ContrastNER: Contrastive-based Prompt Tuning for Few-shot NER

Amirhossein Layegh[†], Amir H. Payberah[†], Ahmet Soylu[‡], Dumitru Roman[§], Mihhail Matskin [†]
[†]KTH Royal Institute of Technology, Sweden [‡]Oslo Metropolitan University, Norway [§]SINTEF AS, Norway
[†]{amlk, payberah, misha}@kth.se [‡]ahmet.soylu@oslomet.no [§]dumitru.roman@sintef.no

***Abstract*— Prompt-based language models have produced encouraging results in numerous applications, including Named Entity Recognition (NER) tasks. NER aims to identify entities in a sentence and provide their types. However, the strong performance of most available NER approaches is heavily dependent on the design of discrete prompts and a verbalizer to map the model-predicted outputs to entity categories, which are complicated undertakings. To address these challenges, we present ContrastNER, a prompt-based NER framework that employs both discrete and continuous tokens in prompts and uses a contrastive learning approach to learn the continuous prompts and forecast entity types. The experimental results demonstrate that ContrastNER obtains competitive performance to the state-of-the-art NER methods in high-resource settings and outperforms the state-of-the-art models in low-resource circumstances without requiring extensive manual prompt engineering and verbalizer design.**

***Index Terms*—Prompt-based learning, Contrastive learning, Language Models, Named Entity Recognition**

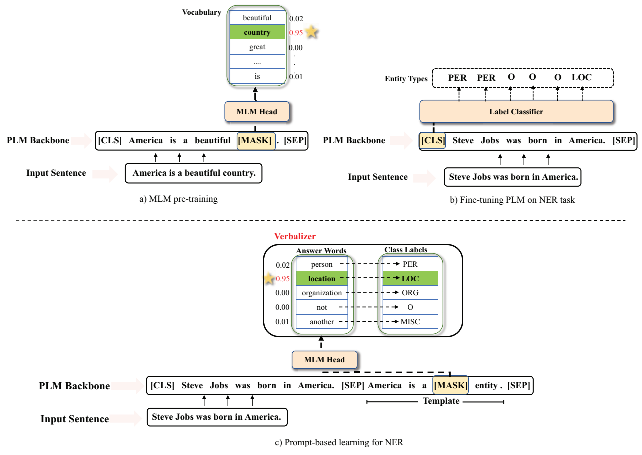


Fig. 1. Illustration of two different paradigms for solving NER task. The top image shows pre-training a language model using the MLM objective (1a) and fine-tuning it for a NER task to predict the entity type for each word in an input sentence (1b). The bottom image shows prompting the input sentence with a template to transform the NER task into an MLM problem to predict the type of a candidate entity in the input sentence (1c).

I. Introduction

Named Entity Recognition (NER) aims to recognize and classify named entities, such as person and location, into the appropriate concept classes. NER plays a crucial role in various applications, including information extraction, ontology population, question answering, machine translation, and semantic annotation, to name a few. Despite extensive research in this area, the state-of-the-art solutions still need more generalization and extensibility due to their reliance on domain-specific knowledge resources such as annotated training corpus. Considering that resources for data annotation are scarce in many domains and annotating a large corpus of text labeled in some domains requires the expertise of experts, low-resource NER tasks become a complex problem.

Pre-training a model on a rich-resource dataset and fine-tuning it on a low-resource downstream task is becoming more prevalent [1]. Typically, in the context of NER, the pre-trained step includes training a model using Masked Language Modeling (MLM) to predict the probability of the observed textual data. Then, the fine-tuning stage fine-tunes the Pre-trained Language Model (PLM) developed in the preceding step to predict the type of identified entities (Figure 1(a) and 1(b)). Recent NER models based on this methodology have demonstrated exemplary performance on NER tasks [2], [3]. However, to ensure high-quality learning, the downstream task requires a substantial amount of labeled data for satisfactory performance. Thus, if no annotated resources exist in the target domain, the model cannot identify the corresponding entity types, indicating poor generalization in low-resource circumstances.

Existing pre-training and fine-tuning NER models have an additional obstacle in the presence of two distinct pre-training and fine-tuning objectives (i.e., MLM in the pre-training and predicting the entity types in the fine-tuning). Initiated by GPT-3 [4], a *prompt-based* technique is one solution to bridging the gap between two different objectives. In this approach, the objectives of both the pre-training and fine-tuning stages are formulated as an MLM problem where the model directly predicts a textual answer to a given prompt [5], [6]. The generated answer will then be mapped to a class label using a *verbalizer* [7] (Figure 1(c)). A recent body of work investigates the setting of low-resource NER by applying prompt-based approaches to integrate the objectives of pre-training and fine-tuning phases [8]–[10]. However, these solutions depend

The work in this paper was partially funded by the projects DataCloud (H2020 101016835), enRichMyData (HE 101070284), Graph-Massivizer (HE 101093202), UPCAST (HE 101093216), and BigDataMine (NFR 309691). The experiments were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

significantly on expensive discrete prompt engineering.

To address these challenges, we present ContrastNER for the NER task that leverages the few-shot learning capabilities of PLMs without manual prompt engineering and verbalizer design. To this end, we introduce *soft-hard prompt tuning* for automatic prompt search in a mixed space of continuous and discrete prompts. Moreover, we employ contrastive learning to combine learning the soft-hard prompt and predicting the entity type without explicitly designing a verbalizer.

In summary, we present the following contributions:

- We propose ContrastNER, a prompt-based model for NER learning using a low-resource dataset and without manual search for appropriate prompts and creation verbalizers. To the best of our knowledge, our work is the first model that uses soft-hard prompt tuning and contrastive learning for prompt-based NER learning.
- We conduct experiments on four publicly available NER datasets demonstrating that our method outperforms the state-of-the-art low-resource prompt-based NER learning techniques.

