Automatic Creation of Named Entity Recognition Datasets by Querying Phrase Representations

Hyunjae Kim[1]   **Jaehyo Yoo**[1]   **Seunghyun Yoon**[2]   **Jaewoo Kang**[1,3]

[1]Korea University   [2]Adobe Research   [3]AIGEN Sciences

{hyunjae-kim,jaehyoyoo,kangj}@korea.ac.kr

syoon@adobe.com

Abstract

Most weakly supervised named entity recognition (NER) models rely on domain-specific dictionaries provided by experts. This approach is infeasible in many domains where dictionaries do not exist. While a phrase retrieval model was used to construct pseudo-dictionaries with entities retrieved from Wikipedia automatically in a recent study, these dictionaries often have limited coverage because the retriever is likely to retrieve popular entities rather than rare ones. In this study, we present a novel framework, HighGEN, that generates NER datasets with high-coverage pseudo-dictionaries. Specifically, we create entity-rich dictionaries with a novel search method, called phrase embedding search, which encourages the retriever to search a space densely populated with various entities. In addition, we use a new verification process based on the embedding distance between candidate entity mentions and entity types to reduce the false-positive noise in weak labels generated by high-coverage dictionaries. We demonstrate that HighGEN outperforms the previous best model by an average F1 score of 4.7 across five NER benchmark datasets.

1 Introduction

Named entity recognition (NER) models often require a vast number of manual annotations for training, which limits their utility in practice. In several studies, external resources such as domain-specific dictionaries have been employed as weak supervision to reduce annotation costs (Shang et al., 2018; Liang et al., 2020; Meng et al., 2021). However, such dictionaries exist only for certain domains and building a dictionary for a new domain requires a high level of expertise and effort.

To address this problem, a recent study proposed a framework called GeNER, which generates NER datasets without hand-crafted dictionaries (Kim et al., 2022). In GeNER, user questions that reflect the needs for NER are received as inputs
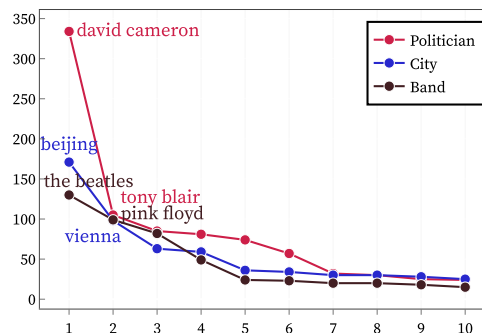

Figure 1: The most frequent ten entities in the top 1,000 phrases retrieved from the 2018-12-20 version of Wikipedia for the three questions: *"Which politician?"*, *"Which city?"*, and *"Which band?"*. Due to a bias in the entity popularity (Chen et al., 2021), a current phrase retrieval model, DensePhrases (Lee et al., 2021), primarily returns popular entities, limiting the coverage of dictionaries.

(e.g., *"Which city?"*), and an open-domain question-answering (QA) system, DensePhrases (Lee et al., 2021), is used to retrieve relevant phrases (i.e., answers) and evidence sentences from Wikipedia. The retrieved phrases constitute a 'pseudo' dictionary, which serves as weak supervision in place of hand-crafted dictionaries. The evidence sentences are annotated based on string matching with the pseudo dictionary, resulting in the final dataset. This approach allows NER models to adapt to new domains for which training data are scarce and domain-specific dictionaries are unavailable.

However, because the entity popularity of Wikipedia is biased (Chen et al., 2021; Leszczynski et al., 2022), existing open-domain QA models tend to retrieve popular entities rather than rare ones. This limits the coverage of dictionaries generated by GeNER. Figure 1 shows examples of a bias in the entity population retrieved from the open-domain QA model. "David Cameron," "Beijing," and "The Beatles" frequently appear in the top 1,000 retrieved phrases for each type of ques-

tion. Low-coverage dictionaries created from these biased results can cause incomplete annotations (i.e., false-negative entities), which impedes the training of NER models. Unfortunately, increasing the number of retrieved phrases (i.e., larger top-k) is not an appropriate solution because it is computationally inefficient and causes a high false-positive rate in the dictionary. Therefore, a new search method that can efficiently retrieve diverse entities with a reasonable top-k and a new NER dataset generation framework based on this search method are needed.

In this study, we present HighGEN, an advanced framework for generating NER datasets with automatically constructed 'high-coverage' dictionaries. Specifically, we first obtain phrases and sentences and constitute an initial dictionary in a similar manner to GeNER. Subsequently, we expand the initial dictionary using a *phrase embedding search*, in which the embeddings of the retrieved phrases are averaged to re-formulate query vectors. These new queries specify contexts in which different entities of the same type appear, allowing our retriever to search over a vector space in which various entities are densely populated.[1] The expanded dictionary is used to annotate the retrieved sentences. Because a larger dictionary can induce more false-positive annotations during rule-based string matching, we introduce a new verification process to ensure that weak labels annotated by the string matching are correct. The verification process is performed by comparing the distance between the embeddings of a candidate entity and the target entity type.

We trained recent NER models (Liu et al., 2019; Lee et al., 2020; Liang et al., 2020; Meng et al., 2021) with the datasets generated by HighGEN and evaluated the models on five datasets. Our models outperformed the baseline models trained using the previous best model GeNER by an average F1 score of 4.7 (Section 4). In addition, we show an additional advantage of HighGEN over GeNER, which generates datasets using only a few hand-labeled examples without input user questions. HighGEN outperformed few-shot NER models on two datasets (Section 5). Finally, we perform an analysis of the factors affecting the retrieval diversity and NER performance (Section 6). We make the following contributions:[2]

- We propose a HighGEN framework that generates NER datasets with entity-rich dictionaries that are automatically constructed from unlabeled Wikipedia corpus.

- We present two novel methods in HighGEN: (i) phrase embedding search to overcome the limitations of the current open-domain phrase retriever and successfully increase the entity recall rate and (ii) distance-based verification to effectively reduce the false-positive noise in weak labels.

- HighGEN outperformed the previous-best weakly-supervised model GeNER by an F1 score of 4.7 on five datasets. In few-shot NER, HighGEN created datasets using few-shot examples as queries and outperformed current few-shot NER models on two datasets.

