Natural Language Processing Leveraging Entity-Aware Representations

José Luís Mourão Malaquias
IST, University of Lisbon
Lisbon
Portugal
jlmalaquias@malaquias.eu

Abstract

Recent Pre-trained Language Models (PLM), based on self-attention and the Transformer neural architecture, produce a contextualized representation of words. Leveraging such representations in other neural networks enabled state-of-the-art results in the corresponding downstream Natural Language Processing (NLP) tasks. A new line of research focuses on studying how to incorporate named entity components within a PLM. The motivation is that named entities convey essential information that can further improve the performance of a model in a NLP task when taken into account. However, current entity-focused models face limitations, such as the inability to process long inputs. This is especially noticeable in recent complex tasks, which demand a good understanding of long text sequences. In this article, we propose Long-LUKE, a model based on the entity-aware LUKE PLM capable of processing long text sequences. In particular, it obtains interesting results on four well-known datasets: Open Entity and FIGER (entity typing), and TACRED and ReDocRED (relation classification).

***Keywords*—** Named Entities, Long Documents, Transformers

1 Introduction

Recently, Pre-trained Language Models (PLM) such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), based on self-attention and the Transformer neural architecture, outperformed previous approaches to modelling sequences of words within Natural Language Processing (NLP) applications, such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks. The self-attention mechanism proposed in the Transformer architecture (Vaswani et al., 2017) was a disruptive idea, as it allowed the modelling of dependencies without regard to their distance in the input or output sequences. Another disruptive idea was to treat words and entities in a given text as different types of tokens (Yamada et al., 2020; Baldini Soares et al., 2019). Properly representing entities (Qin et al., 2021; Joko et al., 2021) is especially relevant, as many tasks (i.e., Question Answering (QA), Machine Translation, Chatbots) involve identifying these critical elements in the text, like names of people, places, or brands. Furthermore, their performance may benefit from the awareness of the representation of entities as it is mainly from those entities that we can locate the subject and context of a document (Blei et al., 2003).

Specifically, we show an example in Figure 1 to understand why it is relevant to be aware of entities (highlighted words) and their relations within a text. In the example, in order to understand "Where is the Republic of Ireland" we have to consider the clues jointly: (1) The "United Kingdom" is located in "Europe" in sentence 1; (2) The "United Kingdom" includes part of the "island of Ireland" in sentence 2; (3) "Northern Ireland" shares a land border with "Republic of Ireland" in sentence 3; (4) "Northern Ireland" is one of the countries of "United Kingdom" in sentence 6. From the example we learn that in order to understand an entity we must consider its relations to other entities comprehensively. In the example, the entity "United Kingdom" appearing in sentences 1, 2, 4 and 6 help us find the answer. To understand "United Kingdom", we should consider all its connected entities and diverse relations among them, and to understand the relations between pairs of entities, we need to comprehend complex reasoning patterns in the text, i.e., we infer the "Republic of Ireland" is in "Europe" as it shares borders with a country of "United Kingdom", which is located in "Europe". Through the relations between entities, we can capture an entity's true meaning. For example, in sentence 5, with a lack of context, the entity "English channel" could have multiple meanings. A model with a good comprehension of the

United Kingdom

[1] The United Kingdom is a sovereign country in north-western Europe. [2] The United Kingdom includes the island of Great Britain, the north-eastern part of the island of Ireland, and many smaller islands within the British Isles. [3] Norther Ireland shares a land border with the Republic of Ireland. [4] The United Kingdom is surrounded by the Atlantic Ocean, with the North Sea to the east, the English Channel to the south and the Celtic Sea to the south-west. [5] The Irish Sea separates Great Britain and Ireland. [6] The United Kingdom consists of four countries: England, Scotland, Wales and Northern Ireland. [7] The capital and largest city is London.

Figure 1: An example for a document "United Kingdom", in which all entities are underlined. We highlight the important entities and relations to find out "Where is the Republic of Ireland".

text performs better on tasks that rely on context, especially if we need to compare multiple relations along a multiple-sentenced document to perceive such knowledge. However, most PLMs do not consider entities and their relations among multiple entities at the document level, whose context involves complex reasoning patterns.

A recently proposed entity-aware model (Yamada et al., 2020), which focuses on explicitly model entities, significantly improved the performance on entity-related tasks, such as entity typing, relation classification, named entity recogni-

tion, and others.

