Automatic Context Pattern Generation for Entity Set Expansion

Yinghui Li, Shulin Huang, Xinwei Zhang, Qingyu Zhou, Yangning Li, Ruiyang Liu, Yunbo Cao, Hai-Tao Zheng *Member, IEEE*, and Ying Shen *Senior Member, IEEE*

Abstract—Entity Set Expansion (ESE) is a valuable task that aims to find entities of the target semantic class described by given seed entities. Various Natural Language Processing (NLP) and Information Retrieval (IR) downstream applications have benefited from ESE due to its ability to discover knowledge. Although existing corpus-based ESE methods have achieved great progress, they still rely on corpora with high-quality entity information annotated, because most of them need to obtain the context patterns through the position of the entity in a sentence. Therefore, the quality of the given corpora and their entity annotation has become the bottleneck that limits the performance of such methods. To overcome this dilemma and make the ESE models free from the dependence on entity annotation, our work aims to explore a new ESE paradigm, namely corpus-independent ESE. Specifically, we devise a context pattern generation module that utilizes autoregressive language models (e.g., GPT-2) to automatically generate high-quality context patterns for entities. In addition, we propose the GAPA, a novel ESE framework that leverages the aforementioned **G**ener**A**ted **PA**tterns to expand target entities. Extensive experiments and detailed analyses on three widely used datasets demonstrate the effectiveness of our method. All the codes of our experiments are available at https://github.com/geekjuruo/GAPA.

Index Terms—Artificial Intelligence, Information Search and Retrieval, Natural Language Processing, Language Models.

1 Introduction

Entity Set Expansion (ESE) - i.e., aiming to expand complete entities that belong to the same semantic class as a few seed entities - is a promising task in IR and NLP [1], [2], [3]. For example, given the input seed entity set {*"China"*, *"Japan"*, *"Canada"*}, an ESE model is expected to output more new entities (e.g., *"Germany"*, *"Australia"*, *"Singapore"*, ...) that all belong to the same `Country` class as the seed entities. Thanks to its ability to automatically discover knowledge and mine semantics, the ESE task can benefit kinds of IR and NLP downstream applications, such as Web Search [4], Knowledge Graph Completion [5], and Question Answering [6].

In recent years, various corpus-based bootstrapping methods have been studied and became mainstream solutions for ESE [7], [8], [9], [10]. Given a specific corpus w ith entities annotated, these methods mainly bootstrap the seed entity set by iteratively selecting context patterns and ranking expanded entities [11], [12], [13], [14]. Taking Figure 1 as an example, when expanding the seed entity set {*"Nevada"*, *"Ohio"*, *"Texas"*} which belongs to the `US States` class, corpus-based ESE methods will extract related context patterns from the given annotated corpus, such as "County, _, USA.". Then iterative expansion is performed according to similarity between the context and entity representations. However, the biggest constraint faced by these corpus-based methods is that their performance relies heavily on the entity annotation quality of the given corpus. When there is no given corpus or the entity annotation of the given corpus is very noisy (which is very common and normal in real-world application scenarios), the entity expansion performance of these corpus-based ESE methods will undoubtedly be greatly reduced, because it will be difficult for them to extract high-quality context patterns. Therefore, how to make the ESE methods get rid of the dependence on the entity annotation of the corpus is a problem worthy of studying in ESE. To address this problem, our work brings the ESE task into a new paradigm, namely corpus-independent ESE which does not need to be given a specific corpus with entity annotations.

In this study, we propose to empower ESE with context patterns automatically generated from the autoregressive language models, such as GPT-2. Intuitively, given an entity, the language generation model can utilize this entity as guiding text to automatically generate the corresponding context (i.e., prev-text and next-text of entities). The context automatically generated by pre-trained language model can naturally be regarded as a context pattern that the ESE model can make use of. Note that most of the widely used language generation models are well pre-trained without any artificially designed features or entity annotation, so that the context patterns they generate are without the help of human involvement and corpora annotation. Furthermore,

- *Yinghui Li, Shulin Huang, Yangning Li, and Hai-Tao Zheng are with Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong 518055, China, and also with Peng Cheng Laboratory, Shenzhen, Guangdong 518066, China. E-mail: {liyinghu20, sl-huang21, liyn20} @mails.tsinghua.edu.cn, zheng.haitao@sz.tsinghua.edu.cn.*
- *Xinwei Zhang and Ying Shen are with the School of Intelligent Systems Engineering, Sun Yat-Sen University, Guangzhou, Guangdong 510275, China. E-mail: zhangxw57@mail2.sysu.edu.cn, sheny76@mail.sysu.edu.cn.*
- *Qingyu Zhou and Yunbo Cao are with the Tencent Cloud Xiaowei, Beijing 100190, China. E-mail: qyzhgm@gmail.com, yunbocao@tencent.com.*
- *Ruiyang Liu is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100190, China. E-mail: lry20@mails.tsinghua.edu.cn.*

Manuscript received 17 July 2022; revised 8 Mar 2023.
Yinghui Li and Shulin Huang contribute equally for this work.
Corresponding authors: Hai-Tao Zheng and Ying Shen.