2 Preliminaries

2.1 Weakly Supervised NER

The aim of NER is to identify named entities in text and classify them into predefined entity types. Let $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$ be a dataset, where $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$ is a list of N unlabeled sentences and $\mathbf{Y} = \{\mathbf{y}_n\}_{n=1}^N$ is a list of N corresponding token-level label sequences. While supervised learning relies on the human-annotated labels, $\mathbf{Y}$, to train models, in weakly supervised NER, the weak labels $\hat{\mathbf{Y}}$ are generated using string matching between a domain-specific dictionary, $\mathcal{V}$, built by experts and the unlabeled sentences, $\mathbf{X}$ (Yang et al., 2018; Shang et al., 2018; Peng et al., 2019; Cao et al., 2019; Yang and Katiyar, 2020; Liang et al., 2020; Meng et al., 2021). Hand-crafted labeling rules are utilized in another line of studies (Fries et al., 2017; Ratner et al., 2017; Safranchik et al., 2020; Zhao et al., 2021); however, these rules are difficult to apply to new entity types. Recently, Kim et al. (2022) proposed GeNER, in which weak labels are generated with a pseudo-dictionary, $\hat{\mathcal{V}}$, created using a phrase retrieval model. We follow their approach but present an advanced framework for addressing the low-coverage problem and obtaining more entity-rich dictionaries and NER datasets.

2.2 DensePhrases

DensePhrases (Lee et al., 2021) is a phrase retrieval model that finds relevant phrases for natural language inputs in a Wikipedia corpus. Unlike the

---

[1] We provide an explanation of its working principle (Section 3.2) and an analysis of its retrieval diversity (Section 6.1).

[2] We will make our code publicly available.

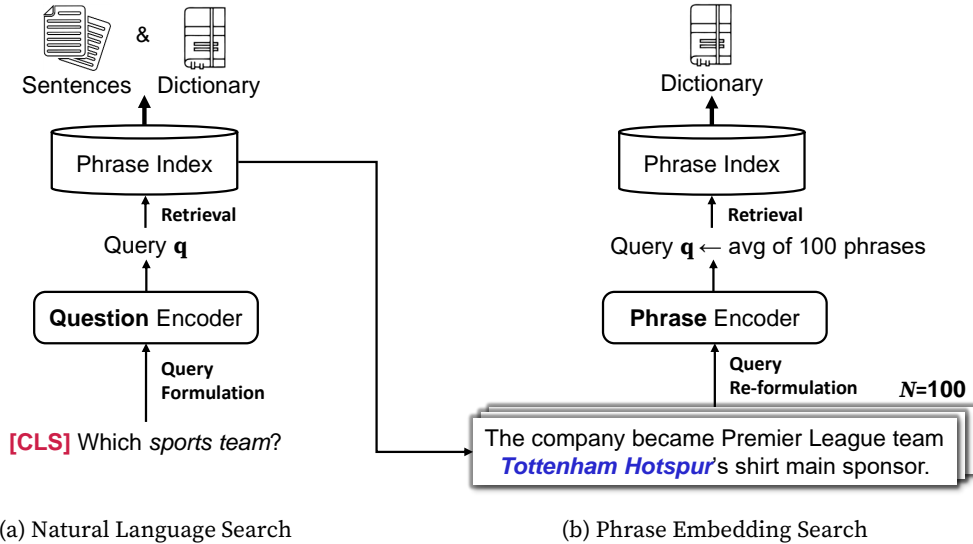**(a) Natural Language Search** **(b) Phrase Embedding Search**

Figure 2: Overview of the natural language search and phrase embedding search in our HighGEN framework. (a) Natural language search retrieves unlabeled sentences ($\hat{\mathbf{X}}_1$) and an initial dictionary ($\hat{\mathcal{V}}_1$) for the given question, "*Which sports team?*" (Section 3.1). (b) Phrase embedding search further retrieves an additional dictionary ($\hat{\mathcal{V}}_2$) using the top 100 phrases retrieved from the natural language search (Section 3.2). Note that the retrieved sentences and phrases are fed into the dictionary matching and verification stage (see Section 3.3 and Figure 3).

retriever-reader approach, which first retrieves evidence passages from Wikipedia and then finds the answer (Chen et al., 2017), DensePhrases retrieves answers directly from dense phrase vectors of the entire English Wikipedia as follows:

$$
\begin{aligned}
\mathbf{s} &= E_s(s, x), \quad \mathbf{q} = E_q(q), \\
(s^*, x^*) &= \arg\max_{(s,x)\in\mathcal{W}}(\mathbf{s}^\top \mathbf{q}),
\end{aligned} \quad (1)
$$

where s is a phrase, a sequence of words from evidence text x (i.e., sentence, passage, etc.); $\mathcal{W}$ is the set of all phrase-evidence pairs in Wikipedia. The input question q is converted into the query vector $\mathbf{q}$ by the question encoder, E_q. Subsequently, relevant phrases are retrieved based on the similarity scores between the query vector $\mathbf{q}$ and phrase vector $\mathbf{s}$, which is represented as the concatenation of the start and end vectors of the phrase, produced by the phrase encoder, E_s. All phrase vectors are 'pre-indexed' before inference, which greatly improves run-time efficiency (Seo et al., 2019; Lee et al., 2021). In the context of weakly supervised NER, DensePhrases can be used as a database to obtain candidate entities for specific NER needs, along with sentences to construct the final NER corpus (Kim et al., 2022).

2.3 Entity Popularity Bias

Chen et al. (2021) found that current document retrievers exhibit entity popularity bias in which the models prefer popular entities over rare ones and encounter problem in disambiguating entities in open-domain tasks. For instance, the models returned documents related to the company Apple for questions about the British rock band Apple or the 1980 film The Apple. Similarly, we raised the problem that phrase retrievers mainly provide popular entities for NER owing to the biased nature of Wikipedia in terms of entity popularity, which limits the coverage of dictionaries.

3 Method

HighGEN comprises three stages of natural language search, phrase embedding search (Figure 2), and dictionary matching and verification (Figure 3). We highlight that the natural language search is similarly used in GeNER, but the last two stages are novel and first proposed in our study.

3.1 Natural Language Search

Query formulation. Let $T = \{t_1, ..., t_L\}$ be a set of L target entity types. The concrete needs for these entity types are translated into simple questions. The questions follow the template of "*Which* [TYPE]*?*," where the [TYPE] token is substituted for each entity type of interest. For instance, the question is formulated as "*Which* city*?*" if the target entity type t is city.

