SPOT: Knowledge-Enhanced Language Representations for Information Extraction

Jiacheng Li
University of California, San Diego
j9li@eng.ucsd.edu

Yannis Katsis
IBM Research
yannis.katsis@ibm.com

Tyler Baldwin
IBM Research
tbaldwin@us.ibm.com

Ho-Cheol Kim
IBM Research
hckim@us.ibm.com

Andrew Bartko
University of California, San Diego
abartko@eng.ucsd.edu

Julian McAuley
University of California, San Diego
jmcauley@eng.ucsd.edu

Chun-Nan Hsu
University of California, San Diego
chunnan@ucsd.edu

ABSTRACT

Knowledge-enhanced pre-trained models for language representation have been shown to be more effective in knowledge base construction tasks (i.e., relation extraction) than language models such as BERT. These knowledge-enhanced language models incorporate knowledge into pre-training to generate representations of entities or relationships. However, existing methods typically represent each entity with a separate embedding. As a result, these methods struggle to represent out-of-vocabulary entities and a large amount of parameters, on top of their underlying token models (i.e., the transformer), must be used and the number of entities that can be handled is limited in practice due to memory constraints. Moreover, existing models still struggle to represent entities and relationships simultaneously. To address these problems, we propose a new pre-trained model that learns representations of both entities and relationships from token spans and span pairs in the text respectively. By encoding spans efficiently with span modules, our model can represent both entities and their relationships but requires fewer parameters than existing models. We pre-trained our model with the knowledge graph extracted from Wikipedia and test it on a broad range of supervised and unsupervised information extraction tasks. Results show that our model learns better representations for both entities and relationships than baselines, while in supervised settings, fine-tuning our model outperforms RoBERTa consistently and achieves competitive results on information extraction tasks.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**.

KEYWORDS

language model, knowledge representation, pre-trained model, information extraction, representation learning

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10.
https://doi.org/10.1145/3511808.3557459

ACM Reference Format:
Jiacheng Li, Yannis Katsis, Tyler Baldwin, Ho-Cheol Kim, Andrew Bartko, Julian McAuley, and Chun-Nan Hsu. 2022. SPOT: Knowledge-Enhanced Language Representations for Information Extraction. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557459

1 INTRODUCTION

Language models pre-trained on a large amount of text such as BERT [3], GPT [42], and BART [21] have achieved the state-of-the-art on a wide variety of natural language processing (NLP) tasks. The various self-supervised learning objectives, such as masked language modeling [3], enable language models to effectively learn syntactic and semantic information from large text corpora without annotations. The learned information can then be transferred to downstream tasks, such as text classification [57], named entity recognition [45], and question answering [44].

Knowledge graphs (KG) provide a rich source of knowledge that can benefit information extraction tasks such as named entity recognition [11], relation extraction [67] and event extraction [54]. Despite pre-trained models achieving success on a broad range of tasks, recent studies [40] suggested that language models pre-trained with unstructured text struggled to generate vectorized representations (i.e., embeddings) of entities and relationships and injecting prior knowledge from KG to language models were attempted.

Many attempts have been made to inject knowledge into pre-trained language models [17, 20, 39, 58, 61, 63, 68]. These previous works typically used separate embeddings for knowledge (i.e., entities and relationships in a KG) during pre-training or fine-tuning on downstream tasks. For example, ERNIE [68] first applied TransE [1] on KG to obtain entity embeddings and then infused entities' embeddings into the language model. LUKE [63] trained entity embeddings with token embeddings together in a transformer [55] by predicting masked tokens or entities. However, separate embeddings occupy extra memory and limit the number of knowledge entries that the model can handle. As is shown in Table 1, ERNIE and KnowBERT [39] need more than 400 million parameters for entity embeddings. After restricting the number of entities, LUKE needs 128 million parameters for 500 thousand entities. K-Adapter [58]

Methods	# params. (Millions)	Entity representation	Relation representation
ERNIE [68]	483.9	Yes	No
KnowBERT [39]	413.3	Yes	No
LUKE [63]	128	Yes	No
K-Adapter [58]	42	No	No
SPOT (Ours)	21	Yes	Yes

Table 1: Comparison of current knowledge-enhanced language models based on the number of additional parameters (in millions) and whether they learn entity or relation representations.

uses only 42 million additional parameters [1], but cannot generate entity representations. Hence, to represent entities, previous works required a large number of additional parameters. Additionally, the separate embedding tables limit the ability to representing out-of-vocabulary entities. For example, LUKE was pre-trained on the wikipedia article for entity representation but it struggles to represent biomedical entities in our entity clustering experiments (Section 4.3.1). Furthermore, existing methods focused only on entity representations and lack the capacity to represent relationships to support downstream tasks for knowledge base construction such as relation extraction. As shown in our results (Figures 3), simple methods based on entity represenations (e.g. by concatenation) struggled to produce informative relation representations. ERICA [41] proposed to apply contrastive learning on entity and relation representations, but its use of simple average pooling on token representations cannot represent entities effectively especially for tasks that need entity boundaries such as joint entity and relation extraction. As a result, we still need an effective pre-trained model that can incorporate external knowledge graphs into language modeling, and simultaneously learn representations of both entities and relationships without adding hundreds of millions of parameters to the model to support knowledge base construction tasks.

In this paper, we presented SPOT (SPan-based knOwledge Transformer) [2], a span-based language model that can be pre-trained to represent knowledge (i.e., entities and relationships) by encoding spans from input text. Intuitively, words, entities and relationships are hierarchical (i.e., entities contains words, relationships contains entities), it is possible to learn representations of entities from words and representations of relationships from entities. Hence, to incorporate knowledge from a KG, we can represent entities and relations by token spans and span pairs, respectively. There are two advantages of encoding spans: (1) The pre-trained span encoder and pair encoder can effectively represent knowledge with much less parameters than works with separate embeddings. Different from previous works that learn an embedding for each entity, SPOT learns transformation from the language model's token representations to entity representations. Therefore, as shown in Table 1, SPOT needs only 21 million parameters (16 million for entities and 5 million for relationships) to incorporate knowledge into language model. (2) Based on a span encoder and span pair encoder, SPOT learns both entity representations and relation representations in a unified way. Specifically, the span encoder learns entity representations from tokens and the span pair encoder learns relation representations from entities. In this hierarchical way, SPOT can effectively represent entities which do not appear in pre-training because our entity representations consist of token representations. With spans, SPOT incorporates knowledge into language modeling and also generates representations of both entities and relations.

Specifically, to inject knowledge into a span-based model, we first designed an effective model architecture to encode spans and span pairs. Then, we adopted three pre-training tasks that jointly learn knowledge at three different levels, i.e., token, entity and relation. Specifically, at the token level, we masked random tokens as in the masked language model (MLM) used in the pre-training of BERT [3]. At the entity and relation levels, the pre-training tasks are to predict entities and relations based on representations generated by the span encoder and span pair encoder, respectively. In this way, SPOT learns to infer entities and relations from both entity mentions and their contexts in the training text during the pre-training.