II. Preliminaries and Problem Definition

In this section, first, we introduce the basic concepts of NER and then explain the fine-tuning of standard NER and prompt-based learning for few-shot NER.

A. Named Entity Recognition (NER)

NER aims to identify *entities* in a sentence and provide their *entity types*. Entities (e.g., noun/verb phrases) are the main parts of a sentence, and entity types are the labels for each entity, which are characterized based on predefined categories, such as place, person, organization, and suchlike.

In a NER dataset, an example typically is a pair of $(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X} = \{x_1, \cdots, x_m\}$, is a sentence containing m words, and $\mathbf{Y} = \{y_1, \cdots, y_m\}$ are the corresponding labels for each word in $\mathbf{X}$ that specify their entity types. For instance, the labels for $\mathbf{X}$ = {"Steve", "Jobs", "was", "born", "in", "America", "."}, are $\mathbf{Y}$ = {"PERSON", "PERSON", "O", "O", "O", "LOCATION", "O"}, where "PERSON", "O", and "LOCATION" indicate the organization entity type, not a named entity, and location entity type, respectively. The goal of a NER task is to predict the label (entity type) y_i for each word x_i. There are various methods for creating a NER task, which are discussed in the following sub-sections.

B. Pre-training and Fine-tuning Models for NER

One approach to training a model for a NER task involves fine-tuning PLM on a downstream NER task. This procedure usually entails two steps: (1) *pre-training* and (2) *fine-tuning*. In the first step, a model is trained using massive unlabeled text data. To this end (as illustrated in Figure 1(a)), a certain percentage of the input tokens (words) are randomly corrupted (replaced with [MASK] token), and the final corresponding vectors are given into a softmax over the vocabulary to predict these masked tokens. This process is known as *Masked Language Modeling* (MLM). Finally, the checkpoint of the trained model, $\mathcal{L}$, is saved as a PLM and will be fine-tuned on different downstream tasks.

In the second stage, using a NER dataset, the PLM $\mathcal{L}$ is fine-tuned on a downstream NER task. To do this, each input sentence $\mathbf{X} = \{x_1, \cdots, x_m\}$ in the dataset is first converted into the input sequence as {[CLS], $x_1, \cdots, x_m$, [SEP]}, where [CLS] is a special token and [SEP] is the end of the sequence. Then $\mathcal{L}$ encodes all tokens of $\mathbf{X}$ into an input embedding $\{h_{\texttt{[CLS]}}, h_{x_1}, h_{x_2}, \cdots, h_{x_m}, h_{\texttt{[SEP]}}\}$. Typically, a label-specific classifier is employed to compute the probability distribution of $h_{\texttt{[CLS]}}$ over the entity label space {"PER", "ORG", "MISC", $\cdots$} to assign correct entity types (Figure 1(b)). Finally, $\mathcal{L}$ is fine-tuned by minimizing the cross-entropy of the loss function.

C. Prompt-based Learning for Few-shot NER

A rich-resource dataset is usually required to train a NER model, which is not always available. Therefore, we should examine *few-shot NER*, where very few examples of each label (entity type) in the dataset are available. A common strategy for few-shot NER is to adapt a PLM $\mathcal{L}$ trained with MLM on a rich-resource dataset to a low-resource target NER dataset containing few examples per entity type [9], [11], [12].

One method to train a model for few-shot NER is to use prompt-based learning [13]. This approach reformulates the downstream NER task as an MLM problem using a textual prompt template. To this end, it is necessary to define two functions: (1) a prompt template $\mathcal{T}$ and (2) a verbalizer $\mathcal{M}$. The prompt template $\mathcal{T}$ organizes the input sentence $\mathbf{X}$, masked token, and prompt tokens as $\mathcal{T}(\mathbf{X})$ = {$\mathbf{X}$ $\langle$candidate_entity$\rangle$ is a [MASK] entity} (see Figure 1(c)). For instance, in a NER task with $\mathbf{X}$ = {Steve Jobs was born in America.}, the $\mathcal{T}(\mathbf{X})$ = {Steve Jobs was born in America. America is a [MASK] entity}, where America represents $\langle$candidate_entity$\rangle$.

After constructing $\mathcal{T}(\mathbf{X})$ for each entity (e.g., Steve and America), the PLM $\mathcal{L}$ will predict the mask token [MASK], which will be translated into an entity type by a verbalizer $\mathcal{M}$. For example, a verbalizer $\mathcal{M}$ can be defined as below:

$$\begin{gathered}\mathcal{M}(\texttt{"organization"}) \rightarrow \texttt{ORG}\\ \mathcal{M}(\texttt{"person"}) \rightarrow \texttt{PER}\\ \mathcal{M}(\texttt{"location"}) \rightarrow \texttt{LOC}\\ \cdots\end{gathered}$$

Handcrafting prompt templates using discrete tokens in natural language (e.g., "$\langle$candidate_entity$\rangle$ is") is challenging. An alternative approach is to use *soft prompts*, where continuous tokens are added to the input and are updated rather than discrete tokens [14]. P-tuning [6] is a model that uses prompt tuning to prevent prompt engineering with discrete tokens. P-tuning applies $\mathcal{T}$ on an input $\mathbf{X}$ and creates {$\mathbf{X}$ $h_0 \cdots h_n$ [MASK]}, where $\{h_0 \cdots h_n\}$ are continues prompts.

Consequently, P-tuning uses PLM $\mathcal{L}$ to create an embedding of `[MASK]` and applies a lightweight neural network, called *prompt encoder*, to learn the embedding of continuous prompts. During fine-tuning, $\mathcal{L}$'s parameters are frozen, and only the parameters of the prompt encoder and $\{h_0 \cdots h_n\}$ are updated. [6].