Natural Language Search
Q: Which actor?
[1] . . . including Best British Film, Best British Director for Danny Boyle and Best British Actor for *Ewan McGregor*.
[2] His first movie role was in "The Detective," which starred *Frank Sinatra*.
Q: Which athlete?
[1] The nation's most famous Olympic athlete is *Eric Moussambani*, who achieved some international notoriety for . . .
[2] *Donovan Bailey* holds the men's world record with a time of 5.56 seconds and Irina Privalova holds the women's . . .
Phrase Embedding Search
Q: Which actor?
[1] *Owen Ash Weingott* (21 June 1921 - 2013 12 October 2002) was an Australian actor and director although . . . ,
[2] *Ron Vawter* (December 9, 1948 - 2013 April 16, 1994) was an American actor and a founding member of . . . ,
Q: Which athlete?
[1] *Yuri Floriani* (born 25 December 1981) is an Italian steeplechase runner.
[2] *Jeremy Porter Linn* (born January 6, 1975) is an American former competition swimmer, Olympic medalist, and . . .

Table 1: Comparison of context diversity of sentences retrieved using natural language search and phrase embedding search for the two questions. Sentences by the phrase embedding search tend to have similar patterns.

Retrieval. Input questions are fed into the phrase retrieval model, DensePhrases, to retrieve the top k phrases s^* and sentences x^* (see Section 2.2). For L different questions, a total of $k_1 + \cdots + k_L$ sentences are used as the unlabeled sentences, $\hat{\mathbf{X}}_1$. The retrieved phrases are used as the pseudo-dictionary, $\hat{\mathcal{V}}_1$, which comprises phrase s and corresponding type t pairs (e.g., Beijing–city).

3.2 Phrase Embedding Search

Query re-formulation. As mentioned in Section 1, the coverage of the initial dictionary $\hat{\mathcal{V}}_1$ is often limited because of the entity popularity bias. Our solution to search for diverse entities is very simple. We re-formulate queries by averaging the phrase vectors as follows:

$$\mathbf{q} = \frac{1}{N} \sum_{n=1}^{N} E_s(s_n, x_n), \qquad (2)$$

where s_n and x_n are the n-th top phrase and corresponding sentence from the natural language search. We used only the top 100 phrases for each question (i.e., $N = 100$) because a larger number of phrases did not improve retrieval quality in our initial experiments.

Retrieval. For L new queries obtained by Equation (2), a total of $k'_1 + \cdots + k'_L$ phrases are additionally retrieved by Equation (1) and constitute a new dictionary $\hat{\mathcal{V}}_2$. Subsequently, we merge $\hat{\mathcal{V}}_1$ and $\hat{\mathcal{V}}_2$ to obtain the final dictionary $\hat{\mathcal{V}}$. Note that we do not use the retrieved sentences $\hat{\mathbf{X}}_2$ because we found using only $\hat{\mathbf{X}}_1$ as the final unlabeled sentences (i.e., $\hat{\mathbf{X}}$) resulted in better NER performance.[3]

---

[3] A related analysis is included in Section 6.2.

Interpretation. Natural language search results in the retriever performing 'broad' searches for all the Wikipedia contexts relevant to the target entity class. In contrast, phrase embedding search, which averages phrase vectors of the same entity type, can be viewed as providing prompts that implicitly represent certain contextual patterns in which entities of the target class often appear. Having the retriever perform 'narrow' searches by focusing on specific contexts leads to a wide variety of entities with less bias towards popular ones. This is because (1) the same entities rarely appear repeatedly in a specific context, (2) whereas different entities of the same type frequently appear in a similar context as they are generally interchangeable.

Our qualitative analysis supports our claim above. We retrieved 5k sentences using two questions, "*Which actor?*" and "*Which athlete?*", and manually analyzed 100 sentences sampled from them. Table 1 shows that sentences by the phrase embedding search exhibit clear patterns in their contexts, whereas those by the natural language search do not. Specifically, 91 and 94 of the 100 sentences for the actor and athlete types had similar patterns, respectively. Further analysis shows that this property of the phrase embedding search contributes significantly to improving entity diversity (Section 6.1) and NER performance (Section 6.2).

3.3 Dictionary Matching & Verification

Dictionary matching. After $\hat{\mathbf{X}}$ and $\hat{\mathcal{V}}$ are obtained, dictionary matching is performed to generate weak labels, $\hat{\mathbf{Y}}$. Specifically, if a string in the unlabeled sentence matches an entity name in the dictionary, the string is labeled with the corre-
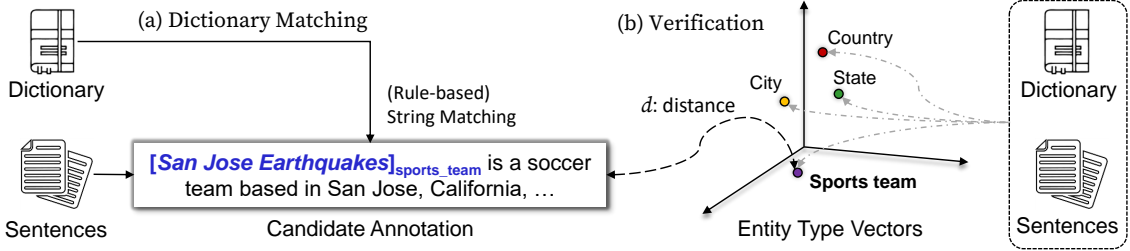
Figure 3: Overview of the (a) dictionary matching and (b) verification stage in our HighGEN framework. This annotates sentences based on the vector distance between a candidate entity (e.g., "San Jose Earthquakes") and entity type (e.g, "sport team") that is represented as the average of phrase vectors in the retrieval results.

sponding entity type. However, this method cannot handle *label ambiguity* inherent in entities[4] because it relies only on lexical information without leveraging contextual information of phrases. The false-positive noise due to label ambiguity is amplified as the dictionary size increases, making it difficult to effectively use our expanded dictionary $\hat{\mathcal{V}}$.

Verification. Candidate annotations provided by dictionary matching are passed to the verification stage. Let e be a matched string in the sentence and $\bar{T}$ be the matched entity types (a subset of T). The verification function $\mathcal{L}$ is defined as follows:

$$\mathcal{L}(e, \bar{T}) = \begin{cases} t^* & \text{if } d(\mathbf{e}, \mathbf{t}^*) < \tau, \\ \text{not entity} & \text{otherwise,} \end{cases} \quad (3)$$

$$t^* = \arg\min_{t_l \in \bar{T}} d(\mathbf{e}, \mathbf{t}_l), \ \mathbf{t}_l = \frac{1}{k_l} \sum_{n=1}^{k_l} E_s(s_n, x_n),$$

where d is the Euclidean distance function; $\mathbf{e}$ is the phrase vector of the candidate string; $\mathbf{t}_l$ is the l-th type vector; τ is the cut-off value. The string is labeled with the nearest type t^*, or unlabeled if the distance is higher than the cut-off value. The type vector is calculated by averaging all the retrieved phrase vectors of the entity type, based on the assumption that the mean vector of phrases is a good representative of the entity class. In addition, the cut-off value is also calculated using phrase vectors. Specifically, the function d computes the distance scores between the type vector t_l and all the phrase vectors of the type. The distribution of the distance scores is then standardized, and the score of 'z' times the standard deviation from the mean is used as the cut-off value (e.g., $z = 3$).[5]

---

[4]Even the same string can be labeled with different entity types depending on the context, leading to label ambiguity. For instance, "Liverpool" could be a city or a sports team.