We pre-trained SPOT on Wikipedia text and used Wikidata as the corresponding knowledge graph. After pre-training, we conducted extensive experiments on five benchmark datasets across entity and relationship clustering tasks and three supervised information extraction tasks, namely joint entity and relation extraction, entity typing and relation extraction. Experiments showed that fine-tuning our pre-trained model consistently outperformed RoBERTa [26], and achieved new state-of-the-art performance on four datasets. In addition, we compared our model with RoBERTa and LUKE by visualization to assess how well our pre-trained model learns representations of entities and relationships from pre-training. Results indicated that our pre-trained model learned meaningful representations for both entities and relationships to support various knowledge base construction tasks.

Our contributions are in three folds:

- We proposed to apply spans to effectively inject knowledge into a language model and generate highly informative entity and relationship representations.
- We designed a novel pre-training objective and trained a span-based framework with the Wikipedia dataset.
- Extensive experiments were conducted and showed that SPOT outperformed other knowledge-enhanced language model on information extraction tasks and generated superior knowledge representations.

2 RELATED WORKS

2.1 Joint Entity and Relation Extraction

Since entity detection and relation classification are two essential tasks for knowledge base construction, numerous works [22, 46, 47] were proposed on the two tasks. Because entity and their relationship recognition can benefit from exploiting interrelated signals, many models for joint detection of entities and relations were proposed recently. Most approaches used special tagging scheme for this tasks. Miwa and Sasaki [34] modeled joint entity and relation extraction as a table-filling problem, where each cell of the table corresponded to a word pair of the sentence. The BILOU tags were filled into the diagonal of the table and relation types were predicted in the off-diagonal cells. Similar to Miwa and Sasaki [34], Gupta

[1] Additional parameters do not include parameters used in the backbone model (e.g., BERT) for fair comparison.
[2] The codebase will be released upon acceptance.

et al. [8] also formulated joint learning as table filling problem but used a bidirectional recurrent neural network to label each word pair. Different from the previous works with special tagging scheme, Miwa and Bansal [33] first applied a bidirectional sequential LSTM to tag the entities with BILOU scheme, and then a tree-structured RNN encoded the dependency tree between each entity pair to predict the relation type. IO-based tagging models cannot assign multiple tags on one token which limited the ability to recognizing multiple entities containing the common tokens. Hence, span-based approaches which performed an exhaustive search over all spans in a sentence were investigated and these approaches can cover overlapping entities. Dixit and Al-Onaizan [4] and Luan et al. [28] used span representations derived from a BiLSTM over concatenated ELMo [38] word and character embeddings. These representations were then used across the downstream tasks including entity recognition and relation classification. DyGIE [29] followed Luan et al. [28] and added a graph propagation step to capture the interactions among spans. With emerging of contextualized span representations, further improvements were observed. DyGIE++ [56] replaced the BiLSTM encoder with BERT. Other span applications included semantic role labeling [35] and co-reference resolution [18]. In this work, we used spans to incorporate entities and relationships from a knowledge graph into a language model. Due to the flexibility of spans, our model can output knowledge-aware token representations and knowledge representations simultaneously to support joint extraction of entities and relationships.

2.2 Pre-trained Language Representations

Early research on language representations focused on static unsupervised word representations such as Word2Vec [32] and GloVe [37]. The basic idea was to leverage co-occurrences to learn latent word vectors that approximately reflected word semantics. Dai and Le [2] and Howard and Ruder [10] first pre-trained universal language representations on unlabeled text, and applied task-specific fine-tuning on downstream tasks. Recent studies [30, 38] showed that contextual language representations were more powerful than static word embeddings because words could have different meanings in different contexts. Advances of transformer-based language models [3, 26, 42] continued to improve contextual word representations with more efficient large-scale models and novel pre-training objectives. These approaches demonstrated their superiority in various downstream NLP tasks. Hence, many language model extensions had been proposed to further improve the performance. SpanBERT [12] extended BERT by masking contiguous random spans rather than random tokens. Different from SpanBERT which predicted tokens by spans to obtain token representations, in our model, we apply a hierarchical structure (i.e., tokens, spans, span pairs) to represent tokens, entities and relationships in sentences. Song et al. [49] and Raffel et al. [43] explored the impacts of various model architectures and others [5, 16, 43] explored enlarged model sizes to improve general language understanding ability. MASS [49] and BART [21] extended the transformer encoder to the sequence-to-sequence architecture for pre-training. Multilingual learning [13, 15, 53] and multi-modal learning [27, 51, 52] were introduced to the pre-training. Although these pre-trained language models achieved success in various NLP tasks, they still focused on token-level representations but ignored the entities and their relations existing in the sentences, which are crucial for downstream tasks related to knowledge extraction and management. Our model is also based on the transformer but we focused on injecting knowledge from knowledge graphs into pre-trained models built on spans. Compared to the aforementioned pre-trained language models, our model is able to incorporate knowledge into representation learning and generate highly informative entity and relation representations for downstream tasks.

2.3 Knowledge-Enhanced Language Representations

Contextual pre-trained language models provided word representations with rich semantic and syntactic information, but these models still struggled to represent knowledge (i.e., entities and relationships), Efforts were made to improve learning of the representations of entities and relationships by injecting knowledge graphs into language models. Early attempts enforced language models to memorize information about entities in a knowledge graph with novel pre-training objectives. For example, ERNIE [68] aligned entities from Wikipedia sentences with fact triples in WikiData via TransE [1]. Their pre-training objective was to predict correct token-entity alignment from token and entity embeddings. KnowBERT [39] incorporated knowledge bases into BERT using knowledge attention and re-contextualization. Both ERNIE and KnowBERT enhanced language modeling by static entity embeddings separately learned from KGs. WKLM [61] replaced entity mentions in the original document and the model was trained to distinguish the correct entity mention from randomly chosen ones. KEPLER [59] encoded textual entity descriptions with pre-trained language models as entity embeddings, and then jointly optimized the knowledge embedding and language modeling objectives. GreaseLM [66] improved question answering by fusing encoded representations from pre-trained language models and graph neural networks over multiple layers of modality interaction (i.e., graphs and text) operations to obtain information from both modalities. Hence, this method allowed language context representations to be grounded by structured world knowledge. LUKE [63] applied trainable entity embeddings and an improved transformer architecture to learn word and entity representations together. Another line of works [14, 41, 64] modeled the intrinsic relational facts in text data, making it easy to represent out-of-KG knowledge in downstream tasks. Some works [36, 47] focused only on relationships and learned to extract relations from text by comparing the sentences that share the same entity pair or distantly supervised relation in KG.