For example, for the input sentence $\mathbf{X}$ = `{Steve Jobs was born in America.}` and the trainable continuous prompt $\{h_0 \cdots h_n\}$, the prompt template is $\mathcal{T}(\mathbf{X})$ = `{Steve Jobs was born in America.` $h_0 \cdots h_i$ `[MASK]}`. During fine-tuning, $\mathcal{L}$ predicts the `[MASK]` embedding. They use sequence tagging to solve NER tasks by assigning labels marking at the beginning and the end of some entity classes, and the prompt encoder learns the embedding of $\{h_0 \cdots h_n\}$.

III. Related work

A. Named Entity Recognition

Using sequence labeling and Conditional Random Fields (CRF) [15] to associate each word in the input text with a label is a popular method to formulate NER tasks. Earlier works investigated utilizing several neural architectures such as BiLSTM [16] or CNN [17] and training the model in a supervised learning paradigm, which often involves a large amount of annotated data [18] [19] [20] [21]. Recently, PLMs have shown significant improvements in NER by using large-scale transformer-based architectures as the backbone for learning text representation [2]. The state-of-the-art results achieved by [3], [22] propose a new pre-trained contextualized representation of words with a transformer-based architecture pre-trained on a large set of the entity-annotated corpus. Despite the satisfactory performance achieved by PLMs in NER tasks, these approaches are primarily developed for supervised rich-resource NER datasets, which have limited generalization capability in low-resource datasets [23].

A more recent line of work has focused on enhancing model learning capabilities to maximize the use of existing sparse data and reduce reliance on data examples. One line of earlier work on low-resource NER has included prototype-based techniques that apply meta-learning [24] to few-shot NER. The majority of these approaches [12], [25], [26] employ a prototype-based metric, often k-nearest neighbor, to learn the representation of similar entities from different domains. However, in these approaches, the network parameters of the NER model cannot be updated, resulting in poor performance when adapting the model to a target domain with few available examples.

B. Prompt-based Learning

Recently, starting from GPT-3 [4], prompt-based learning has arisen to bridge the gap between the objectives of pre-training and fine-tuning. These approaches reformulate the downstream task by incorporating a template that transforms the input sentence into one that resembles the examples solved during pre-training. This strategy aims to fully apply acquired knowledge from pre-training to the downstream task. As stated in [27], a well-chosen prompt can be equivalent to hundreds of data points; hence, prompt-based learning can be highly advantageous for solving low-resource tasks. Another line of research studied a lightweight alternative to fine-tuning known as prompt-tuning, which optimizes a continuous task-specific vector as a prompt while leaving the parameters of the language model unchanged [28] [6]. However, the efficiency of prompt-tuning for complex sequence labeling tasks such as NER has yet to be verified.

Regarding NER, TemplateNER [11] is a template-based prompt method using BART [8] that treats the NER task as a language model ranking problem. This model manually creates a template for each class and separately populates each created template with all candidate entity spans extracted from the input sentence. The model then assigns a label to each entity candidate span based on the respective template score. In contrast to TemplateNER, instead of manually searching for an appropriate template, which is labor-intensive and time-consuming, we propose inserting some adjustable tokens into the template to search automatically for the ideal prompt template. LightNER [9] introduces prompt-tuning to the attention layer by incorporating continuous prompts into the attention layer. Moreover, LightNER constructs a unified semantic aware space to remove label-specific classifiers placed on top of encoders. Like LightNER, we eliminate the label-specific layers that map the generated answer to a class label. Instead of the attention layer, we insert soft prompts into the input sentence, and the creation of an answer space is no longer required.

C. Contrastive Learning

Recent works have studied contrastive learning for visual representation [29] [30], graph representations [31], and a variety of NLP tasks including sentence-level text representation [32] [33], relation extraction [34], machine translation [35], sentiment analysis [36], knowledge graph embeddings [37], caption generation [38]. Contrastive representation learning is intuitively similar to learning by comparison in that it aims to project similar samples close together in the embedding space while mapping dissimilar samples further apart [39]. Khosla et al. [40] study applying contrastive learning in a fully supervised setting and demonstrated that batch contrastive techniques outperform the cross-entropy loss and traditional contrastive losses, such as triplet [41], max-margin [42], and the N-pairs [43] loss. This work applies supervised batch contrastive learning to prompt-based few-shot NER to differentiate between different classes (entity types) in a sentence.

IV. Overview of ContrastNER

In this section, we introduce ContrastNER, our proposed prompt-based model for few-shot NER, using two main techniques: (1) soft-hard prompt tuning (IV-A) and (2) a verbalizer-free mechanism (IV-B). In the following subsections, we explain the details of ContrastNER (depicted in Figure 2).

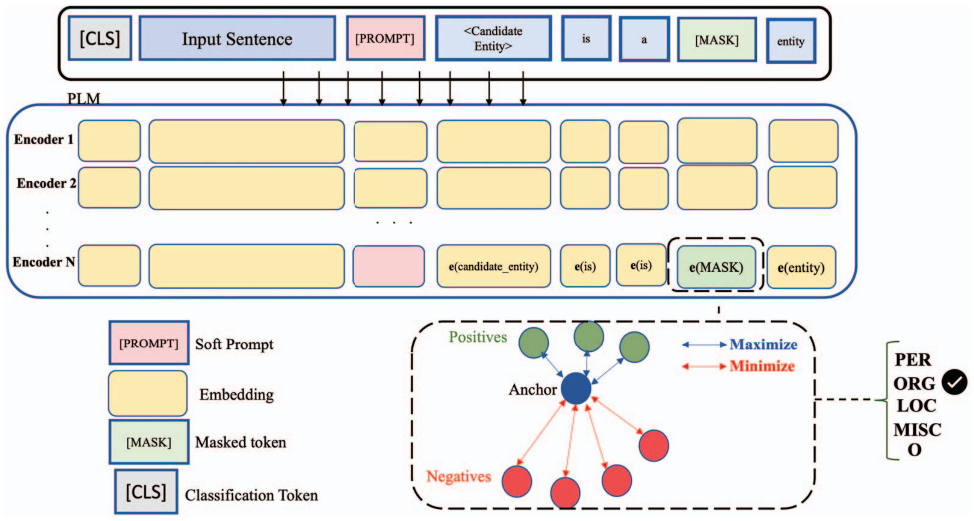


Fig. 2. Overview of ContrastNER.