We will focus on studying which aspects of its implementation can benefit from recent studies, making Language Understanding with Knowledge-based Embeddings (LUKE) able to handle larger textual inputs (Beltagy et al., 2020; Zaheer et al., 2020) due to the quadratic cost of the attention mechanism.

In this context, we present Long-LUKE, an extended model from LUKE capable of handling longer sequences of text. Additionally, our model presents a different self-attention pattern from LUKE. Even though the entity-focused attention operations remain the same, the textual attention pattern follows the Longformer (Beltagy et al., 2020) mechanism that scales linearly with sequence length. Following prior work on entity-oriented transformers, we evaluate Long-LUKE on various downstream tasks. Our model achieves the same performance for short-sequenced datasets as all baseline models. Also, it obtains exciting results on both DocRED (Yao et al., 2019) and ReDocRED (Tan et al., 2022), both relation classification datasets which include long documents.

The remainder of this paper is organized in the following manner: Section 2 describes related existing work, Section 3 details the proposed approaches, while Section 4 presents the experimental evaluation methodology and the obtained results. Section 5 summarizes our contributions and discusses future improvements to our approach.

2 Related Work

This section discusses approaches that relate to the proposed work. NLP frequently involves tasks regarding entities, such as: Entity Typing (ET), Named Entity Recognition (NER), QA and Relation Classification (RC), just to name a few. A good representation of entities is a decisive part of a good model and can be challenging.

Although transformer-based representations, i.e., produced by BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), correspond to good word representations, they are not very effective in representing entities because Named Entities are "open" linguistic constructs, meaning they are difficult to model (i.e., it is not possible to have all existing names of people in a training dataset). Even though the Transformer self-attention mechanism captures complex relationships between words, it cannot establish the relationship between entities when they are composed of more than one word. Although recent approaches (Qin et al., 2021; Hsieh, 2017) proposed improving entity representations by aiming to obtain a deep understanding of the entities and their relations in text, most models cannot model entity representations effectively.

2.1 Language Understanding with Knowledge-based Embeddings

Language Understanding with Knowledge-based Embeddings (LUKE) (Yamada et al., 2020) is a language model based on RoBERTa (Liu et al., 2019) designed to improve the performance of entity-related tasks. This approach adds entity embeddings as well as a new entity-aware self-attention mechanism. LUKE learns how to compute entity representations and deliver them in the model output. The model is pre-trained through a standard Masked Language Model (MLM) task and a new pre-training task that focuses on learning dedicated entity-based representations. The model learns entity representations by specifically targeting entities on a separate MLM task. The dataset used to train the novel task is extracted from a large amount of Wikipedia data, where the entities are the hyperlinks present on the Wikipedia website. A unique attribute of LUKE is the treatment of words and entities as independent tokens, representing each one. Regarding the self-attention mechanism of a model, the idea is to relate tokens using attention scores. The traditional attention mechanism does not expect two different kinds of tokens. Therefore Yamada et al. (2020) proposes a new implementation which is aware of what kind of tokens the mechanism is dealing with when it is computing the attention score between the pair of tokens.

The idea mentioned above was implemented through an entity-aware query mechanism that handles query matrices with different weights for each possible combination of pairs of tokens. LUKE delivered state-of-the-art performances in 5 entity-related tasks, namely Entity typing, Relation classification, Named Entity Recognition, and QA.

Although LUKE outperformed some models in tasks, it also encountered some drawbacks. As LUKE highly depends on datasets that label their entities, if a new entity that does not belong in the vocabulary is introduced, the model has to fall back to the [UNK] entity representation, damaging its performance in the target NLP task.

Another LUKE limitation is the inability to handle long sequences of text as its maximum sequence length is only 512 tokens. This is due to the self-attention operation, which scales quadratically with the sequence length. Recently, several studies proposed similar methods to address this limitation (Zaheer et al., 2020; Rabe and Staats, 2021; Jiang et al., 2020; Kitaev et al., 2020). A possible solution for this issue would be to borrow technical ideas from Longformer (Beltagy et al., 2020) to reduce the quadratic dependency to linear.