Unlike the methods mentioned above, we used spans and span pairs to represent entities and relationships. After the pre-training, our knowledge enhanced language model can incorporate knowledge of KG into token representations and directly output meaningful representations of entities and relationships from given spans and span pairs without fine-tuning. Compared to previous works that used separate embedding tables for entities [39, 63, 68] or encoded entities using multiple language models [59], our model is novel in terms of leveraging span-based representations to achieve simplicity, efficiency and effectiveness.

3 METHOD

This section presents the overall framework of SPOT and its detailed implementation, including the model architecture (Section 3.2) and the novel pre-training task designed for incorporating the knowledge of entities and relationships from a knowledge graph (KG) (Section 3.5).

3.1 Notation

Given a text corpus and a corresponding KG, We denote a token sequence in the corpus as $\{w_1, \ldots, w_n\}$, where n is the length of the token sequence. Meanwhile, the entity span sequence aligning to the given tokens is $\{e_1, \ldots, e_m\}$, where $m \leq n$ is the number of entities contained in the token sequence. Each entity span is described by its start si and end ei) indices in the token sequence $e_i = (\text{si}, \text{ei})$. Finally, let $\{r_{ij}\}$ be the relation between entities e_i and e_j when the entities are related in the KG, where $1 \leq i < j \leq m$. SPOT will output the embeddings $\mathbf{w}$, $\mathbf{e}$, $\mathbf{r}$ to represent tokens, entities and relationships respectively.

3.2 Model Architecture

As shown in Figure 1, SPOT consists of three components:

(1) a textual encoder (TextEnd) responsible for capturing lexical and syntactic information from the input tokens;
(2) a span encoder (SpanEnd) that learns to generate the representation of contiguous tokens (spans) in the text as an entity; and
(3) a span pair encoder (PairEnd) that learns to generate the representation of span pairs capturing relation information between spans.

Each component (i.e., token, span, and span pair encoders) generate representations that accommodate a different information/knowledge type (i.e., word, entity, relation) in the text corpus and KG.

Textual encoder. Given a token sequence $\{w_1, \ldots, w_n\}$ and its corresponding entity spans $\{e_1, \ldots, e_m\}$, the textual encoder first sums the token embedding, segment embedding, and positional embedding for each token to compute its input embedding, and then computes lexical and syntactic features $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ by a multi-head self-attention mechanism by:

$$\{\mathbf{w}_1, \ldots, \mathbf{w}_n\} = \texttt{TextEnd}(\{w_1, \ldots, w_n\}) \tag{1}$$

where TextEnd is a multi-layer bidirectional Transformer, identical to its implementation in BERT [3].

Span encoder. Different from previous works that used separate embeddings for each entity in the KG, span representations are computed from token-level features $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ by SpanEnd for all entity spans $\{e_1, \ldots, e_m\}$ to obtain their entity representations $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$:

$$\begin{aligned}\{\mathbf{e}_1, \ldots, \mathbf{e}_m\} =& \texttt{SpanEnd}(\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}, \\ & \{e_1, \ldots, e_m\})\end{aligned} \tag{2}$$

Details of the span encoder SpanEnd will be described in Section 3.3.

Span pair encoder. To further capture the relations between entities in the KG, PairEnd computes relation representations between any two pairs of spans:

$$\mathbf{r}_{ij} = \texttt{PairEnd}(\{\mathbf{e}_i, \mathbf{e}_j\}). \tag{3}$$

$\mathbf{r}_{ij}$ is the relation representation of entities e_i and e_j.
Details of pair encoder PairEnd will be described in Section 3.4.

3.3 Span Encoder

As explained above, the goal of SpanEnd is to compute the span representations $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ from the token representations $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ and the entity spans $\{e_1, \ldots, e_m\}$. In this work, we explored three different methods to learn the representations of entity spans.

Boundary points. Span representations with boundary information were first applied to question answering [19]. We referred to this method as the EndPoint representation. In this method, the textual encoder outputs are concatenated at the endpoints of a span to jointly encode its inside and outside information. To distinguish different spans sharing the same endpoints (e.g., spans "`deep unsupervised learning`" and "`deep learning`"), the entity width embedding $\mathbf{E}_\mathbf{w} = \{e^1_w, \ldots, e^l_w\}$ is concatenated with the endpoint representation, where $\mathbf{E}_\mathbf{w} \in \mathbb{R}^{l \times d}$, l is the max length of the spans, d is the dimension of the embeddings, and where si and ei are the start and end positions of span e_i.

$$\mathbf{e}_i^{\text{endpoint}} = \left[\mathbf{w}_{si}, \mathbf{w}_{ei}, e_w^{(ei-si+1)}\right] \tag{4}$$

Self-attention. Lee et al. [18] described a method to learn a task-specific notion of a span head using an attention mechanism over words in each span. Given token representations $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ and a span $e_i = (\text{start}, \text{end})$, a self-attentive span representation is shown as follows:

$$\alpha_j = \text{FFNN}(\mathbf{w}_j) \tag{5}$$

$$a_{i,j} = \frac{\exp(\alpha_j)}{\sum_{k=\text{start}}^{\text{end}} \exp(\alpha_k)} \tag{6}$$

$$\mathbf{e}_i^{\text{selfattn}} = \sum_{j=\text{start}}^{\text{end}} a_{i,j} \cdot \mathbf{w}_j \tag{7}$$

where $\mathbf{e}_i^{\text{selfattn}}$ is a weighted sum of the token representations of tokens in an entity span e_j. The weights $a_{i,j}$ are automatically learned and $\text{FFNN}(\cdot)$ is a feed-forward neural network.

Max Pooling. The span representation $\mathbf{e}_i^{\text{pooling}}$ is obtained by maxpooling the token representations corresponding to each span. To be specific, $\mathbf{e}_i^{\text{pooling}} = \text{MaxPool}([\mathbf{w}_{\text{si}}, \ldots, \mathbf{w}_{\text{ei}}])$, where si and ei are the start and end positions of span e_i.

In our experiments the concatenation of $\mathbf{e}_i^{\text{endpoint}}$ and $\mathbf{e}_i^{\text{selfattn}}$ yielded the best results, which is why we adopt this as SPOT's final span representation: $\mathbf{e}_i = [\mathbf{e}_i^{\text{endpoint}}, \mathbf{e}_i^{\text{selfattn}}]$. Results of the comparison between the different span representation methods can be found in Section 4.8.