A. Soft-Hard Prompt Tuning

Finding appropriate discrete prompt (a.k.a. *hard prompt*)[1] templates in natural language for NER models is challenging [5], [11], [44], [45]. Existing hard prompt-based NER models, such as LAMA [46], have shown that a single word change in prompts can cause an extreme difference in the results; hence an approach that is not sensitive to different discrete prompts would be advantageous in prompt-based learning models. As explained in Section II-C, continuous prompts (a.k.a. *soft prompts*) and tuning them using a prompt encoder is an approach to solve the hard prompt challenges [28] [6] [14]. However, this approach leads to two challenges: (1) it is still inferior to fine-tuning approach when the model size is not significant [47], and (2) adding a prompt encoder to learn the soft prompt results in learning extra parameters related to the prompt encoder.

To address these challenges, we propose the *soft-hard prompt*, where we use both soft (continuous) and hard (discrete) prompts in the fine-tuning paradigm. To this end, we transform a given input sentence $\mathbf{X} = \{x_1, \cdots, x_m\}$, into m new input sentences using a prompt template $\mathcal{T}$, such that each new sentence has three parts: (1) the original sentence $\mathbf{X}$, (2) a soft prompt $\{h_0 \cdots h_n\}$, and (3) a masked hard prompt. Suggested by [11], the masked hard prompt initialized as "$\langle$`candidate_entity`$\rangle$ `is a [MASK] entity`". For example, $\mathbf{X}$ = {`"Steve", "Jobs", "was", "born", "in", "America"`} will be transformed into six sentences as shown in Table I. To learn the hard token `[MASK]` and soft tokens $\{h_0 \cdots h_n\}$, we use a PLM $\mathcal{L}$ and pass $\mathcal{T}(X)$ as input to it. Then we optimize the soft tokens by the loss function L_S (Equation 3) computed as the cross-entropy between the actual entity type and the predicted `[MASK]` entity type by $\mathcal{L}$ (the details are in Section IV-B).

[1]Throughout the paper, we will interchangeably use *discrete prompt* and *hard prompt* but convey the same meaning as discrete prompt templates.

TABLE I: A sentence $\mathbf{X}$ and generated soft-hard prompts by $\mathcal{T}(\mathbf{X})$.

A sample input sentence X
`Steve Jobs was born in America`
Generated soft-hard prompts by $\mathcal{T}(\mathbf{X})$
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `Steve is a [MASK] entity.`
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `Jobs is a [MASK] entity.`
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `was is a [MASK] entity.`
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `born is a [MASK] entity.`
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `in is a [MASK] entity.`
`Steve Jobs was born in America` $[h_1]\cdots[h_p]$ `America is a [MASK] entity.`

B. Verbalizer-Free Mapping

Previous prompt-based approaches for NER often require a verbalizer to create a one-to-one mapping between the predicted token for `[MASK]` in a template and an entity type [13], which is computationally intensive. In addition, using a manually crafted verbalizer in few-shot NER, where the label space of the source and target domains may differ, leads to incompatibilities that negatively impact the model generalization [48]. Moreover, a verbalizer usually only considers the semantic relationship between the predicted token and a few words specified in the verbalizer (e.g., the *location* for `LOCATION` entity and the *person* for `PERSON` entity). However, the predicted token may have semantic relationships with other words (e.g., *place* for `LOCATION` entity, and *people* for `PERSON`), which the verbalizer will ignore.

To tackle the challenges of verbalizers mentioned above, we unify the models for learning the soft-hard prompt and predicting the entity type and propose a novel objective that includes supervised contrastive learning terms for fine-tuning a pre-trained language model. To do so, instead of forwarding the embedding of `[MASK]` through a verbalizer to detect the label (entity type), we use PLM $\mathcal{L}$ to directly predict the embedding of the label by fine-tuning it using a contrastive learning-based task [36], [39].

During training $\mathcal{L}$, we first transform all input examples within the training batch $\mathcal{B}$ using the soft-hard prompt template $\mathcal{T}$. Then, for each example $i \in \mathcal{B}$ (with label y_i), we select a set of *positive* examples with similar labels to i, $P(X) = \{i^+ \, |y_{i^+} = y_i, i^+ \in \mathcal{B}, i^+ \neq i\}$ and a set of *negative* examples with different labels, $N(i) = \{i^- \, |y_{i^-} \neq y_i, i^- \in \mathcal{B}, i^- \neq i\}$. Assume t_i denotes the embedding of the predicted label of [MASK] for the example i, and t_{i^+} and t_{i^-} are the embedding of the labels of positive and each negative examples, respectively, where $i^+ \in P(i)$ and $i^- \in N(i)$. Then, we use the contrastive model [32] to maximize the *within-class* similarity $sim(t_i, t_{i^+})$ of t_i and t_{i^+}, and minimize the *between-class* similarity $sim(t_i, t_{i^-})$ of t_i and t_{i^-}, where $sim(t_1, t_2)$ is the cosine similarity of t_1 and t_2. We define the contrastive learning loss function as below:

$$L_C = -\log \sum_{i^+ \in P(i)} \frac{e^{sim(t_i, t_{i^+})/\tau}}{e^{sim(t_i, t_{i^+})} + e^{sim(t_i, t_{i^-})/\tau}} \tag{1}$$

where τ is a temperature hyperparameter. In short, for each example i, the contrastive learning aims to learn the embedding of [MASK] and assign the appropriate label (entity type) to it by pulling semantically close examples with the same label together (positives) and pushing apart examples with a different label (negatives).