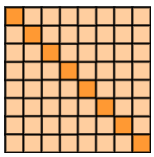
2.2 The Long-Document Transformer

Longformer (Beltagy et al., 2020) is a model containing an approach to reduce the time and memory complexity of the self-attention mechanism. It proposes a new self-attention pattern composed of several elements that define the pairs of tokens matched in the attention calculation. The **sliding window attention pattern** (Figure 2b) applies a fixed-size window surrounding each token. Several of the aforementioned windowed attention layers are stacked to generate a broad receptive field. Each token attends to $\frac{1}{2}w$ tokens on each side, where w stands for the defined window size. This pattern causes $\mathcal{O}(n \times w)$ complexity, which is linear to the input sequence length n. Subsequently, Longformer introduces size dilation gaps d to the window (Figure 2c). In the Longformer implementation, the authors varied the size of the windows across layers. Notably, small window sizes are used for lower layers. The window size increases as we move to higher layers, enabling the top layers to learn a higher-level representation of the entire sequence while having the lower layers capture local information. Additionally, it also provides a balance between efficiency and performance. Lower layers do not use dilated sliding windows to maximize their capacity to learn the local context.

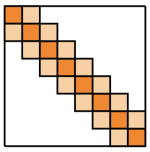
The **global attention pattern** (Figure 2d) is added on predefined input locations in order to learn task-specific representations, such as [CLS], [UNK], and [SEP]. In this pattern, the attention operation is symmetric, meaning a token with global attention attends to all tokens across the sequence and vice versa. The number of tokens covered by global attention is small relative to n. Therefore the complexity of the combined attention patterns is still $\mathcal{O}(n)$. The **combined attention pattern** is shown in Figure 2d. It is possible to observe that it corresponds to the combination of the previous patterns. Converting a Transformer into a Long-Transformer is still effective even when applied only during finetuning, which avoids re-training the PLM from scratch.

3 Long-LUKE

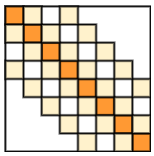
Following recent trends in NLP, we present Long-LUKE. This new entity-based architecture combines the concepts of



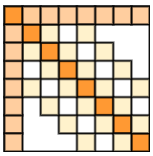
(a) Full n^2 attention



(b) Sliding window



(c) Dilated Sliding window



(d) Global + Sliding window

Figure 2: Long-Transformer attention patterns.

using complex entity representations from LUKE and modifying the attention pattern (Beltagy et al., 2020) to enhance the model's performance on long sequenced documents. For the most part, our model maintains the same architecture of the original LUKE implementation. Particularly, Long-LUKE has both a Transformer (Vaswani et al., 2017) component from BERT, since it treats both words and entities as independent tokens, and Long-Transformer (Figure 3) characteristics.

3.1 Input Representation

A unique attribute that Long-LUKE shares with LUKE is the treatment of words and entities as independent tokens, as seen in Figure 3. The input representation of a token, which is either a word or an entity, is obtained by applying the following embeddings:

1. Token embedding: represents the type of token(i.e., A for words or B for entities);
2. Positional embedding: represents the token's absolute position in a word sequence. If an entity includes more than one token, its position embedding D is calculated by averaging the embeddings of the corresponding token positions C that form the entity;
3. Entity type embedding (e): means that the token is an entity.

The input representation of a word (i.e., left side of the input sequence in Figure 3) is given by adding the token and position embeddings, whereas to obtain the entity representation (i.e., *Buenos Aires*$_{[MASK]}$, *Argentina*) we add all the embeddings. As the masked input entity relates to multiple token entities (*i.e., Buenos Aires*), we average the corresponding position embeddings to encode its position. For example, the entity *Buenos Aires* occupies positions 2 and 3 of the input word sequence. Therefore to obtain the positional entity embedding, we have $\frac{D_2+D_3}{2}$. Similarly to previous model implementations (Devlin et al., 2019; Liu et al., 2019), LUKE makes use of special tokens: [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token.

3.2 Attention Pattern

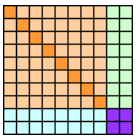
An essential contribution of Long-LUKE is that it extends LUKE using our *Long entity-aware* self-attention mechanism by merging the self-attention patterns described by Vaswani et al. (2017) and Beltagy et al. (2020). As our model is aware of what kind of tokens it is dealing with when computing the attention score (e_{ij}) between the pairs of x_i and x_j tokens, we developed our attention mechanism such that the attention scores where both tokens are words (e_{ww}) are computed through Longformer methodology, maintaining the LUKE operations for the remaining cases (e_{we}, e_{ew}, and e_{ee}). The attention pattern of $w2w$, $w2e$, $e2w$, and $e2e$ are illustrated in Figure 4 by the colours orange, green, blue and purple, respectively. Formally, the attention mechanism is computed as follows:

$$\mathrm{e}_{ij} = \begin{cases} \text{Sliding Window Attention , if } x_i \text{ and } x_j \text{ are words} \\ \mathbf{K}x_j^T\mathbf{Q}_{w2e}x_i \text{ , if } x_i \text{ words and } x_j \text{ entity} \\ \mathbf{K}x_j^T\mathbf{Q}_{e2w}x_i \text{ , if } x_j \text{ words and } x_i \text{ entity} \\ \mathbf{K}x_j^T\mathbf{Q}_{e2e}x_i \text{ , if } x_i \text{ and } x_j \text{ are entities} \end{cases}, \quad (1)$$

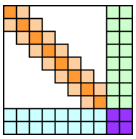
where $\mathbf{Q}$, $\mathbf{Q}_{w2e}$, $\mathbf{Q}_{e2w}$ and $\mathbf{Q}_{e2e}$ are query matrices, and where $\mathbf{K}$ and $\mathbf{V}$ are respectively the key and value matrices.

It is beneficial to use the information of entity tokens when computing the attention scores. However, at the same time, we would like to avoid the quadratic attention complexity attached to long text sequences (Figure 4a). In the LUKE implementation, each kind of attention score between pairs of tokens is computed separately due to each having its query and key matrices, and only then are the scores concatenated and fed into a *Softmax* and multiplied by the V matrix to obtain the final attention score. With this in mind, we acknowledge that the attention scores associated with entities have at most $\mathcal{O}(n)$ complexity for a sequence of n tokens. It would be beneficial to replace the word-to-word attention pattern, which causes $\mathcal{O}(n^2)$ complexity. This implementation does not modify the attention pattern of the entity-related scores. Consequently, we used a non-dilated sliding window attention pattern from Longformer (Figure 4b - orange), reducing the complexity to $\mathcal{O}(n \times w)$. In the context of our combined attention pattern (Figure 4b), we considered the entity-related attention patterns as "global attention" in the sense that the attention operation is symmetric, meaning a fixed entity token attends to all tokens across the sequence, and all tokens in the sequence attend to it. This global attention is more dynamic than Long-Transformer's as a greater range of tokens appears several times in the text. We hypothesize that such global attention improves the ability of our model to deal with entity-related tasks. Figure 4b displays an example of our combined entity-aware sliding window attention with "global attention" on entity tokens.

The main advantage of Long-LUKE, as displayed in Figure 3, is the ability to maintain the input representation and the architecture and enable it now to receive longer sequences of text in the same tasks. We built different approaches to address the Entity Typing and Relation Classification tasks using the entity representations from Long-LUKE.



(a) LUKE attention.



(b) Long-LUKE attention

Figure 4: Proposed configuration of attention patterns.

4 Experimental Evaluation

In this section, we will start by introducing, in Section 4.1, the datasets we used to perform the experiments. Next, we present the experimental results of our model in the datasets in Section 4.3, along with their corresponding metrics. Furthermore, we also provide a comparison with previous methods and against ablated versions of the proposed approaches.