3.4 Pair Encoder

The pair encoder is responsible for generating a representation of the relation given two spans in a sentence. Relations between two

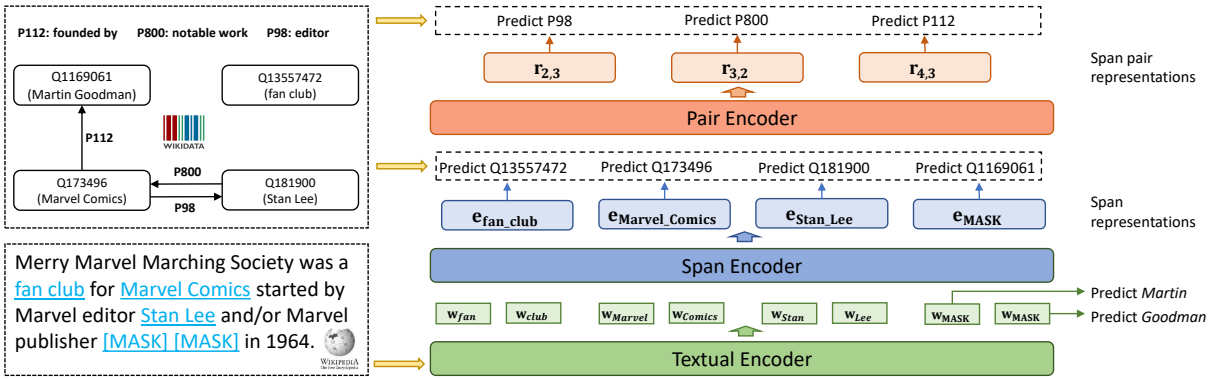

Figure 1: Overview of SPOT using an input sentence from pre-training dataset.

spans are usually determined by both the entity types of the spans and their context in a sentence.

In this paper, self-attention is used to encode span representations efficiently with the contextual information of the spans. Specifically, span representations are sent to self-attention layers and then use concatenation of two contextual span representations to construct the final span pair representation:

$$\{\tilde{\mathbf{e}}_1, \ldots, \tilde{\mathbf{e}}_m\} = \text{SelfAttn}(\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}) \tag{8}$$

$$\mathbf{r}_{ij} = \text{FFNN}([\tilde{\mathbf{e}}_i, \tilde{\mathbf{e}}_j]), \tag{9}$$

where SelfAttn$(\cdot)$ is a multi-head self-attention mechanism and FFNN$(\cdot)$ is a feed forward neural network.

3.5 Pre-training Task

To hierarchically learn representations from tokens, entities and relations at the same time. We propose a three-level pre-training task for our model: (1) token level, (2) entity level, and (3) relation level.

At the *token level*, similar to BERT, SPOT adopts the masked language model (MLM) but has different masking strategies on tokens. The intuition is that the model can learn to infer both masked tokens and masked entities in the input sentence based on their contexts and other entities. To this end, two masking strategies are performed:

(1) token masking: randomly mask 10% of tokens in sentences as other masked language models (e.g., BERT);
(2) entity masking: randomly mask 20% of entities [3] in sentences.

Following previous works, 10% of masked tokens are replaced with randomly selected tokens and keep 10% of tokens unchanged. The objective is the log-likelihood that maximizes the probability of masked tokens and is denoted by:

$$\mathcal{L}_{\text{MLM}} = -\sum_{w^* \in \mathcal{M}} \log P(w^* | \mathbf{w}_i) \tag{10}$$

where w^* is the masked token introduced in our two masking strategies and the masked token set is denoted by $\mathcal{M}$; $\mathbf{w}_i$ is the token representation from the token level of SPOT.

At the *entity level*, entity prediction is based on the span encoder's entity representation $\mathbf{e}_i$. Because of the large number and unbalanced distribution of entities, entity-level loss is computed by the adaptive softmax [7] with log-likelihood for pre-training,

$$\mathcal{L}_{\text{ENT}} = -\sum_{e^* \in \mathcal{D}} \log P(e^* | \mathbf{e}_i) \tag{11}$$

where e^* are gold entities in training documents $\mathcal{D}$. To endow the model with the ability to recognize entities from spans that do not express any entities. For each training instance, spans are randomly sampled as negatives which have the same number as entities in a sentence. Specifically, all spans up to the max length 8 are enumerated in our pre-training. Because most spans do not express any entities, we assumed that the random sampling will not sample any entities and used the sampled spans as negatives.

At the *relation level*, relationships are predicted between entity i and entity j from the entity pair representation $\mathbf{r}_{ij}$. Similar to tokens and entities, the log-likelihood is calculated to maximize the probability of the relation between two entities. The loss for relation $\mathcal{L}_{\text{REL}}$ is calculated by:

$$\mathcal{L}_{\text{REL}} = -\sum_{r^* \in \mathcal{D}} \log P(r^* | \mathbf{r}_i) \tag{12}$$

where r^* are ground-truth relations in the training documents $\mathcal{D}$.

The entire network is trained to convergence by minimizing the summation of the three losses.

$$\mathcal{L} = \mathcal{L}_{\text{MLM}} + \mathcal{L}_{\text{ENT}} + \mathcal{L}_{\text{REL}} \tag{13}$$

4 EXPERIMENTS

4.1 Pre-training Dataset

We used English Wikipedia as our pre-training corpus and aligned the text to Wikidata. Text with hyperlinks will be linked to an entity in Wikidata and two entities will be assigned with a relation in one sentence if they have a property in Wikidata. After discarding sentences without any relations, we used a subset of the whole dataset for pre-training. The subset corpus includes 31,263,279 sentences, 10,598,458 entities and 1,002 relations. For efficient pre-training, we only included the most frequent 1M entities in the corpus. Due to the unbalanced distribution of entities, the 1M entities cover 80% of phrases with hyperlinks in Wikipedia.

[3] All tokens in the entity are masked if an entity is selected to be masked.

4.2 Parameter Settings and Training Details

The backbone model of our textual encoder is $\text{RoBERTa}_{\text{large}}$. We implemented our method using Huggingface's Pytorch transformers [4]. The total amount of parameters of RoBERTa is about 355M. The span module and span pair module have 16M and 5M parameters respectively. We can see that our knowledge modules are much smaller than the language module. In the span encoder, the max length of spans is set to 8 and spans longer than this value are truncated. Two-layer self-attention with 8 heads is applied to obtain contextual span representations in the pair encoder. The hidden size of the span and pair encoder is 1,024 (same for the textual encoder). We pre-trained our model on the Wiki corpus for three epochs. To accelerate the training process, the max sequence length is reduced from 512 to 256 as the computation of self-attention is quadratic in the length. The batch size for pre-training is 96. We set the learning rate as 5e-5 and optimize our model with Adam. We pre-trained the model with three 32GB NVIDIA Tesla V100 GPUs for 23 days.

For all information extraction tasks, we used the Adam optimizer and select the best learning rate from {1e-5, 2e-5, 5e-5}, the best warm-up steps from {0, 300, 1000} and the best weight decay from {0.01, 1e-5}. We used a single GPU for all fine-tuning on downstream information extraction tasks. Linear learning rate decay strategy was adopted and gradient was clipped to 5 for all experiments. The dropout ratio was 0.1 for language model and 0.3 for task-specific layers. The batch size was 16 for joint learning entities and relationships, entity typing, and 32 for relation extraction task. We adopted early stop by using the best model on a development set before tested the model on a test set.