In case of having multiple examples in $P(i)$ and $N(i)$, we can rewrite the contrastive learning loss function as:

$$L_C = -\log \sum_{i^+ \in P(i)} \frac{e^{sim(t_i, t_{i^+})/\tau}}{\sum_{a \in A(i)} e^{sim(t_i, t_{i^-})/\tau}} \tag{2}$$

where $A(i)$ denotes a collection of all in-batch examples except i. Inspired by [49], to fine-tune $\mathcal{L}$'s parameters and update the soft prompt parameters, we define the overall loss function L as the weighted average of L_C and L_S, where L_S is a cross-entropy to update the soft prompt parameters:

$$L_S = -\sum_{i \in \mathcal{B}} y_i \cdot \log \mathcal{L}(y'_i | \mathcal{T}(i)) \tag{3}$$

$$L = \lambda L_C + (1 - \lambda) L_S P \tag{4}$$

where λ is a scalar weighting hyperparameter that we tune, and y_i and y'_i are the correct and the predicted labels of i, respectively.

During inference, we first transform all test instances into the format of the soft-hard prompt template and then take the predicted label embedding t_j of a test example j to generate the label (entity type) directly by comparing t_j to the k-nearest examples to t_j in the training set.

V. Evaluation

In this section, we conduct extensive experiments in standard and low-resource settings to evaluate ContrastNER and its effectiveness in the few-shot NER settings.

A. Datasets and Baselines

As a rich NER dataset, following [9] and [11], we used the English version of CoNLL03 [50] that includes four features (columns) for each sample: `id`, `tokens`, `pos_tags`, `chunk_tags` and `ner_tags`. The feature `tokens` represents the input sentence, and `ner_tags` specifies the type of mentioned entities in the input sentence, which contains four types of named entities: `LOCATION`, `PERSON`, `ORGANIZATION`, and `MISCELLANEOUS`. As low-resource datasets, we employed three datasets: (1) MIT Restaurant Review [51], (2) MIT Movie Review [51], and (3) Airline Travel Information Systems (ATIS) [52]. To evaluate the few-shot performance on NER datasets, we randomly sampled K instances per entity type from each low-resource dataset by setting K to 10, 20, 50, 100, 200, and 500. We then reported the average performance of five randomly sampled data splits to avoid dramatic changes for different data splits.

In our experiments, we compare ContrastNER with the following NER methods as baselines:

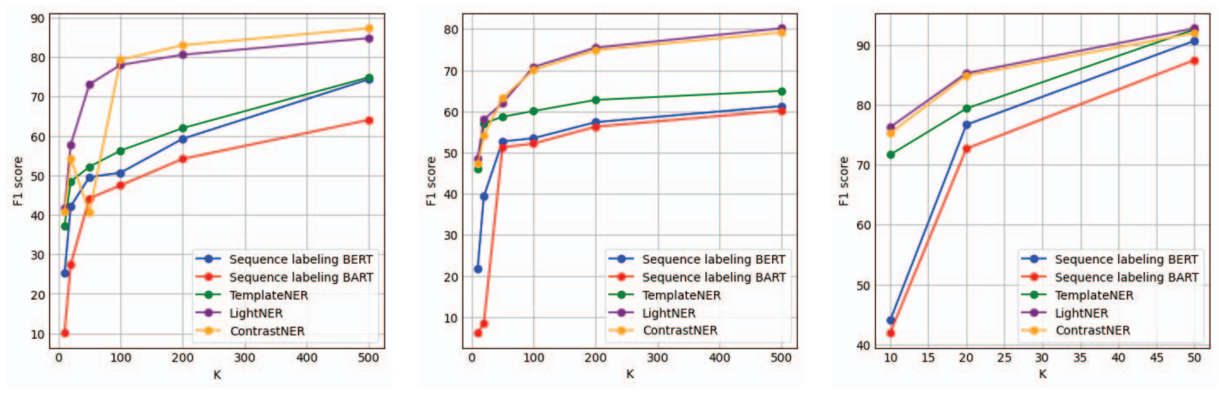
- Sequence Labeling BERT/BART [2]: Traditional sequence labeling methods where the pre-trained BERT and BART [8] models are employed to generate word sequence representations. A label-specific classifier is trained on the top of PLM to map the generated representations to entity types (labels).
- LUKE [3]: A transformer-based model with an entity-aware self-attention layer that generates contextualized word representations. Then, the pre-trained model is fine-tuned using entity typing downstream task and linear classifiers to predict the type of an entity in the given sentence.
- BART-NER [53]: A generative seq2seq method that converts the NER task into a unified sequence generation problem.
- TemplateNER [11] and LightNER [9]: Prompt-based models that use BART [8].

We conducted the experiments on a Tesla T4 GPU with 32 cores and 576 GB of RAM. We used RoBERTa [54], provided by Hugging Face[2], as the PLM $\mathcal{L}$ in our implementation. We set τ = 2, $\lambda = 0.5$, and trained the model using Adam optimizer [55] with a learning rate $5e-3$ and a batch size of 32.

B. Standard NER Setting

We first evaluated ContrastNER using the CoNLL03 [50], a rich NER dataset. Table II compares the results of ContrastNER and the baselines. As shown in Table II, although we developed ContrastNER for few-shot NER, it performs competitively in a rich-resource setting, showing the remarkable ability of our technique to identify the entities and their types in an input text.