[5]The distribution of the distance scores is generally balanced; thus, we used a usual method to compute the cut-off value without any other tricks to balance the distribution.

4 Experiments

In this experiment, it was assumed that human-annotated datasets did not exist; thus, our models were trained only using synthetic data $\{\hat{\mathbf{X}}, \hat{\mathbf{Y}}\}$ by HighGEN. To avoid excessive hyperparameter search, we used the same sets of input questions and the same number of sentences for each question (i.e., $k_1, \ldots, k_L$) as those used in the previous study (Kim et al., 2022). A new hyperparameter introduced in HighGEN, the number of phrases retrieved by phrase embedding search (i.e., $k'_1, \ldots, k'_L$), was set to 30k. Please refer to in Appendix A for the full list of hyperparameters and implementation details. For metrics, the entity-level precision, recall, and F1 scores were used (Tjong Kim Sang and De Meulder, 2003).

4.1 Datasets

We used five datasets from four domains. Following Kim et al. (2022), we did not use the *MISC* and *other* classes because they are vague to represent with some user questions. (i) CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) consists of Reuters news articles with three coarse-grained entity types of person, location, and organization. (ii) Wikigold (Balasuriya et al., 2009) is a small-size dataset that consists of Wikipedia documents with the same entity types as CoNLL-2003. (iii) WNUT-16 (Strauss et al., 2016) consists of nine entity types annotated in tweets, such as TV show, movie, and musician. (iv) Two biomedical domain datasets, NCBI-disease (Doğan et al., 2014) and BC5CDR (Li et al., 2016), are collections of PubMed abstracts with manually annotated diseases (NCBI-disease) or disease and chemical entities (BC5CDR). The benchmark statistics are listed in Table B.2 (Appendix).

Model	CoNLL-2003	Wikigold	WNUT-16	NCBI-disease	BC5CDR
Full Dictionary					
+ Standard	74.4 (80.5/69.1)	54.9 (53.8/56.1)	45.3 (44.3/46.2)	66.6 (67.5/65.7)	79.7 (82.8/76.8)
+ BOND	83.5 (82.1/84.9)	55.7 (46.0/70.8)	35.0 (30.6/40.9)	67.0 (63.7/70.6)	81.1 (76.6/86.1)
+ RoSTER	85.8 (84.3/87.3)	73.1 (67.1/80.2)	28.9 (43.1/21.8)	74.3 (75.9/72.7)	80.7 (78.6/83.0)
Pseudo-dictionary					
GeNER					
+ Standard	56.3 (72.9/45.8)	41.3 (58.6/31.9)	36.5 (41.3/32.6)	45.9 (59.0/37.6)	64.9 (76.6/56.3)
+ BOND	64.5 (70.7/59.3)	59.5 (65.2/54.7)	42.1 (36.7/49.5)	67.0 (70.8/63.5)	69.3 (69.0/69.7)
+ RoSTER	67.8 (77.9/60.0)	55.8 (66.9/47.9)	51.8 (49.1/54.8)	71.0 (74.1/68.1)	72.1 (74.6/69.7)
HighGEN (**Ours**)					
+ Standard	58.0 (73.3/48.0)	43.6 (59.5/34.4)	38.5 (42.2/35.4)	53.3 (66.4/44.6)	72.2 (77.9/67.3)
+ BOND	66.0 (65.5/66.5)	**68.2** (67.2/69.2)	40.2 (32.6/52.3)	70.2 (72.9/67.6)	72.9 (69.5/76.7)
+ RoSTER	**73.3** (78.5/68.7)	67.5 (68.5/66.5)	**53.4** (49.0/58.8)	**73.2** (77.4/69.4)	**74.6** (73.3/76.0)
HighGEN + RoSTER (for ablation study)					
w/o $\mathcal{L}$	70.6 (68.2/73.1)	65.7 (56.8/78.0)	35.1 (24.3/63.6)	71.4 (69.7/73.2)	72.2 (68.3/76.6)

Table 2: Main results on the test sets of five NER benchmarks. F1 score (precision/recall) is reported.

4.2 NER Models

We trained three types of NER models on our synthetic data. We provide descriptions of the models below, but we cannot cover all the details; readers interested in details are therefore recommended to refer to Liang et al. (2020) and Meng et al. (2021). Note that we did not use validation sets to find the best model parameters during training to avoid excessive parameter tuning. The implementation details are provided in Appendix A.

Standard: This type of model consists of a pre-trained language model for encoding input sequences and a linear layer for token-level prediction. We used RoBERTa (Liu et al., 2019) as the language model for the news, Wikipedia, and Twitter domains and BioBERT (Lee et al., 2020) for the biomedical domain.

BOND (Liang et al., 2020): This model is based on self-training, which is a learning algorithm that corrects weak labels with the power of large-scale language models. Specifically, a *teacher* model (similar to the standard model above) is initially trained on the weakly-labeled corpus and used to re-annotate the corpus based on its predictions. This re-annotation process allows the model to remove noisy labels and further identify missing entities. A *student* model with the same model structure as the teacher model is trained on the re-annotated corpus. The teacher model is updated by the student model's parameters in the next round and performs the re-annotation process again. This process is repeated until the maximum training step is reached.

RoSTER (Meng et al., 2021): In RoSTER, the *generalized cross-entropy (GCE)* loss is used to a standard model, which is designed to be more robust to noise than a normal cross-entropy loss. During the GCE training, weak labels are removed at every update step if the model assigns low confidence scores. Using the algorithm described above, five randomly initialized models are trained, and a new model is trained to approximate the average predictions of the five models. Finally, the new model is further trained with *language model augmented self-training*, which jointly approximates the teacher model's predictions for the given (1) original sequence and (2) augmented sequence with some tokens replaced by a language model.

4.3 In-domain Resources

Baseline models are classified into two categories based on the amount of in-domain resources required during training.

GeNER (Kim et al., 2022): GeNER is the only baseline model that uses the same amount of resources as HighGEN. GeNER retrieves phrases and unlabeled sentences using natural language search and performs string matching to create datasets.