4.3 Clustering with Pre-trained Embeddings

To assess the quality of the representations of entities and relationships learned by SPOT, we conducted entity embedding clustering and relation embedding clustering and compared to a set of competitive language models as our baselines.

4.3.1 Entity Clustering. Entity clustering is conducted on BC5CDR [23], which is the BioCreative V Chemical and Disease Recognition task corpus. It contains 18,307 sentences from PubMed articles, with 15,953 chemical and 13,318 disease entities.

We compared SPOT to the following baselines:

- **GloVe** [37]. Pre-trained word embeddings on 6B tokens and the dimension is 300. We used averaged word embeddings as its entity representations.
- **BERT** [3]. Option 1: averaging token representations in an entity span (BERT-Ave.); 2: substituting entities with `[MASK]` tokens, and use `[MASK]` representations generated as the entity embeddings (BERT-MASK); or 3: concatenating representations of the first and the last tokens as the entity embeddings (BERT-End.).
- **LUKE** [63]. The contextual entity representations from LUKE are used.
- **ERICA** [41]. Applying the mean-pooling operation over their representations generated for consecutive tokens to obtain local entity representations.

[4] https://huggingface.co/transformers/

Table 2: Entity clustering results on BC5CDR.

Metrics	ACC	NMI	ARI
GloVe	0.587	0.026	0.030
BERT-Ave.	0.857	0.489	0.510
BERT-MASK	0.551	0.000	0.002
BERT-End.	0.552	0.000	0.003
LUKE	0.794	0.411	0.346
ERICA	0.923	0.628	0.715
SPOT	**0.928**	**0.645**	**0.731**

For SPOT, the outputs from span module represented entities. K-Means was applied to create the clusters for each entity. We followed previous clustering work [62] and adopted Accuracy (ACC), Normalized Mutual Information (NMI), and Adjusted Rand Index (ARI) [50] to evaluate the quality of the clusters from different models.

Table 2 shows the evaluation results of the entity clustering. In all metrics, SPOT achieved the best results compared to all baselines. Without any label, SPOT was able to distinguish disease entities and chemical entities with an accuracy of 0.928. Although ERICA achieved a close accuracy to SPOT, clear margins on NMI and ARI indicate that entities of the same class might were more concentrated with SPOT than ERICA. For BERT, average pooling was the best method to represent entities from token representations. SPOT outperformed LUKE though LUKE was designed specifically for entity representations with a large number of parameters to encode every entity. Averaging GloVe, BERT-Ave. and BERT-End. cannot provide effective entity representations. From the results, we can see that our pre-trained span encoder can output high-quality entity representations from token representations.

4.3.2 Relation Clustering. Relation clustering was performed on NYT24 [65], which contains 66,196 sentences and 24 relationships between entities distantly labeled by a knowledge base. In this experiment, we randomly sampled 4 relationships from the whole dataset and then clustered relationships into 4 groups. The sampling and clustering was repeated 5 times to reduce the bias of relationship selection.

We compared SPOT to the following baselines:

- **GloVe** [37]. The entity representations introduced in Section 4.3.1 were used as relation representations.
- **BERT** [3]. Two options: 1. Obtaining relation representations by concatenating entity representations in BERT-Ave.; 2. concatenating entity representations from BERT-End.
- **RoBERTa** [26]. Using the same method as BERT to obtain relation representations but loading the pre-trained parameters from RoBERTa.
- **LUKE** [63]. Concatenating entity representations as relation representations.
- **ERICA** [41]. The same relation representations as introduced by the authors were adopted.

For SPOT, the outputs from the span pair module represent relationships. The clustering method and evaluation metrics were the same as Section 4.3.1.

The results of relation clustering are shown in Table 3. Overall, SPOT achieved the best results compared to all baselines. We can

Table 3: Relation clustering results on NYT24. The averaged results are from 5 independently repeated experiments.

Metrics	ACC	NMI	ARI
GloVe	0.493	0.282	0.190
BERT-Ave.	0.501	0.230	0.134
BERT-End.	0.504	0.228	0.113
RoBERTa-Ave.	0.488	0.240	0.185
RoBERTa-End.	0.493	0.122	0.089
LUKE	0.557	0.388	0.274
ERICA	0.711	0.525	0.449
SPOT	**0.756**	**0.533**	**0.453**

see that knowledge-enhanced models (e.g., LUKE, ERICA and SPOT) largely outperform general language models (e.g., BERT, RoBERTa and GloVe). Because ERICA and SPOT incorporated both entity and relation knowledge during pre-training, these two models performed significantly better than LUKE on relation clustering.

From the entity and relation clustering, we can see that SPOT can effectively represent entities and relationships by the span modules.

4.4 Joint Entity and Relation Extraction

Given a sentence, joint entity and relation extraction methods both extract entities in the sentence and predict relations between these entities. A span and span pair framework which is consistent with our pre-training framework was applied for this task. All models were fine-tuned and evaluated on WebNLG [6] which contains 216 relation types. The maximum length of spans we considered was 5. We compared our model with the following models:

- **TPLinker** [60], which is a benchmark method used in joint entity and relation extraction;
- **TDEER** [24], which is considered as the state-of-the-art method in this task and the backbone is BERT;
- **BERT** [3] and **RoBERTa** [26], which are widely used as text encoders for various tasks;
- **SpanBERT** [12] which masks and predicts contiguous random spans instead of random tokens;
- **CorefBERT** [64] is a pre-training method that incorporates the coreferential relations in context;
- **ERICA** [41] improves entity and relation understanding by contrastive learning.

Among these baselines, **TPLinker** and **TDEER** are task-specific methods designed for joint entity and relation extraction; **BERT** and **RoBERTa** are language models for general purposes; **SpanBERT**, **CorefBERT** and **ERICA** are knowledge-enhanced language models. For a fair comparison with other language models, we did not load pre-trained parameters of span and span pair encoder in SPOT but trained them as a part of the fine-tuning.

Results in Table 4 show that SPOT achieved the highest Recall and F1 compared to task-specific models and other language models. Specifically, task-specific models performed better than other language models because their designed modules for this task. Knowledge-enhanced language models outperformed general language models, which indicates the effectiveness of incorporating knowledge into language models for joint entity and relation extraction. SPOT significantly outperformed all language models,

Table 4: Results on joint entity and relation extraction (WebNLG). We reported precision, recall and micro F1.

Metrics	Precision	Recall	F1
TPLinker	91.8	92.0	91.9
TDEER	**93.8**	92.4	93.1
BERT	91.3	92.5	91.8
RoBERTa	91.3	92.5	91.1
SpanBERT	92.1	91.9	92.0
CorefBERT	91.8	92.6	92.2
ERICA	91.6	92.6	92.1
SPOT	93.4	**93.1**	**93.3**

Table 5: Results on entity typing (FIGER). We reported precision, recall and micro F1 on the test set.