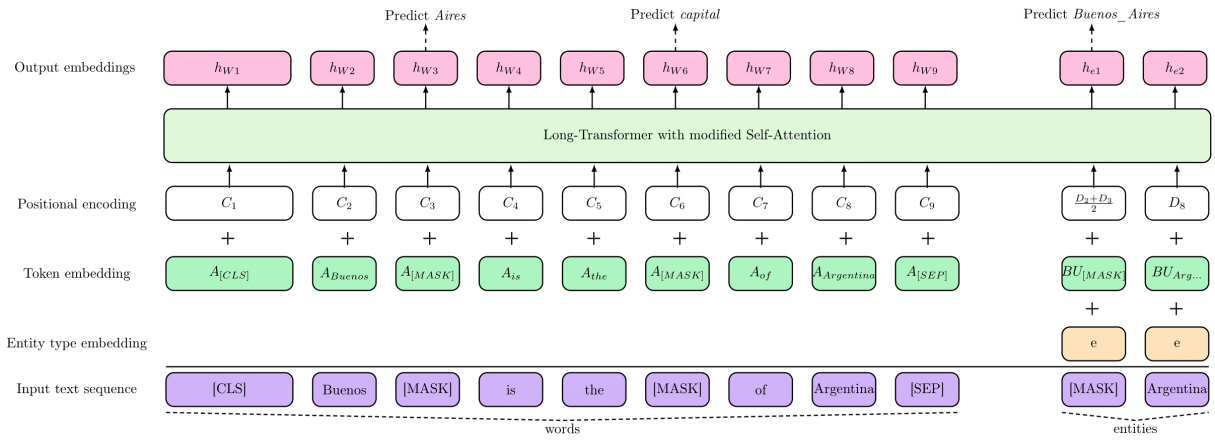


Figure 3: Input representation from Long-LUKE

4.1 Datasets

To evaluate the performance of our model in different tasks and domains, we relied on a wide variety of datasets used in previous studies: two entity typing datasets, namely FIGER (Ling and Weld, 2012) and OpenEntity (Choi et al., 2018); TA-CRED (Zhang et al., 2017), an intra-sentence relation classification dataset; and a A Large-Scale Document-Level Relation Extraction Dataset (DocRED) (Yao et al., 2019). Due to the fact of DocRED having limitations, we evaluated the relation extraction on an improved version of the dataset, namely Revisiting DocRED - Addressing the Overlooked False Negative Problem in Relation Extraction (ReDocRED) (Tan et al., 2022).

4.2 Experimental Setup

Long-LUKE is based on BERT (Devlin et al., 2019), containing $D = 1024$ hidden dimensions, 24 hidden layers, 16 attention heads each with 64 head dimension, and 256 entity token embeddings. The parameters on our word and entity embeddings are $483M$ and $128M$, respectively. Our model was implemented based on Huggingface and Pytorch. We used the finetuned models of LUKE-Large as the base for our work. We adapt it into a Longformer without needing additional training. We extend the limit sequence limit to 1024 tokens (i.e., initializing the additional position embeddings by copying the embeddings of the first positions) and alter the implementation of the self-attention operations within each layer while keeping the model parameters. In theory, by doing this, we should be able to model entities in extended sequence examples, which is especially important to perform relation extraction between entity pairs far apart from each other. Our model is then trained to predict relations between pairs of entities in the same or different sentence or to predict the type of entity according to the context surrounding the target entity. Specifically, we started from the finetuned *Huggingface* Transformer models of *studio-ousia/luke-large*. We adapted it to a Long-Luke through the procedure described in Section 3. From the Longformer implementation, we chose to use the regular sliding window attention pattern instead of the dilated sliding window pattern as it was only used in a PyTorch memory-efficient slow implementation and another implementation using an optimized custom CUDA kernel implemented using TVM (Chen et al., 2018), targeted for the language modelling experiments. Due to time and memory restraints, we could not explore the TVM implementation. The Pytorch-friendly implementation, used for the pre-training/finetuning setting, only supports the non-dilated case. We set the value of max_sequence as 1024 for all experiments. The global attention strategy is only applied to entity-related tokens. We tuned the hyperparameters on the development dataset. We trained in one Quadro RTX 8000 and evaluated our model following the metrics of each dataset. Regarding the training of our model for each dataset and experiment, we tried to use the hyperparameters cited in Vaswani et al. (2017). We chose sliding window size $= 256$ for all layers, maximum word length $= 1024$, batch size $= 64$, learning rate $= 1e-5$, and the *Adam* optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e-6$. For the ReDocRED relation classification task, we finetuned Long-LUKE with *Adafactor* using learning rates $5e-5$ over 7 epochs and evaluated our model with standard F_1, following Yao et al., 2019. On the remaining datasets, we finetuned our model over 3 epochs. For every experiment, we finetuned the model with a multi-class Linear classification head on top, having the input features be the output of Long-LUKE, and the output features the number of classes of each dataset.

4.3 Experimental Results

We performed our first experiments on entity typing (see Figure 5), which is the task of identifying the type of an entity given a sentence. We use the Open Entity (Choi et al., 2018), the FIGER (Ling and Weld, 2012), and the DocRED (Yao et al., 2019) datasets. We report loose micro-precision, recall, and F1 for both datasets and employ the micro-F1 as the primary metric.

We performed the following experiments on relation extraction, which is the task that assigns the correct relation between *head* and *tail* entities in a document. We used TACRED dataset (Zhang et al., 2017), a large-scale intra-sentence relation classification dataset, as well as a dataset focused on inter-sentence relations, named ReDocRED (Tan et al., 2022).