Full dictionary: Full-dictionary models use large-scale dictionaries that comprises numerous entities hand-labeled by experts. For the CoNLL-2003, Wikigold, and WNUT-16 datasets, each dictionary was constructed using Wikidata and dozens of gazetteers compiled from multiple websites (Liang et al., 2020). For NCBI-disease and BC5CDR, the dictionary was constructed by com-

bining the MeSH database and Comparative Toxicogenomics Database (more than 300k disease and chemical entities) (Shang et al., 2018). These dictionaries were used to generate weak labels based on string matches with *in-domain corpus*, which is an unlabeled version of the original training corpus.

4.4 Results

Table 2 shows that HighGEN outperformed GeNER on five datasets by average F1 scores of 4.2, 3.0, and 4.7 for the standard, BOND, and RoSTER models, respectively. Performance improvements were particularly evident in recall. When the verification method was not applied (i.e., w/o $\mathcal{L}$), the performance dropped by an average F1 score of 5.4 (mostly in precision). A high NER performance can be expected with full dictionaries, but they cannot be built without tremendous effort of experts. We emphasize that our method of automatically creating high-coverage pseudo-dictionaries and NER datasets is a promising way to achieve competitive performance with minimal effort.

5 Few-shot NER

We show an additional use case for HighGEN to create NER datasets using only a few hand-labeled examples, without using input questions. This can eliminate a tuning/engineering effort of users that might be required for designing appropriate questions to identify NER needs, which is a distinct advantage of HighGEN over GeNER. Specifically, HighGEN takes sentences with annotated phrases as input and retrieves $\hat{\mathbf{X}}_2$ and $\hat{\mathcal{V}}_2$ using the phrase embedding search (defined in Equations (1) and (2)), which are used as the unlabeled sentences and pseudo-dictionary to produce the final dataset.

We tested two types of models. (1) The entity-level model uses every annotated phrase as a separate query; thus, the number of queries equals the number of human annotations. On the other hand, (2) the class-level model first averages phrase vectors of the same entity types and uses them as queries; thus, the number of queries equals the number of entity types. The entity-level model would have an advantage in terms of entity recall and the class-level model can mitigate noise that each phrase vector may contain.

Setups. We sampled datasets from CoNLL-2003 and BC5CDR so that each dataset consists of five sentences per entity type, which results in 20 and 10

Model	CoNLL-2003	BC5CDR
5-shot sentences (per entity type)		
Supervised	53.5	55.0
+ NSP	61.4	-
+ Self-training	65.4	-
QUIP	74.0	65.7
HighGEN (entity)	**75.6**	68.2
HighGEN (class)	73.2	**72.5**

Table 3: F1 scores of HighGEN and baseline models in few-shot NER. Note that the scores of the baseline models are from previous studies (Huang et al., 2021; Jia et al., 2022; Kim et al., 2022).

examples for CoNLL-2003 and BC5CDR, respectively. [6] All experimental results were averaged over five sampled datasets. We used the models of Huang et al. (2021) and Jia et al. (2022) as baselines, and among them, QUIP (Jia et al., 2022) is the previous best model in few-shot NER (details on the models are presented in Appendix C).[7] For HighGEN, we retrieved the same number of sentences for each query, and the total number of sentences was 120k for CoNLL-2003 and and 10k for BC5CDR. We initially trained RoSTER on our synthetic data and then fine-tuned the model on few-shot examples.

Results. Table 3 shows that our entity- and class-level models outperformed QUIP by an average F1 score of 2.1 and 3.0 on the two datasets, respectively. For CoNLL-2003, the entity-level model was better than the class-level model because entities of the same entity type often belong to different sub-categories. For instance, "Volkswagen" and "University of Cambridge" belong to the same *organization* type in CoNLL-2003 but their subcategories are "company" and "institution," respectively. Therefore, it is difficult to group them into a single vector and it is important to widely cover various entities using separate queries for each subcategory. On the other hand, entities in BC5CDR can be naturally grouped by disease or chemical type, which allows the class-level model to perform well. Additionally, biomedical entity names often contain domain-specific terms, numbers, special characters, and abbreviations that are difficult to encode with a general-purpose phrase encoder,

---

[6]Unlike the experiments in Section 4, the *MISC* type was included for a fair comparison with baseline models.

[7]Other few-shot NER models were excluded because they used a sufficient amount of 'source' data (Yang and Katiyar, 2020; Cui et al., 2021), which is different from our setups.

Type	P@100			Div@10k		
	NL	PE	△	NL	PE	△
PER	98.3	99.0	0.7	37.1	81.4	44.3
LOC	97.7	99.0	1.3	2.6	2.9	0.3
ORG	95.7	95.0	−0.7	36.2	61.5	25.3
BIO	96.5	98.5	2.0	20.1	62.7	42.6
Average	97.1	97.9	0.8	24.0	52.1	28.1

Table 4: Retrieval results from natural language search (NL) and phrase embedding search (PE) for the four entity types: person (PER), location (LOC), organization (ORG), and biomedicine (BIO). △: performance difference between PE and NL.

making their vector representations relatively more error-prone. The class-model can produce good representations by averaging phrase vectors.

6 Analysis

6.1 Retrieval Performance

We compared natural language search and phrase embedding search in terms of their accuracy and diversity. With reference to Kim et al. (2022), we used 11 fine-grained questions within the following four coarse-grained entity types of (i) person (athlete, politician, actor), (ii) location (country, city, state in the USA), (iii) organization (sports team, company, institution), and (iv) biomedicine (disease, drug). We report the average scores for each coarse-grained entity type.

Metrics. (i) The precision at 100 (P@100) represents the accuracy of the top 100 retrieved phrases. Because there are no gold annotations for the retrieved phrases, we manually determined whether the phrases correspond to the correct entity types. (ii) Diversity at 10k (Div@10k) calculates the percentage of unique phrases out of the top 10k phrases based on their lowercase strings.

Results. The phrase embedding search largely outperformed the natural language search by a macro average of 28.1 diversity across the four types without loss of accuracy. The diversity scores for the location entity types did not improve significantly because there are only limited numbers of names for locations such as countries in the real world, but the diversity scores for the other types improved dramatically (+ 37.4 diversity).