Metrics	Precision	Recall	F1
BERT	66.4	88.5	75.8
RoBERTa	65.1	88.1	74.9
SpanBERT	66.4	79.9	72.5
ERNIE	-	-	73.4
LUKE	**69.9**	89.0	**78.3**
WKLM	-	-	77
CorefBERT	62.4	82.2	72.2
ERICA	-	-	77.0
SPOT	68.5	**89.2**	77.5

which demonstrates that SPOT represented the knowledge better with our span and span pair modules.

4.5 Entity Typing

Entity typing aims at predicting the type of an entity given a sentence and its position. All models were trained and evaluated on the dataset FIGER [25] which contains 113 entity types, 2 million and 10, 000 distantly labeled sentences for training and validation; and 563 human labeled sentences as the test set. We compared our model with **BERT**, **RoBERTa**, **SpanBERT**, **CorefBERT**, **ERICA** and the following models:

- **ERNIE** incorporates knowledge graph information into BERT to enhance entity representations;
- **LUKE** treats words and entities in a given as independent tokens and outputs contextualized representations of them;
- **WKLM** employs a zero-shot fact completion task to improve pre-trained language models by involving knowledge.

Following Zhang et al. [68], two special tokens `[ENT]` are inserted into sentences to highlight entity mentions for language models such as **BERT** [5].

From the results listed in Table 5 we observe that, overall, knowledge-enhances models (e.g., LUKE, WKLM, ERICA) achieved significant improvements compared to general language models (e.g., BERT, RoBERTa, SpanBERT). LUKE achieved the best precision and F1 and SPOT achieved the best recall. Compared to our backbone

[5]For LUKE, we used the their contextual entity representations to predict the types; results of ERICA, ERNIE and WKLM were from their papers; other baselines used inserted special tokens to represent entities.

Table 6: Results on relation extraction (SemEval2010). We reported precision, recall and macro F1 on the test set.

Metrics	Precision	Recall	Macro F1
BERT	88.6	90.4	89.4
RoBERTa	88.4	89.0	88.7
SpanBERT	87.9	89.7	88.8
KnowBERT	89.1	89.1	89.1
MTB	88.1	90.1	89.2
CP	88.6	90.4	89.5
CorefBERT	89.2	89.2	89.2
LUKE	89.3	91.3	90.3
ERICA	89.6	89.0	89.2
SPOT	**89.9**	**91.4**	**90.6**

RoBERTa, SPOT improved RoBERTa by 2.6 (F1) which indicates the effectiveness of our span modules for knowledge incorporation.

4.6 Relation Extraction

Relation extraction aims at determining the correct relation between two entities in a given sentence. We evaluated all models on SemEval2010 [9] which contains 18 relation types, 8,000 sentences for training and 2,717 sentences for test. Macro F1 was used for SemEval2010 as the official evaluation. We compared SPOT with **BERT**, **RoBERTa**, **SpanBERT**, **CorefBERT**, **LUKE**, **ERICA** and three following models:

- **KnowBERT** [39] outputs contextual word embeddings enhanced by entity representations from knowledge graphs.
- **MTB** [48] is a pre-trained model designed for relation extraction by distinguishing if the two sentences express the same relationship.
- **CP** [36], a contrastive learning method that trains models on distantly labeled datasets.

To evaluate our model on this task, following Peters et al. [39], different tokens `[HD]` and `[TL]` were inserted for head entities and tail entities respectively and the contextual word representations for `[HD]` and `[TL]` were concatenated to predict the relationship.

Results in Table 6 show that SPOT outperformed all baselines and LUKE achieved similar results as SPOT. We can still see some improvements by incorporating external knowledge into language models because the F1 scores of all knowledge-enhanced models were larger than 89.

4.7 Representation Visualization

This section reports our study on whether our pre-trained span encoder and pair encoder can output meaningful entity and relationship representations without fine-tuning on task-specific datasets.

4.7.1 Entity Representations. To show SPOT can output meaningful entity representations without fine-tuning on downstream tasks, we applied pre-trained SPOT on BC5CDR and CoNLL2003 datasets to predict entity representations for annotated entities in the datasets. If an entity appears multiple times in the dataset, The mean of span representations was used as the entity representation. UMAP [31] was adopted to reduce the dimension of entity vectors. The cosine similarity was the metric of distance between vectors and the number of neighbors was set to 10 for all experiments. The results were shown in Figure 2a and 2e. We can see that SPOT distinguished chemical and disease entities well with an obvious gap between two clusters. Compared to BC5CDR, it is more difficult for pre-trained models to separate person names and organizations in CoNLL2003 because words in these two categories are low-frequency with many overlaps. However, SPOT can still group entities of names or organizations together.

To obtain entity representations without fine-tuning from RoBERTa, we adopted two methods:

(1) Concatenating the representations of the first and the last tokens in a span. This method is denoted as EndPoint;
(2) Maxpooling the representations of all tokens in a span.

The figures show that RoBERTa cannot separate different types of entities well especially for low-frequency entities such as person names and organizations in CoNLL2003.

The results (Figure 2b and 2f) show that LUKE achieved similar results to SPOT. However, note that LUKE requires 128M parameters for 500K entities compared to 16M for 1M entities with SPOT.

4.7.2 Relation Representations. Previous pre-trained models with knowledge focus on how to infuse knowledge into models [39, 61, 68] and LUKE [63] can output entity representations, but few can represent relationships without fine-tuning. In contrast, SPOT uses its pre-trained span pair module to output relation representations. Again, we used the same vector projection method (i.e., UMAP) with entity representations and considered four relations in NYT24:

(1) `place_lived` (purple) between person and place;
(2) `nationality` (blue) between person and country;
(3) `country` (green) between country and place;
(4) `capital` (yellow) between place and country.

As shown in Figure 3a, SPOT can separate the four different relations with four clearly distinguishable clusters.

Recall that EndPoint and Maxpooling were adopted to obtain entity representations from RoBERTa. To represent relations from RoBERTa, head and tail entities were concatenated to construct their relation representations. Figures 3c and 3d show that neither method can represent relations without fine-tuning on this task-specific dataset. Only `place_lived` (purple) groups together while the other three scatter.

We concatenated [6] contextual entity representations from LUKE to represent relations. Figure 3b shows that `country` green and `capital` yellow points scatter, suggesting that LUKE cannot represent these relations well without fine-tuning.

