operates by maintaining the k-best candidates, enabling more manageable computation and reducing the complexity of the search process.

Ranking: In the ranking stage, the task-specific relevance of triples is determined using a ranking model trained on a relational corpus that is not specific to the task at hand. In the generating stage, k candidate triples are generated for each argument pair. However, it is crucial to note that the sequences within these candidates pertain not only to the relational aspect but also to the argument pairs themselves. The primary objective of the ranking stage is to identify triples that explicitly convey the relational information between the argument pair, as this information holds significant importance for Open Information Extraction (OIE).

The Attention Mechanism

The attention mechanism enables the model to selectively attend to different parts of the input sequence. The process of calculating the attention matrix involves the following steps. The input sequence is transformed into query, key, and value vectors using linear transformations. The similarity between the query and key vectors is then computed through dot products. To stabilize training, the dot products are scaled by dividing them by the square root of the key vector dimension. The scaled dot products are passed through a softmax function, producing attention weights. These weights determine the contribution of each value vector to the final output. The attention matrix is formed by the pairwise attention weights between the query and key vectors. Next, the attention weights are applied to the value vectors using element-wise multiplication, and the results are summed. This step generates a context vector that captures relevant information based on the attention weights. The context vector is then utilized in subsequent layers for further processing or serves as the output for the task at hand. By calculating the attention matrix, the Transformers library enables the model to capture dependencies and focus on significant segments of the input sequence.

Results

The framework was tested on different popular OIE datasets like OIE2016 [1], NYT, WEB, PENN and compared against various systems that are listed above. Currently, DeepEx is SOTA for OIE.

Open Information Extraction (F1 and AUC)

	OIE2016		WEB		NYT		PENN	
ClausIE	58.8	37.6	44.9	40.1	29.6	22.9	34.6	28.4
Open IE4 [4]	59.6	41.7	55.7	40.5	38.3	24.0	42.6	28.1
PropS	55.6	33.8	58.9	48.0	37.2	22.1	39.1	27.7
RnnOIE	67.0	44.5	58.1	43.5	28.3	10.4	34.5	18.3
MAMA	36.6	12.8	54.3	30.3	32.9	9.4	33.0	9.0
DEEPEX (Zero-Shot)	**72.6**	**58.6**	**91.2**	**82.4**	**85.5**	**72.5**	**88.5**	**81.5**

Figure 3. Wang et al., [2]

Drawbacks and future work

The approach used by the framework for OIE is undoubtedly interesting, however, there is room for improvement. The authors performed error analysis towards this end.

- The framework uses Spacy to extract NP-pairs. According to the authors, 33.3% of the errors were caused due to Spacy not extracting the NP's correctly. Choosing a better NP extraction method or algorithm would probably help in fixing this.
- Another way of improving results could be to choose better noun pairs. In the current approach, each pair is considered which wastes time and computing power. A number of linguistic constraints could be used to choose better pairs. Each pair could have a "weight" that would factor into the ranking.
- Sometimes, a noun can be a predicate. This happens in 12% of the error cases.
- Another error potential is that in 10% of the cases, the predicate is not between the two nouns, but on the outside.

References

Figure 4. All references can be found by scanning the QR code

[1] Gabriel Stanovsky and Ido Dagan.
Creating a large benchmark for open information extraction.
In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2300–2305, 2016.

[2] Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song.
Zero-shot information extraction as a unified text-to-triple translation.
arXiv preprint arXiv:2109.11171, 2021.

[3] Chenguang Wang, Xiao Liu, and Dawn Song.
Ielm: An open information extraction benchmark for pre-trained language models, 2022.