[2]https://huggingface.co/models



(a) MIT Movies dataset (b) MIT Restaurants dataset (c) ATIS dataset

Fig. 3. Model performance (F1 score) in the cross-domain low-resource settings when the model was trained on the same domain.

TABLE II: Model performance on the CoNLL03 dataset. '†' shows the reported results with $BERT_{large}$ [2] since the result of the original publication is not achieved with the current version of the library (See the discussion at [56] and the reported results at [57]).

Traditional Models	$Precision$	$Recall$	$F1$
Sequence labeling BERT†	91.93	91.54	92.8
Sequence labeling BART	89.60	91.63	90.60
LUKE [3]	-	-	94.30
BART-NER [53]	92.61	93.87	93.24
Few-shot Friendly Models	$Precision$	$Recall$	$F1$
TemplateNER [11]	90.51	93.34	91.90
LightNER [9]	92.39	93.48	92.93
ContrastNER	91.04	93.44	92.22

TABLE III: In-domain Few-shot performance on the CoNLL03. * indicates it is a few-shot entity type.

Models	PERSON	ORGANIZATION	LOCATION*	MISCELLANEOUS*	Overall
Sequence labeling BERT	76.25	75.68	60.72	59.39	68.02
Sequence labeling BART	75.71	73.59	58.73	56.6	66.15
TemplateNER	84.49	72.61	71.98	73.37	75.59
LightNER	90.96	**76.88**	**81.57**	52.08	78.97
ContrastNER	**92.19**	75.79	73.98	**75.13**	**79.27**

C. *In-domain Few-shot NER Setting*

Following [11], we constructed a few-shot learning scenario on the CoNLL03 dataset, where the number of training samples for specific categories is limited by downsampling. Particularly, we considered PERSON and ORGANIZATION as the rich-resource entities and LOCATION and MISCELLANEOUS as the low-resource entities. The few-shot CoNLL03 training dataset contains 4237 training samples including 3836 PERSON, 1924 ORGANIZATION, 100 MISCELLANEOUS, and 100 LOCATION. Table III indicates that ContrastNER outperforms TemplateNER [11] and LightNER [9] by 3.68 and 0.3 $F1$ score, respectively. Moreover, ContrastNER achieves 73.98 and 75.13 $F1$ scores in few-shot LOCATION and MISCELLANEOUS, which is highly competitive with the best-reported result. The illustrated performance proves the effectiveness of our approach in in-domain few-shot NER.

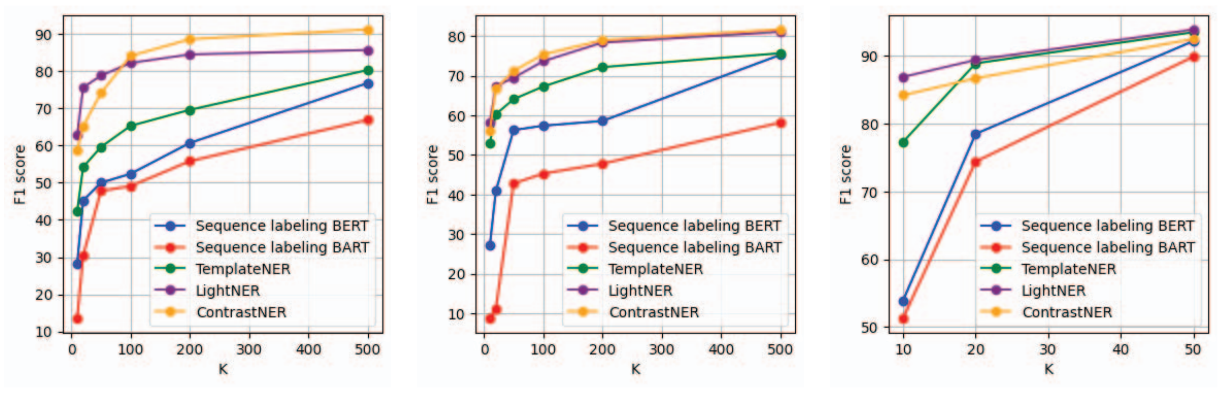
D. *Cross-Domain Few-Shot Setting*

Finally, we evaluated the model performance in scenarios where the class label sets and textual sentences vary from the source domain and only limited labeled data are available for training. Specifically, we randomly sampled a specific number of instances per entity type from the training set as the training data in the target domain to simulate the cross-domain low-resource data scenarios. We first considered direct training on the target domain from scratch without available source domain data. Figure 3 depicts the results of training models directly on target domains and evaluation on the same domain. According to the results, compared to sequence labeling with BERT, BART, and TemplateNER, ContrastNER's results appear more consistent. ContrastNER outperforms these methods, suggesting it can exploit few-shot data better. For example, we achieved an $F1$ score of 70.6 in the 50-shot setting, which is higher than the results of sequence labeling with BERT and BART, and TemplateNER in the 200-shot setting.

We then investigated the amount of knowledge that can be transferred from training on ConLL03 dataset. In this setting, we trained the model on the news domain (ConLL03) and then tested it on different domains. Figure 4 shows the results of training models on the CoNLL03 dataset as a generic domain and its evaluations on other target domains. As can be seen, prompt-based methods outperform the traditional sequence labeling methods regardless of how much training data is provided. Among the prompt-based approaches, ContrastNER indicates the best performance overall. Compared to LightNER, the best-performing NER framework among all state-of-the-art models, it can be witnessed that ContrastNER can compete with this framework in different few-shot settings. At the same time, our approach does not require prompt engineering and verbalizer design. In particular, ContrastNER outperforms LightNER in scenarios with more than 100 examples for each entity label. In other words, the contrastive loss is effective after a certain threshold of training data is reached.