In summary, without fine-tuning, pre-trained language models (e.g., RoBERTa) cannot represent entities and relations by concatenation or maxpooling. SPOT outputs meaningful entity and relation representations without fine-tuning

4.8 Ablation Study

Recall that in Section 3.2, we introduced EndPoint, Self-attention, and Maxpooling as options for SPOT's span encoder and proposed to use span attention to obtain contextual span representations. We conducted experiments with BERT on the NYT24 dataset and compare micro F_1 scores of NER and relation extraction results

[6] Same as the relation representations in LUKE

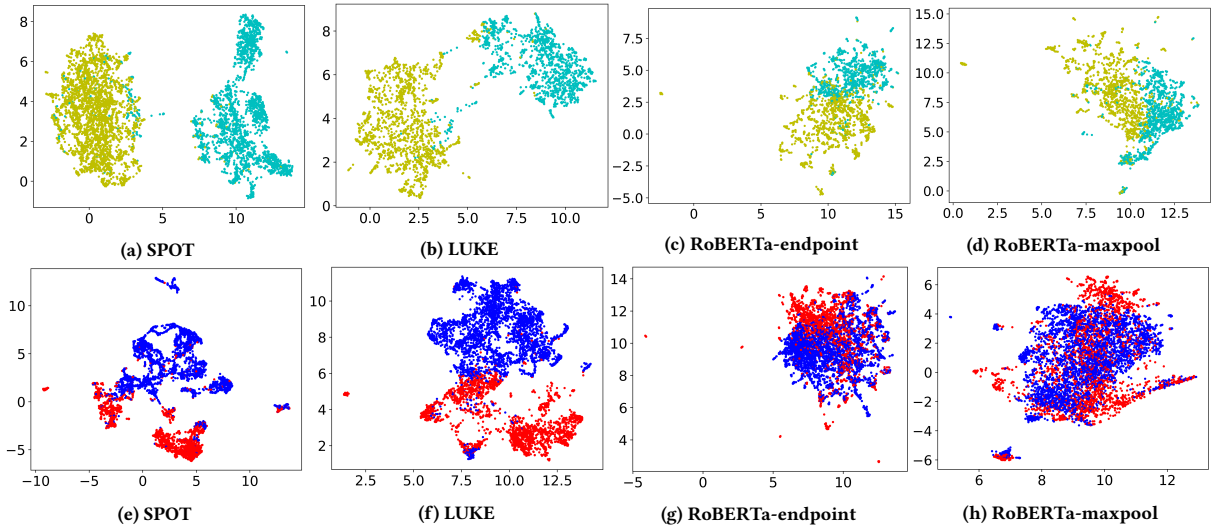

Figure 2: Embedding of Chemical (green) and Disease (yellow) entities in BC5CDR (upper line) and PER (blue) and ORG (red) entities in CoNLL2003 (lower line) from four models of entity representations.

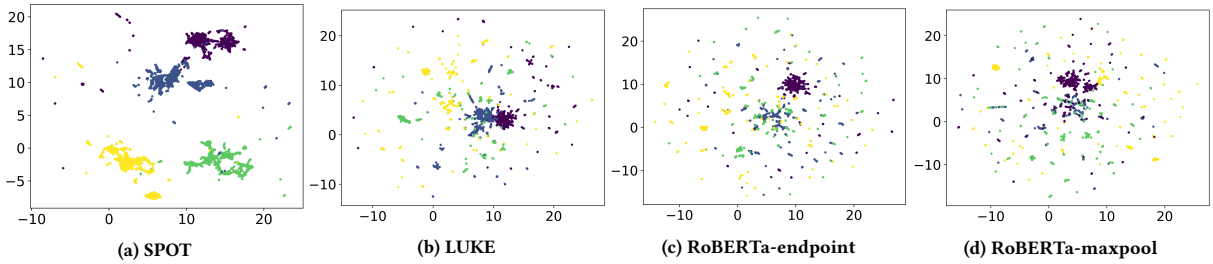

Figure 3: Embedding of four relations in NYT24 - place_lived (purple), nationality (blue), country (green), and capital (yellow) - from four models of relation representations.

Table 7: Results of ablation study. The last column shows relation results using self-attention for spans.

	Entity	Relation	+Span attn
EndPoint	94.92	76.99	77.48
Self-attention	73.92	55.80	-
MaxPooling	94.06	76.66	-
End+Att	**95.05**	78.80	**79.5**
End+Max	94.59	78.23	-
Att+Max	87.27	62.08	-

to validate our choice. We set the same hyper-parameters for all experiments. Specifically, the batch size was 16 and Adam optimizer was adopted with learning rate 2e-5 and linear learning rate decay. All gradients were clipped to 5 and warm up steps were 300. We adopted early stop by using the best model on development set and tested on a test set. When we used two kinds of encoders, we concatenated the two outputs of the encoders and fed the concatenation to a linear classifier.

Table 7 suggests that the most important method for span representation is EndPoint plus Self-attention (End+Att). Self-attention is known as ineffective when used solely as a span encoder but here we show that it can improve relation representations when working with EndPoint. Contextual span representations by span attention improve relation results because more contextual information can be learned which is important for relation extraction.

5 CONCLUSION

In this paper, we proposed SPOT to represent knowledge by spans and described a method to incorporate knowledge information into a language representation model. Accordingly, we proposed a span encoder and a span pair encoder to represent knowledge (i.e., entities and relationships in a knowledge graph) in text. SPOT can represent knowledge by spans without extra entity embeddings and entity lookup, requiring a much less number of parameters than existing knowledge-enhanced language models attempting to represent knowledge in language models. Our experimental results demonstrated that SPOT generated high-quality entity and relation representations based on our clustering experiments and achieved superior results on three supervised information extraction tasks. Overall, the results show that SPOT was more effective in representing entities and relationships without fine-tuning while requiring an order of magnitude less parameters than previous methods.

ACKNOWLEDGMENTS

This work is supported by IBM Research AI through the AI Horizons Network.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, J. Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*.

[2] Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. In *NIPS*.

[3] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.

[4] Kalpit Dixit and Yaser Al-Onaizan. 2019. Span-Level Model for Relation Extraction. In *ACL*.

[5] William Fedus, Barret Zoph, and Noam M. Shazeer. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *ArXiv* abs/2101.03961 (2021).

[6] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating Training Corpora for NLG Micro-Planners. In *ACL*.

[7] Edouard Grave, Armand Joulin, Moustapha Cissé, David Grangier, and H. Jégou. 2017. Efficient softmax approximation for GPUs. *ArXiv* abs/1609.04309 (2017).

[8] Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction. In *COLING*.

[9] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Uppsala, Sweden, 33–38. https://www.aclweb.org/anthology/S10-1006

[10] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *ACL*.

[11] Yang Jiao, Jiacheng Li, Jiaman Wu, Dezhi Hong, Rajesh K. Gupta, and Jingbo Shang. 2020. SeNsER: Learning Cross-Building Sensor Metadata Tagger. In *FINDINGS*.

[12] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics* 8 (2020), 64–77.

[13] D. Kondratyuk. 2019. 75 Languages, 1 Model: Parsing Universal Dependencies Universally. *ArXiv* abs/1904.02099 (2019).

[14] Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. 2020. A Mutual Information Maximization Perspective of Language Representation Learning. *ArXiv* abs/1910.08350 (2020).

[15] Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. In *NeurIPS*.

[16] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *ArXiv* abs/1909.11942 (2020).