While both queries produced accurate top results (P@100), the accuracy tends to decrease as the top-k increased, which makes it difficult to increase the
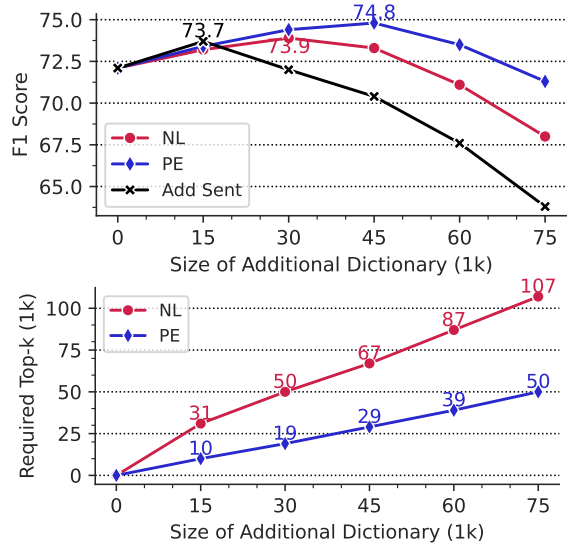

Figure 4: Performance of RoSTER models on BC5CDR with different sizes of the additional dictionary and the top-k to reach a certain dictionary size by the natural language search (NL) and phrase embedding search (PE). 'Add Sent' presents the performance of the model trained with additional sentences by the phrase embedding search (i.e., $\hat{X}_1 + \hat{X}_2$). The size of the initial dictionary ($x = 0$) is 12k.

dictionary size by retrieving more phrases. Thus, retrieving diverse entities with a reasonable top-k is not only important for computational efficiency but also helps the retriever to maintain accuracy. In this regard, phrase embedding search has a huge advantage over natural language search. We discuss this further in Section 6.2. In addition, examples of the top phrases retrieved by both search methods are listed in Table D.3 (Appendix).

6.2 Data Size

Effect of dictionary size. Figure 4 shows the NER performance of RoSTER models according to the size of the additional dictionary added to the initial dictionary $\hat{V}_1$. We expanded the dictionary using the natural language search or phrase embedding search. F1 scores were measured on the BC5CDR test set.

The performance of both models increased initially but decreased after the peaks, indicating that there was a trade-off between the size and accuracy of the dictionary. The optimal size of the additional dictionary by the phrase embedding search (i.e., 45k) was larger than that of the natural language search (i.e., 30k). As shown in the second graph in Figure 4, the natural language search required a much larger number of sentences (more

Sentence	Small $\mathcal{V}$ + String	Large $\mathcal{V}$ + String	Large $\mathcal{V}$ + Verif.
[1] . . . and on **Central** and Eastern Europeans living in the . . .	None (O)	Company (×)	None (O)
[2] . . . Foreign Minister **Alexander Downer** and various . . .	None (×)	Politician (O)	Politician (O)
[3] . . . at club level for Cruzeiro, PSV, **Barcelona**, and Inter Milan.	Sport team (O)	City (×)	Sport team (O)

Table 5: Case study of dictionary sizes and dictionary matching methods. Small $\mathcal{V}$: initial dictionary (i.e., $\hat{\mathcal{V}}_1$) consisting of 12k entities. Large $\mathcal{V}$: expanded dictionary (i.e., $\hat{\mathcal{V}}_1 + \hat{\mathcal{V}}_2$) consisting of 134k entities. String: rule-based string matching. Verif.: the verification method. ×: incorrect annotations. O: correct annotations.

than twice as much) than the phrase embedding search to obtain the required dictionary size, which caused more false-positive results to be included in the dictionary.

Effect of Additional Sentences. In addition to using the additional dictionary $\hat{\mathcal{V}}_2$ obtained using phrase embedding search, we tried to use additional sentences $\hat{\mathbf{X}}_2$ along with $\hat{\mathbf{X}}_1$ (see 'Add Sent' in Figure 4). The performance was higher than the other models at low top-k ($x = 15$k), but the performance degraded rapidly as the dictionary size grew. As discussed in Section 3.2, the sentences from the phrase embedding search have similar patterns, and from this result, we conjecture that the limited contextual patterns hindered the model's generalizability. In conclusion, using only $\hat{\mathbf{X}}_1$ for the unlabeled corpus and both $\hat{\mathcal{V}}_1$ and $\hat{\mathcal{V}}_2$ for the dictionary would result in the best NER performance in most cases. However, as shown in Section 5, using $\hat{\mathbf{X}}_2$ and $\hat{\mathcal{V}}_2$ can be a good alternative if users want to avoid effort required in query tuning.

6.3 Case Study

Table 5 shows several examples of how a large dictionary induced noise annotations in dictionary matching and how these annotations were corrected by the verification method. We used nine fine-grained entity types belonging to the person, location, and organization types, which were used in the experiments in Section 6.1. We denote the initial dictionary (i.e., $\hat{\mathcal{V}}_1$) as a small dictionary and the expanded dictionary that consists of the initial and additional dictionaries (i.e., $\hat{\mathcal{V}}_1 + \hat{\mathcal{V}}_2$) as a large dictionary. While the small dictionary could not match the entity "Alexander Downer" owing to its limited coverage, the entity was correctly annotated by a large dictionary. However, the large dictionary incorrectly annotated "Central" as a company, indicating that there is a trade-off between the coverage and accuracy of a dictionary. Also, "Barcelona" appeared mainly as a sports team in the small dictio-

nary, whereas in the large dictionary it frequently appeared as a city and was therefore incorrectly annotated by the latter. In contrast, our verification method had the advantages of both dictionaries; it preserved the high accuracy of the small dictionary while retaining the high coverage of the large dictionary, resulting in correct annotations.

7 Conclusion

In this study, we presented an advanced dataset generation framework, HighGEN, which combines (1) phrase embedding search to address the problem of efficiently retrieving various entities using an open-domain retriever and (2) verification method to deal with false positives in a large dictionary. In the experiments, we demonstrated the superiority of HighGEN using five NER benchmarks and performed extensive ablation studies, comparison of retrieval performance, and analysis of potential uses of the phrase embedding search in few-shot NER scenarios. We hope that our study will provide practical help in several data-poor domains and valuable insights into entity retrieval and weakly supervised NER.

Limitations

Inappropriate initial user questions can negatively affect NER performance. If they are not proper, the QA model returns incorrect phrases, and the phrase embedding queries generated from them will also be erroneous. The absence of a component for controlling this error cascade in our framework should be addressed in future studies.

In addition, our method is dependent on the phrase encoder of DensePhrases. Because the phrase encoder is a general-purpose model trained on Wikipedia-based datasets, its capability may be limited for domain-specific entities. In few-shot NER, the phrase encoder can be sensitive to the quality of given example sentences. Future studies should thoroughly analyze the effect of the

phrase encoder's performance on the resulting NER datasets and NER performance.

References

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. 2009. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18, Suntec, Singapore. Association for Computational Linguistics.

Yixin Cao, Zikun Hu, Tat-seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-resource name tagging learned with weakly labeled data. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 261–270, Hong Kong, China. Association for Computational Linguistics.