E. *Effectiveness of Soft-hard Prompt*

In ContrastNER, we applied the soft-hard prompt tuning technique that combines the soft and hard prompts. Their automatic search in a mixed space of continuous and discrete



(a) MIT Movies dataset (b) MIT Restaurants dataset (c) ATIS dataset

Fig. 4. Model performance (F1 score) in the cross-domain low-resource settings when the model was trained on the CoNLL03 source domain and then evaluated on the target domains.

prompts eliminates the manual effort of prompt engineering. To investigate the effectiveness of our approach, we employed four different hard prompt templates, which are manually designed and used in TemplateNER [11] on the ConLL03 [50] development set (Table IV). Figure 5 represents how selecting various discrete templates affects the performance of ContrastNER and TemplateNER based on $F1$ score. As can be observed, our model produces more stable results, and unlike TemplateNER, the discrete template in ContrastNER does not significantly influence the final performance. Since the soft tokens are adjusted during training, there is a slight variation in the model's performance when the discrete prompt template is changed, demonstrating the efficiency of using soft-hard prompts.

TABLE IV: Different discrete prompts applied on CoNLL03.

	Discrete Template
Template1	`<candidate_entity>` is a `<entity_type>` entity.
Template2	The entity type of `<candidate_entity>` is `<entity_type>`.
Template3	`<candidate_entity>` belongs to `entity_type>` category.
Template4	`<candidate_entity>` should be tagged as `<entity_type>`.

F. Discussion

This study investigated a prompt-based method in the few-shot NER problem. The results generally indicate that prompt-based NER approaches outperform the traditional NER frameworks in few-shot settings where only limited data is provided to train the model. Although prompt-based methods are specifically designed for low-resource scenarios, they also perform competitively with traditional methods in rich-resource scenarios. Determining how well previous prompt-based NER systems perform depends on discovering the best-performing prompt template (prompt engineering) or encountering the best label word space and their mapping to actual class labels (verbalizer engineering). At the same time, ContrastNER exhibits more consistent performance with various discrete prompt templates and eliminates the need to determine the best-performing prompt template and optimal verbalizer manually. Despite this elimination, according to the results, it is relatively straightforward that ContrastNER outperforms baselines in the in-domain few-shot scenario. Moreover, ContrastNER indicates competitive performance in cross-domain scenarios. Due to the contrastive approach, ContrastNER can likely beat the baselines after reaching a certain threshold of training data. It should be noted that this study is primarily concerned with removing the need for manual prompt and verbalizer engineering in the few-shot NER problem.

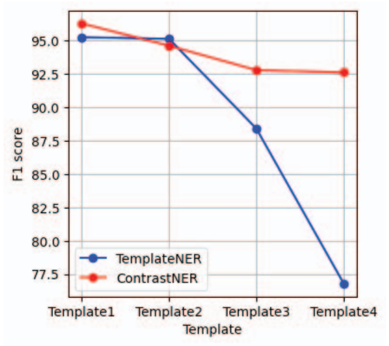


Fig. 5. Model performance ($F1$ score) on the CoNLL03 development set using different discrete prompt templates. '†' shows the reported results in the original work.

VI. Conclusion and Future Work

This paper presents ContrastNER, a prompt-based learning framework for few-shot NER without manual prompt engineering and design of verbalizers using RoBERTa [54] as the backbone model. We present soft-hard prompt tuning for automatic prompt search in a mixed space of continuous and discrete prompts to avoid manual prompt engineering to find the best-performing prompt template. We also employ contrastive learning-based loss to unify learning the soft-hard prompt and predicting the entity type without manually designing a verbalizer. Our experiment results show that

ContrastNER indicates a competitive performance on both rich-resource and few-shot NER. In the future, we plan to extend ContrastNER to the relation extraction task to develop a unified prompt-based learning framework for information extraction. Moreover, we will apply ContrastNER on actual datasets from the DataCloud project, which is a project on defining and managing Big Data pipelines in different applications including digital health systems, autonomous live sports content, and manufacturing analytics. We are going to extract and store structured data from the definitions of Big Data pipelines defined in the form of unstructured natural language.

References

[1] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto, "Luke: Deep contextualized entity representations with entity-aware self-attention," *arXiv preprint arXiv:2010.01057*, 2020.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[5] T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," *arXiv preprint arXiv:2012.15723*, 2020.

[6] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang, "Gpt understands, too," *arXiv preprint arXiv:2103.10385*, 2021.

[7] T. Schick and H. Schütze, "Exploiting cloze questions for few shot text classification and natural language inference," *arXiv preprint arXiv:2001.07676*, 2020.

[8] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[9] X. Chen, N. Zhang, L. Li, X. Xie, S. Deng, C. Tan, F. Huang, L. Si, and H. Chen, "Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner," *arXiv preprint arXiv:2109.00720*, 2021.

[10] A. T. Liu, W. Xiao, H. Zhu, D. Zhang, S.-W. Li, and A. Arnold, "Qaner: Prompting question answering models for few-shot named entity recognition," *arXiv preprint arXiv:2203.01543*, 2022.

[11] L. Cui, Y. Wu, J. Liu, S. Yang, and Y. Zhang, "Template-based named entity recognition using bart," *arXiv preprint arXiv:2106.01760*, 2021.

[12] M. Ziyadi, Y. Sun, A. Goswami, J. Huang, and W. Chen, "Example-based named entity recognition," *arXiv preprint arXiv:2008.10570*, 2020.

[13] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[14] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," *arXiv preprint arXiv:2104.08691*, 2021.

[15] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] J. Wu, "Introduction to convolutional neural networks," *National Key Lab for Novel Software Technology. Nanjing University. China*, vol. 5, no. 23, p. 495, 2017.