[17] Anne Lauscher, Ivan Vulic, E. Ponti, A. Korhonen, and Goran Glavas. 2019. Informing Unsupervised Pretraining with External Linguistic Knowledge. *ArXiv* abs/1909.02339 (2019).

[18] Kenton Lee, Luheng He, M. Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. *ArXiv* abs/1707.07045 (2017).

[19] Kenton Lee, T. Kwiatkowski, Ankur P. Parikh, and Dipanjan Das. 2016. Learning Recurrent Span Representations for Extractive Question Answering. *ArXiv* abs/1611.01436 (2016).

[20] Yoav Levine, Barak Lenz, Or Dagan, Ori Ram, Dan Padnos, Or Sharir, S. Shalev-Shwartz, A. Shashua, and Y. Shoham. 2020. SenseBERT: Driving Some Sense into BERT. *ArXiv* abs/1908.05646 (2020).

[21] M. Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *ArXiv* abs/1910.13461 (2020).

[22] Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. 2021. Weakly Supervised Named Entity Tagging with Learnable Logical Rules. In *ACL*.

[23] J. Li, Yueping Sun, Robin J. Johnson, D. Sciaky, Chih-Hsuan Wei, Robert Leaman, A. P. Davis, C. Mattingly, Thomas C. Wiegers, and Z. Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database: The Journal of Biological Databases and Curation* 2016 (2016).

[24] Xianming Li, Xiaotian Luo, Cheng Jie Dong, Daichuan Yang, Beidi Luan, and Zhen He. 2021. TDEER: An Efficient Translating Decoding Schema for Joint Extraction of Entities and Relations. In *EMNLP*.

[25] Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design Challenges for Entity Linking. *Transactions of the Association for Computational Linguistics* 3 (2015), 315–328.

[26] Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv* abs/1907.11692 (2019).

[27] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *NeurIPS*.

[28] Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In *EMNLP*.

[29] Yi Luan, David Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A General Framework for Information Extraction using Dynamic Span Graphs. *ArXiv* abs/1904.03296 (2019).

[30] Bryan McCann, James Bradbury, Caiming Xiong, and R. Socher. 2017. Learned in Translation: Contextualized Word Vectors. In *NIPS*.

[31] L. McInnes and John Healy. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv* abs/1802.03426 (2018).

[32] Tomas Mikolov, Kai Chen, G. Corrado, and J. Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.

[33] Makoto Miwa and Mohit Bansal. 2016. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. *ArXiv* abs/1601.00770 (2016).

[34] Makoto Miwa and Yutaka Sasaki. 2014. Modeling Joint Entity and Relation Extraction with Table Representation. In *EMNLP*.

[35] Hiroki Ouchi, Hiroyuki Shindo, and Yuji Matsumoto. 2018. A Span Selection Model for Semantic Role Labeling. In *EMNLP*.

[36] Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2020. Learning from Context or Names? An Empirical Study on Neural Relation Extraction. In *EMNLP*.

[37] Jeffrey Pennington, R. Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*.

[38] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*.

[39] Matthew E. Peters, Mark Neumann, IV RobertLLogan, Roy Schwartz, V. Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *EMNLP/IJCNLP*.

[40] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. BERT is Not a Knowledge Base (Yet): Factual Knowledge vs. Name-Based Reasoning in Unsupervised QA. *ArXiv* abs/1911.03681 (2019).

[41] Yujia Qin, Yankai Lin, Ryuichi Takanobu, Zhiyuan Liu, Peng Li, Heng Ji, Minlie Huang, Maosong Sun, and Jie Zhou. 2021. ERICA: Improving Entity and Relation Understanding for Pre-trained Language Models via Contrastive Learning. In *ACL/IJCNLP*.

[42] Alec Radford and Karthik Narasimhan. 2018. Improving Language Understanding by Generative Pre-Training.

[43] Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *ArXiv* abs/1910.10683 (2020).

[44] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP*.

[45] E. T. K. Sang and F. D. Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. *ArXiv* cs.CL/0306050 (2003).

[46] Jingbo Shang, Liyuan Liu, Xiang Ren, Xiaotao Gu, Teng Ren, and Jiawei Han. 2018. Learning Named Entity Tagger using Domain-Specific Dictionary. In *EMNLP*.

[47] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. *ArXiv* abs/1906.03158 (2019).

[48] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and T. Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. *ArXiv* abs/1906.03158 (2019).

[49] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. MASS: Masked Sequence to Sequence Pre-training for Language Generation. In *ICML*.

[50] Douglas L. Steinley. 2004. Properties of the Hubert-Arabie adjusted Rand index. *Psychological methods* 9 3 (2004), 386–96.

[51] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. *ArXiv* abs/1908.08530 (2020).

[52] Chen Sun, Austin Myers, Carl Vondrick, Kevin P. Murphy, and Cordelia Schmid. 2019. VideoBERT: A Joint Model for Video and Language Representation Learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), 7463–7472.

[53] Hao Hao Tan and Mohit Bansal. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. *ArXiv* abs/1908.07490 (2019).

[54] H. Trieu, Thy Thy Tran, Khoa Duong, Anh Nguyen, Makoto Miwa, and S. Ananiadou. 2020. DeepEventMine: end-to-end neural nested event extraction from biomedical texts. *Bioinformatics* 36 (2020), 4910 – 4917.

[55] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *ArXiv* abs/1706.03762 (2017).

[56] David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. *ArXiv* abs/1909.03546 (2019).

[57] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *BlackboxNLP@EMNLP*.

[58] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, X. Huang, Jianshu Ji, Cuihong Cao, Daxin Jiang, and M. Zhou. 2020. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. *ArXiv* abs/2002.01808 (2020).

[59] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juan-Zi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics* 9 (2021), 176–194.

[60] Yucheng Wang, Bowen Yu, Y. Zhang, Tingwen Liu, Hongsong Zhu, and L. Sun. 2020. TPLinker: Single-stage Joint Extraction of Entities and Relations Through Token Pair Linking. In *COLING*.

[61] Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. *ArXiv* abs/1912.09637 (2020).

[62] Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao. 2017. Self-Taught Convolutional Neural Networks for Short Text Clustering. *Neural networks : the official journal of the International Neural Network Society* 88 (2017), 22–31.

[63] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. *ArXiv* abs/2010.01057 (2020).

[64] Deming Ye, Yankai Lin, Jiaju Du, Zhenghao Liu, Maosong Sun, and Zhiyuan Liu. 2020. Coreferential Reasoning Learning for Language Representation. In *EMNLP*.

[65] Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. In *ACL*.

[66] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning Enhanced Language Models for Question Answering. *ArXiv* abs/2201.08860 (2022).

[67] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware Attention and Supervised Data Improve Slot Filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*. 35–45. https://nlp.stanford.edu/pubs/zhang2017tacred.pdf

[68] Zhengyan Zhang, Xu Han, Z. Liu, Xin Jiang, M. Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *ACL*.