Anthony Chen, Pallavi Gudipati, Shayne Longpre, Xiao Ling, and Sameer Singh. 2021. Evaluating entity disambiguation and the role of popularity in retrieval-based NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4472–4485, Online. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Jason Fries, Sen Wu, Alex Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *ArXiv preprint*, abs/1704.06360.

Abbas Ghaddar and Phillippe Langlais. 2017. WiNER: A Wikipedia annotated corpus for named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 413–422, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2021. Few-shot named entity recognition: An empirical baseline study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10408–10423, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Robin Jia, Mike Lewis, and Luke Zettlemoyer. 2022. Question answering infused pre-training of general-purpose contextualized representations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 711–728, Dublin, Ireland. Association for Computational Linguistics.

Hyunjae Kim, Jaehyo Yoo, Seunghyun Yoon, Jinhyuk Lee, and Jaewoo Kang. 2022. Simple questions generate named entity recognition datasets. In *EMNLP*, Abu Dhabi, UAE. Association for Computational Linguistics.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634–6647, Online. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Megan Leszczynski, Daniel Fu, Mayee Chen, and Christopher Re. 2022. TABi: Type-aware bi-encoders for open-domain entity retrieval. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2147–2166, Dublin, Ireland. Association for Computational Linguistics.

Mike 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. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. BOND: bert-assisted open-domain named entity recognition with distant supervision. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1054–1064. ACM.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv preprint*, abs/1907.11692.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10367–10378, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2019. Distantly supervised named entity recognition using positive-unlabeled learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2409–2419, Florence, Italy. Association for Computational Linguistics.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access.

Esteban Safranchik, Shiying Luo, and Stephen H. Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 5570–5578. AAAI Press.

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2054–2064, Brussels, Belgium. Association for Computational Linguistics.

Benjamin Strauss, Bethany Toma, Alan Ritter, Marie-Catherine de Marneffe, and Wei Xu. 2016. Results of the WNUT16 named entity recognition shared task. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 138–144, Osaka, Japan. The COLING 2016 Organizing Committee.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10684–10695. IEEE.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6365–6375, Online. Association for Computational Linguistics.

Xinyan Zhao, Haibo Ding, and Zhe Feng. 2021. GLaRA: Graph-based labeling rule augmentation for weakly supervised named entity recognition. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3636–3649, Online. Association for Computational Linguistics.

A Implementation Details

Input questions. We used the same sets of input questions and the same number of sentences for each question as those used in the previous study (Kim et al., 2022), which are listed in Table A.1. It should be noted that (1) multiple questions for a single entity type were used because entity types in benchmark datasets are often defined in a coarse-grained way (i.e., they include several sub-categories), and using specific and concrete questions for each sub-category is more effective in covering entities in the benchmark as a whole. For instance, using three questions, "Which sports team?", "Which company?", and "Which institution?", is better for covering the *organization* type than a single question "Which organization?". In addition, (2) different questions were used for different benchmarks, even though the entity types had the same category names, because the sub-categories were different due to domain and corpus differences between the benchmarks.

Computational environment. We ran High-GEN and trained all NER models on Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz and a single 24GB GPU (GeForce RTX 3090). When retrieving a huge amount of phrases (e.g., k_l is greater than 100k), we disabled the "cuda" option and run the model on the CPU.

Implementation. We used the official codes provided by previous studies for the implementation of BOND,[8] RoSTER,[9] and GeNER.[10] We used GeNER's repository for the standard models. We did not implement the few-shot models but used the scores provided by Huang et al. (2021), Jia et al. (2022), and Kim et al. (2022). We implemented our phrase embedding search and HighGEN by modifying the code base of GeNER. We will release our code after the paper is accepted.

Hyperparameters.

- **Standard**: Standard models are vulnerable to over-fitting when trained on synthetic data by GeNER or HighGEN. Therefore, we trained RoBERTa and BioBERT-based models for only one epoch with a batch size of 32 and a learning rate of 1e-5. When using full dictionaries, we trained models for ten epochs

---

[8] https://github.com/cliang1453/BOND
[9] https://github.com/yumeng5/RoSTER
[10] https://github.com/dmis-lab/GeNER

for CoNLL-2003 and the biomedical domain datasets, and 20 epochs for the other small datasets (Wikigold and WNUT-16).

- **BOND**: We initially trained the teacher model for one epoch and also self-trained the model for additional one epoch. For the other hyperparameters, we used the ones suggested by the authors.

- **RoSTER**: We referred to the official repository to select hyperparameters. We used the default hyperparameters suggested by the authors, except for noise training epochs and self-training epochs that were set to 1. In addition, when training models on biomedical domain datasets by HighGEN, we used a threshold value of 0.1 in the noisy label removal step.

B Dataset Statistics

Table B.2 lists the statistics of the five benchmark datasets.

C Few-shot Models

Supervised: A standard model (described in Section 4.2) is trained directly on few-shot examples using a token-level cross-entropy loss.

Noisy supervised pre-training (NSP) (Huang et al., 2021): The model is initially trained on a large-scale weakly-labeled corpus, called WiNER (Ghaddar and Langlais, 2017), which consists of Wikipedia documents with weak labels generated using the anchor links and coreference resolution. Subsequently, the model is fine-tuned on few-shot examples.

Self-training (Huang et al., 2021): This model is trained using a current semi-supervised learning method (Xie et al., 2020). Specifically, the model is initially trained using few-shot examples and fine-tuned by self-training on unlabeled training sentences. Note that the detailed algorithm can be different from the self-training methods used in BOND and RoSTER; therefore, please refer to the papers for details.

QUIP (Jia et al., 2022): QUIP was used as the state-of-the-art few-shot model in our experiment. The model is pre-trained with approximately 80 million question-answer pairs that are automatically generated by the BART-large model (Lewis

Dataset	Entity Types (Query Terms)	k_l	k_l'	$|\hat{\mathbf{X}}|$
CoNLL-2003	**person** (athlete, politician, actor) / **location** (country, city, state in the USA) / **organization** (sports team, company, institution)	5k	30k	45k
Wikigold	**person** (athlete, politician, actor, director, musician) / **location** (country, city, state in the USA, road, island) / **organization** (sports team, company, institution, association, band)	4k	30k	60k
WNUT-16	**person** (athlete, politician, actor, author) / **location** (country, city, state in the USA) / **product** (mobile app, software, operating system, car, smart phone) / **facility** (facility, cafe, restaurant, college, music venue, sports facility) / **company** (company, technology company, news agency, magazine) / **sports team** (sports team) / **TV show** (TV show) / **movie** (movie) / **music artist** (band, rapper, musician, singer)	1k	30k	29k
NCBI-disease	**disease** (disease)	35k	30k	35k
BC5CDR	**disease** (disease) / **chemical** (chemical compound, drug)	15k	30k	45k

Table A.1: Questions and hyperparameters used for NER benchmarks. Each question is formulated as "*Which* [TYPE]*?*" and used for the retrieval. k_l and k_l' indicate the number of the top phrases/sentences retrieved from the natural language search and the phrase embedding search for each question, respectively. $|\hat{\mathbf{X}}|$ represents the dataset size (i.e., number of training sentences), which is calculated by multiplying the number of questions by k_l.