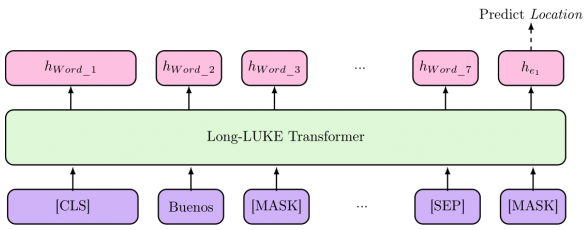


Figure 5: Entity typing task from Long-LUKE.

4.3.1 Results on the FIGER Dataset

We compare Long-Luke with transformer-based models, including LUKE (Yamada et al., 2020), K-Adapter (Wang et al.,

2021), and RoBERTa (Liu et al., 2019). The performance reported in the original paper is successfully reproduced. From Table 1, we observed that our approach obtains the same performance reported in the original LUKE implementation (Yamada et al., 2020), which is expected since, in this dataset, there is not any example that surpasses the 512 tokens limit. This validates that Long-LUKE behaves the same as the original LUKE for the defined window size whenever the maximum sequence of examples does not surpass the original maximum length.

4.3.2 Results on the OpenEntity Dataset

We evaluate a multi-label entity classification task for this dataset based on the corresponding entity representation. From Table 1, both Long-LUKE and LUKE outperform the transformer-based baseline models, including K-Adapter (Wang et al., 2021) and RoBERTa (Liu et al., 2019). The improvement of the Precision and Recall values displays that modelling entities contribute to the model returning more relevant results than irrelevant ones. Also, the increased recall value symbolizes that the model returns more relevant results. Both LUKE and Long-LUKE achieved the same performance values for this dataset, and this is because OpenEntity does not have examples whose sequences are longer than 512 tokens. In this context, Long-LUKE replicates LUKE.

4.3.3 Results on the TA-CRED Dataset

The experimental results for this dataset are in Table 2. Both Long-LUKE and LUKE outperform our established baselines. This dataset only contains concise sequenced examples of a maximum of 50 tokens, which does not enable us to evaluate the performance of our model over long sequences. Nevertheless, we verified that Long-LUKE returns the same performance as the original LUKE and that introducing entities was essential in improving both Precision, Recall, and F_1 scores.

4.3.4 Results on the DocRED Dataset

Thus far, most existing relation extraction methods only focus on extracting intra-sentence relations from single entity pairs. Therefore, to work with more complex inter-sentence relations between pairs of entities, we decided to use DocRED (Yao et al., 2019), which is constructed from Wikipedia and Wikidata. This dataset annotates both named entities and relations. It requires reading multiple sentences in a document to extract entities and predict their relations by synthesizing all information from the document. We find that DocRED is more prominent than existing datasets in many aspects, including the number of documents, words, sentences, and entities, especially in aspects of relation types, relation instances and relational facts. DocRED includes 96 common relation types from Wikidata, covering a wide range of categories: science, art, time, personal life, and more. Concerning the data split, DocRED contains 3,053 training examples, 998 validation examples and 1,000 examples.

In DocRED, an entity usually contains more than one mention occupying different positions in the document. For example, if in a document we have the mentions "Portugal" and "República Portuguesa", they represent the same entity "Portugal", even though they are two different words in the document. In contrast with other RE datasets such as TA-CRED, DocRED has entity linking, meaning all mentions are grouped. Therefore, a relation between 2 pairs of entities in DocRED can be seen by LUKE as a relation between several pairs of mentions (i.e., in DocRED, a relation between an entity A with m mentions and an entity B with n mentions result in $m \times n$ pairs of relations between mentions to be fed into LUKE). For that reason, to obtain the relation between entities, we choose the relation between the entity pair following the more representative relation between mentions. This is because we need to feed the mentions' entity spans into our model to create the inputs, and only mentions have entity spans assigned to themselves. Furthermore, the evaluation dataset is private and does not contain any labels. Therefore we had to predict every relation between every entity in each document. The evaluation was performed through the https://codalab.lisn.upsaclay.fr/competitions/365 platform. Consequently, we had to extend the training dataset using the finetune of our model for this task. The labels in the training dataset only contained examples where there is a relation, disregarding the cases where there is no relation. In order to include the missing negative examples, we considered all the relations between every mention of each example. We marked their relation as N/A whenever they were missing from the original dataset. As a consequence, the number of examples increases. We now have 1,198,650 training examples and 392,158 evaluation examples. Because we do not have entity relation labels in the test dataset, we need to do this to predict only the positive relations expected by the official CODALAB evaluation.