[18] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *arXiv preprint arXiv:1603.01354*, 2016.

[19] J. P. Chiu and E. Nichols, "Named entity recognition with bidirectional lstm-cnns," *Transactions of the association for computational linguistics*, vol. 4, pp. 357–370, 2016.

[20] H. Chen, Z. Lin, G. Ding, J. Lou, Y. Zhang, and B. Karlsson, "Grn: Gated relation network to enhance convolutional neural network for named entity recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 6236–6243.

[21] S. Ilić, E. Marrese-Taylor, J. A. Balazs, and Y. Matsuo, "Deep contextualized word representations for detecting sarcasm and irony," *arXiv preprint arXiv:1809.09795*, 2018.

[22] Y. Luo, F. Xiao, and H. Zhao, "Hierarchical contextualized representation for named entity recognition," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 8441–8448.

[23] A. Fritzler, V. Logacheva, and M. Kretov, "Few-shot classification in named entity recognition task," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 993–1000.

[24] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *International conference on learning representations*, 2017.

[25] J. Li, S. Shang, and L. Shao, "Metaner: Named entity recognition with meta-learning," in *Proceedings of The Web Conference 2020*, 2020, pp. 429–440.

[26] N. Ding, G. Xu, Y. Chen, X. Wang, X. Han, P. Xie, H.-T. Zheng, and Z. Liu, "Few-nerd: A few-shot named entity recognition dataset," *arXiv preprint arXiv:2105.07464*, 2021.

[27] T. L. Scao and A. M. Rush, "How many data points is a prompt worth?" *arXiv preprint arXiv:2103.08493*, 2021.

[28] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," *arXiv preprint arXiv:2101.00190*, 2021.

[29] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[30] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *Ieee Access*, vol. 8, pp. 193 907–193 934, 2020.

[31] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax." *ICLR (Poster)*, vol. 2, no. 3, p. 4, 2019.

[32] T. X. Y. Gao and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[33] D. Iter, K. Guu, L. Lansing, and D. Jurafsky, "Pretraining with contrastive sentence objectives improves discourse performance of language models," *arXiv preprint arXiv:2005.10389*, 2020.

[34] N. Ding, X. Wang, Y. Fu, G. Xu, R. Wang, P. Xie, Y. Shen, F. Huang, H.-T. Zheng, and R. Zhang, "Prototypical representation learning for relation extraction," *arXiv preprint arXiv:2103.11647*, 2021.

[35] Z. Yang, Y. Cheng, Y. Liu, and M. Sun, "Reducing word omission errors in neural machine translation: A contrastive learning approach," 2019.

[36] Z. C. W. M. Q. F. L. R. X. S. H. Xu and J. Huang, "Making pre-trained language models end-to-end few-shot learners with contrastive prompt tuning," *arXiv preprint arXiv:2204.00166*, 2022.

[37] A. J. Bose, H. Ling, and Y. Cao, "Adversarial contrastive estimation," *arXiv preprint arXiv:1805.03642*, 2018.

[38] R. Vedantam, S. Bengio, K. Murphy, D. Parikh, and G. Chechik, "Context-aware captions from context-agnostic supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 251–260.

[39] S. C. Hadsell, Raia and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. of the CVPR'06*, vol. 2. IEEE, 2006, pp. 1735–1742.

[40] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in neural information processing systems*, vol. 33, pp. 18 661–18 673, 2020.

[41] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[42] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," *arXiv preprint arXiv:1612.02295*, 2016.

[43] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," *Advances in neural information processing systems*, vol. 29, 2016.

[44] Z. F. F. X. J. A. Jiang and G. Neubig, "How can we know what language models know?" *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 423–438, 2020.

[45] L. Reynolds and K. McDonell, "Prompt programming for large language models: Beyond the few-shot paradigm," in *Extended Abstracts of the*

2021 CHI Conference on Human Factors in Computing Systems, 2021, pp. 1–7.

[46] T. L. P. B. A. W. Y. M. A. H. . R. Petroni, F.; Rocktäschel, "Language models as knowledge bases?" *arXiv preprint arXiv:1909.01066*, 2019.

[47] X. K. J. Y. F. Z. D. Z. Y. Liu and J. Tang, "P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks," *arXiv preprint arXiv:2110.07602*, 2021.

[48] T. Schick and H. Schütze, "It's not just size that matters: Small language models are also few-shot learners," *arXiv preprint arXiv:2009.07118*, 2020.

[49] J. C. A. . S. V. Gunel, B.; Du, "Supervised contrastive learning for pre-trained language model fine-tuning," *arXiv preprint arXiv:2011.01403*, 2020.

[50] E. F. Sang and F. D. Meulder, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," *arXiv preprint cs/0306050*, 2003.

[51] P. C. S. . G. J. Liu, J.; Pasupat, "Asgard: A portable architecture for multilingual dialogue systems," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8386–8390.

[52] G. C. A. C. Y. G. J. D. L. Hakkani-Tür, D.; Tür and Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm." in *Interspeech*, 2016, pp. 715–719.

[53] T. D. J. G. Q. Z. Z. Yan, H.; Gui and X. Qiu, "A unified generative framework for various ner subtasks," *arXiv preprint arXiv:2106.01223*, 2021.

[54] M. O. N. G. J. D. M. J. D. C. O. L. M. L. L. Z. Liu, Yinhan and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[55] D. P. Kingma and J. Ba., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[56] "MS Windows NT kernel description," https://github.com/google-research/bert/issues/223, accessed: 2022-11-10.

[57] B. T. Akbik, A. and R. Vollgraf, "Pooled contextualized embeddings for named entity recognition," in *Proc. of NAACL*, 2019, pp. 724–728.