Domain (Corpus)	Dataset (# Types)	Training		Validation		Test	
		# Sents	# Labels	# Sents	# Labels	# Sents	# Labels
News (Reuters)	CoNLL-2003 (3)	14,987	20,061	3,469	5,022	3,685	4,947
Wikipedia	Wikigold (3)	1,142	1,842	280	523	274	484
Twitter	WNUT-16 (9)	2,394	1,271	1,000	529	3,850	2,889
Biomedicine (PubMed)	NCBI-disease (1)	5,432	5,134	923	787	942	960
	BC5CDR (2)	4,582	9,387	4,602	9,596	4,812	9,809

Table B.2: Statistics of NER benchmark datasets. # Types: number of entity types. # Sents: number of sentences. # Labels: number of entity-level human annotations.

et al., 2020), enabling the model to generate high-quality phrase representations, and therefore, achieve strong performance in several few-shot downstream tasks such as NER and QA. After pretraining, the prediction layer of QUIP is initialized with the embeddings of *question prompts*, which has shown to be more effective in few-shot experiments than random initialization. For instance, '*Who is a person?*'' was used for the *person* type and "*What is a location?*" was used for the *location* type. We used the same question prompts as those used in the study of Jia et al. (2022) for CoNLL-2003, and those used in the study of Kim et al. (2022) for BC5CDR.

D Retrieved Entities

Table D.3 shows the top 20 phrases retrieved by the natural language search and phrase embedding search for the four entity types of politician, company, disease, and drug. The phrases from both

search methods are generally accurate except for some noisy ones, but the phrase embedding search outperformed the natural language search in terms of the diversity of the retrieved phrases.

Natural Language Search			
Politician	**Company**	**Disease**	**Drug**
Ed Miliband	Foxconn	Leprosy	morphine
David Cameron	Boeing	cirrhosis	opium
David Cameron	Plessey	leprosy	alcohol
David Cameron	Marconi	polio	heroin
David Cameron	Sony Corporation	leprosy	morphine
Nick Clegg	Packard Bell	syphilis	chlorpromazine
David Cameron	Airbus	typhus –	Copaxone
David Cameron	Olympus	Cholera	aspirin
David Cameron	Airbus	syphilis	heroin
Douglas Hurd	Airbus	tuberculosis	Vioxx
Ted Heath	Nokia	typhus	heroin
David Cameron	Paramount	leprosy	imipramine
David Cameron	Seagate	tuberculosis	cocaine
Gordon Brown	Cisco	Leprosy	Thalidomide
Gordon Brown	Cisco	Leprosy	LSD
Margaret Thatcher	News Corporation	syphilis	cocaine
Jeremy Corbyn	Nokia	leprosy	Cisplatin
Harold Wilson	Mattel	polio	penicillin
David Cameron	Seagate	typhus	cannabis
David Cameron	Airbus Group	Measles	Opioids
Phrase Embedding Search			
Politician	**Company**	**Disease**	**Drug**
David Anthony Laws	Unicer Unicer	Leprosy	Adrafinil
Stefan Löfven	Boeing	Leptospirosis	Nitrous oxide
Michael Ignatieff	Diesel	hereditary rheumatic syndrome	ivermectin
Tony Benn	Arctic	chronic fatigue syndrome	Pentothal
John Major	Monster	Mal de Débarquement syndrome	Camptothecin
Sir Oswald Mosley	Samsung	seasickness	Glybera
George Galloway	Gateway 2000	Guillain Barre Syndrome	Trimecaine
Arthur Gordon Lishman	Airbus	Leptospirosis	Gerovital H3
William Hague	Fiat	Smallpox	Elaterin
Sarah Louise Teather	Fiat	Crohn's disease	Prozac
Robert Owen Biggs Wilson	American DeForest	Achromatopsia	Methamphetamine
Helle Thorning-Schmidt	TNT	Leprosy	metronidazole
Philip Andrew Davies	Tenneco Automotive	Haff disease	Desvenlafaxine
Vince Gair	AgustaWestland	rhabdomyolysis	4-Fluoroamphetamine
Paul William Barry Marsden	Anshe Chung Studios	Möbius syndrome	ephedra
Jeremy William Bray	Raytheon Systems Ltd	Hansen's Disease	ephedrine
Michael Howard	Microsoft	Lady Windermere syndrome	Alseroxylon
Bruce Hawker	Airbus	McCune–Albright syndrome	Benzydamine
Andrew David Smith	Diesel	Grover's disease	Diclofenamide
Peter David Shore	Microsoft	Lipodermatosclerosis	Cefdinir

Table D.3: Top 20 phrases retrieved by the natural language search and phrase embedding search for the four entity types: politician, company, disease, and drug.

A   For every submission:

☑ A1. Did you describe the limitations of your work?
the Limitations section

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
abstract and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B   ☑ Did you use or create scientific artifacts?

Seciton 2.2 - DensePhrases, Section 4.1 - Benchmark datasets, Section 4.2 - NER models, Section 4.3 - in-domain dictionaries and GeNER, and the Appendix for all details.

☑ B1. Did you cite the creators of artifacts you used?
Seciton 2.2 - DensePhrases, Section 4.1 - Benchmark datasets, Section 4.2 - NER models, Section 4.3 - in-domain dictionaries and GeNER, and the Appendix for all details.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. All artifacts are freely available for research purposes.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. All artifacts used with the intended use. Our code and synthetic datasets will be available online upon acceptance, and they can be used for both research and industrial purposes.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Seciton 2.2 - DensePhrases, Section 4.1 - Benchmark datasets, Section 4.2 - NER models, Section 4.3 - in-domain dictionaries and GeNER, and the Appendix for all details.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4.1 - Benchmark datasets, Section 4.3 - in-domain dictionaries, and the Appendix for all details.

C ☑ **Did you run computational experiments?**

Sections 4 and 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Sections 4 and 5, and the Appendix

☒ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
We will add statistics information in the final version.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix

D ☒ **Did you use human annotators (e.g., crowdworkers) or research with human participants?**

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.