Several documents of DocRED have text sequences longer than the usual 512 tokens most models can deal with. Consequently, we can also use this dataset to evaluate the performance of our implemented self-attention suitable for longer sequences. However, in my implementation, I used an unbalanced training dataset for finetuning with too many N/A examples, which caused all relations to be predicted as N/A in the official evaluation. Due to being unable to filtrate the positive relations, the unavailability of the real relation test labels, the large number of training examples, and our GPU memory limitations, we found that it would be too costly to re-train both our model and our baselines on this dataset. As an alternative, we decided to evaluate DocRED on a different task other than relation classification. We evaluated Long-Document Entity Typing, taking advantage of the dataset's available labels for the types of entities. From Table 3, both LUKE and Long-LUKE obtain up to expected results. Comparing both implementations, Long-LUKE has access to information that LUKE does not. Long-LUKE can access a greater range of context, enabling the model to identify the target entity type successfully. Such differences are noticeable in the slight increase of Precision, Recall, and F_1. Despite Long-LUKE improving over LUKE, we do not have other baselines to compare.

4.3.5 Results on the ReDocRED Dataset

Analyzing the experiments that used DocRED, we can note that it is challenging for existing RE methods, which is a sign that document-level RE remains an open problem which requires further efforts. Tan et al. (2022) found that DocRED is incomplete for relation extraction tasks due to the several false negative samples, therefore developed ReDocRED (Tan et al., 2022), which was built by re-annotating over four thousand documents in DocRED dataset by adding the missed relation triples. For the relation extraction task, ReDocRED introduced an additional classification label (N/A), meaning the test set contains all positive and negative relation labels, which does not happen in the CODALAB official evaluation task for the DocRED dataset. We obtained the experimental results using a 70% partial negative sample due to the extensive data samples of the dataset. Also, we considered every relation between entities three sentences apart from each other as having the N/A relation. This technique helps us discard the relations between pairs of entities with a low probability of being related. Concerning the results in Table 2, it is noticeable that both LUKE and Long-LUKE benefit from having entity representations since they delivered higher F_1 scores compared to their baseline counterparts. Furthermore, Long-LUKE obtained a slightly better score than LUKE because of its ability to cover more expansive sequences of texts

and acquire a richer understanding of the context of the entity pair.

Method	OpenEntity (entity typing)			FIGER (entity typing)		
	P	R	micro-F_1	Acc	macro-F_1	micro-F_1
Long-LUKE	**79.69**	**76.5**	**78.1**	**64.73**	**88.28**	**88.26**
LUKE	**79.69**	**76.5**	**78.1**	**64.73**	**88.28**	**88.26**
K-Adapter	79.3	75.8	77.5	59.5	84.52	80.42
RoBERTa	77.6	75.0	76.2	56.31	82.43	77.83

Table 1: Evaluation metrics for a set of Entity Typing baseline methods.

Method	TA-CRED (relation extraction)			ReDocRED (relation extraction)
	P	R	micro-F_1	F_1
LUKE	70.4	75.1	**72.7**	82.64
Long-LUKE	70.4	75.1	**72.7**	**85.02**
RoBERTa	70.2	72.4	71.3	-
BERT-base	67.2	64.8	66.0	53.20
DocuNET-RoBERTa-large	-	-	-	79.09

Table 2: Evaluation metrics for a set of Relation Extraction baseline methods.

Method	DocRED (entity typing)		
	P	R	F_1
Long-LUKE	**73.09**	**73.09**	**73.09**
LUKE	71.28	71.28	71.28

Table 3: Evaluation metrics for another set of Entity Typing baseline methods.

5 Conclusions and Future Work

We proposed a new entity-aware method for entity typing and relation extraction, extending the previous LUKE (Yamada et al., 2020) and Longformer (Beltagy et al., 2020) approaches. We tested our model over multiple datasets, showing that some tasks benefit from having the representation of entities and the richer context surrounding such entities.

For future work, we can consider other methods for handling long inputs besides the Longformer (i.e., memory-efficient attention implementations (Rabe and Staats, 2021), or other sparse attention patterns such as those in the Hypercube Transformer (Wang et al., 2022)) or Big Bird (Zaheer et al., 2020). It would be interesting to apply the pre-training task of Improving Entity and Relation Understanding for Pre-trained Language Models via Contrastive Learning (ERICA) (Qin et al., 2021) to our model. This task would improve the model's knowledge of inter-sentence entity relations, and it should be able to increase the evaluation scores on the relation extraction task of DocRED. Also, note that our model was not pre-trained over more data. It would be beneficial to run a task similar to MLM with really long sequenced examples. By doing this the model would further its knowledge of entity saliency and entity relations, which we believe would increase the scores of all tasks overall. Also, in further experiments, we would evaluate our model on the OntoNotes 5.0 (Weischedel et al., 2022) and LongtoNotes (Shridhar et al., 2022) datasets to test the performance limit of our model, as the datasets contain several examples in which sequences surpass 1024 tokens, the number we defined as the largest sequence to be fed into our model. The idea would be to stress further the length of the textual inputs that need to be analyzed. Another idea for future work could be to evaluate our model on other relevant tasks, such as keyphrase extraction. Finally, one could also evaluate the performance of entity-related models on the ReDocRED dataset without the negative examples, meaning all the samples of targeted pairs of entities would have an existing relation. Despite being an interesting experiment, we could not execute it due to time constraints.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (ACL): Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. doi: 10.18653/v1/N19-1423.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *Proceedings of the International Conference on Neural Information Processing Systems*, page 6000–6010, 2017.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454, 2020.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 2895–2905, 2019.

Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. ERICA: Improving Entity and Relation Understanding for Pre-trained Language Models via Contrastive Learning. In *Proceedings of the Meeting of the Association for Computational Linguistics (ACL) and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3350–3363, 2021.

Hideaki Joko, Faegheh Hasibi, Krisztian Balog, and Arjen P. de Vries. *Conversational Entity Linking: Problem Definition and Datasets*, page 2390–2397. 2021.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null): 993–1022, mar 2003. ISSN 1532-4435.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The Long-Document Transformer. *Computing Research Repository (CoRR)*, abs/2004.05150, 2020.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big Bird: Transformers for Longer Sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297, 2020.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. DocRED: A large-scale Document-Level Relation Extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777. Association for Computational Linguistics, July 2019. doi: 10.18653/v1/P19-1074.

Qingyu Tan, Lu Xu, Lidong Bing, and Hwee Tou Ng. Revisiting docred - addressing the overlooked false negative prob-

lem in relation extraction. *Computing Research Repository (CoRR)*, 2022.

Y Liu, M Ott, N Goyal, J Du, M Joshi, D Chen, O Levy, M Lewis, L Zettlemoyer, and V Stoyanov. Roberta: A Robustly optimized BERT pretraining approach, 2019. *arXiv preprint arXiv:1907.11692*, 364, 2019.

Jun-Ting Hsieh. Effective word representation for named entity recognition. 2017.

Markus N. Rabe and Charles Staats. Self-attention Does Not Need O(n^2) Memory. *Computing Research Repository (CoRR)*, abs/2112.05682, 2021.

Zi-Hang Jiang, Weihao Yu, Daquan Zhou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Convbert: Improving bert with span-based dynamic convolution. *Advances in Neural Information Processing Systems*, 33:12837–12848, 2020.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *CoRR*, abs/2001.04451, 2020.

Xiao Ling and Daniel S. Weld. Fine-Grained Entity Recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, AAAI'12, page 94–100, 2012.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. Ultra-Fine Entity Typing. In *Proceedings of the Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers)*, pages 87–96, 2018.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, pages 35–45, 2017.

Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Meghan Cowan, Haichen Shen, Leyuan Wang, Yuwei Hu, Luis Ceze, Carlos Guestrin, and Arvind Krishnamurthy. Tvm: An automated end-to-end optimizing compiler for deep learning, 2018.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *In Proceedings of the Association for Computational Linguistics (ACL)*, pages 1405–1418, 2021.

Yuxin Wang, Chu-Tak Lee, Qipeng Guo, Zhangyue Yin, Yunhua Zhou, Xuanjing Huang, and Xipeng Qiu. What dense graph do you need for self-attention?, 2022.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Hovy Eduard, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. OntoNotes Release 5.0, 2022.

Kumar Shridhar, Nicholas Monath, Raghuveer Thirukovalluru, Alessandro Stolfo, Manzil Zaheer, Andrew McCallum, and Mrinmaya Sachan. Longtonotes: Ontonotes with longer coreference chains. *Computing Research Repository (CoRR)*, abs/2210.03650, 2022.