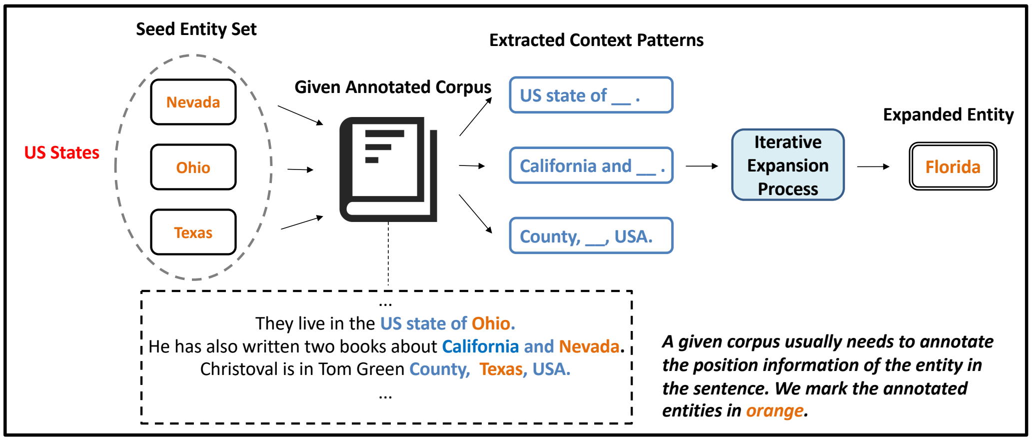


Fig. 1. An example showing the expansion process of the traditional corpus-based ESE methods.

after obtaining automatically generated context patterns, we can utilize them to obtain entity's context representations for measuring entity similarity, based on the important and reasonable assumption that "similar entities should have similar context" [15], [16].

Motivated by the above intuition, we propose a novel iterative ESE framework that consists of three modules: (1) The first, *supervision signal enhancement* module, updates initial seed/candidate entity sets according to the entity similarity which is calculated from the entity representations. (2) The second, *context pattern generation* module, utilizes two separate GPT-2 models in opposite directions to automatically generate amounts of context patterns of entities (e.g., "Bill Gates is [MASK]'s founder." is one context pattern of the entity "Microsoft"). (3) The third, *generated patterns guided expansion* process, scores each entity in the candidate set according to its context similarity with seed entities and adds top-ranked entities into the seed set iteratively. As our proposed method automatically generates context patterns for input entities, our framework does not require any entity annotation information and ESE task-related corpora, so we declare it is corpus-independent for the ESE task.

In summary, the major contributions of this paper are:

1) We first apply reverse GPT-2 to generate the prev-text of entities and automatically derive high-quality context patterns for the ESE task, thus freeing our proposed model from the reliance on the corpora with annotated entities.
2) We propose a novel ESE framework, GAPA, which can utilize the automatically **G**ener**A**ted **PA**tterns to iteratively expand target entities effectively.
3) Extensive experiments and detailed analyses show that GAPA outperforms previous state-of-the-art methods without any manually designed patterns.

The rest of the article is organized as follows. Section 2 provides a comprehensive review of related work on entity set expansion and autoregressive language models. Section 3 describes the technology details of our proposed GAPA. Experimental settings and analyses are presented in Section 4. Finally, we conclude our work and give an outlook on its future directions in Section 5.

2 Related Work

2.1 Entity Set Expansion

Entity Set Expansion (ESE) task has attracted extensive research efforts due to its practical importance and broad application prospects [17], [18], [19]. Benefitting from its ability to mine semantics and discover knowledge, the ESE task can be used for many downstream tasks or applications, such as taxonomy construction [11], Query Optimization [17] and News Recommendation [7]. Early works in ESE mainly focus on web-based methods, such as SEAL [20] and Google Sets [21], which require search engines and web pages as external online resources for expansion. Due to the low efficiency of web-based methods, more recent studies have paid more attention to corpus-based methods [22], [23], [24]. Among various corpus-based methods, the iterative bootstrapping methods are the most mainstream solutions [9], [10], [25]. These bootstrapping methods aim to bootstrap the seed entity by iteratively selecting context patterns and ranking expanded entities. To accurately extract context patterns, these methods require that the given corpus must contain annotations of where entities occur in sentences. For example, the previous state-of-the-art ESE methods CGExpan [24] and ProbExpan [26] both need annotated corpus in their methods. CGExpan needs to have the entity position of a sentence, so as to achieve the contextual representation of the entity by BERT, and ProbExpan needs to mask the entity in the sentence to pre-train the entity-level masked language model. In addition, there is a long-term problem of *sparse supervision* in the ESE task, because it tends to have very few seed entities as input [8]. SynSetExpan [27] proposes to leverage the task of synonym discovery to give ESE model more supervision signal, thereby improving the performance of the ESE model. Different from SynSetExpan, we propose to directly use the entity representations to measure the similarity between entities, and then update the initial seed/candidate sets, so as to achieve the purpose of supervision signal enhancement.

2.2 Autoregressive Language Models

Recently, various pre-trained language models have been applied in various natural language processing tasks and gained good performance [28], [29], [30]. The mainstream pre-trained language models can be divided into two categories, namely autoencoder language models and autoregressive language models. Common autoencoder language models include BERT [31], T5 [32], RoBERTa [33], UniLM [34] and XLM [35]. The typical representative of them, BERT, cannot directly deal with text generation tasks due to its masking pre-training strategy and Transformer-Encoder structure [36]. Therefore, most autoregressive language models are naturally suitable for various natural language generation tasks, such as GPT-1 [37], GPT-2 [38] and ULMFiT [39], because of their unbiased estimation of joint probability and consideration of the correlation between predicted tokens. But in this autoregressive paradigm, language models can only generate the next-text based on the prev-text, that is, generate text in the direction from left to right. The reason for this limitation is that the joint probability derived by the autoregressive language models is decomposed according to the normal text sequence order. To the best of our knowledge, our work is the first to apply autoregressive language models to generate prev-text from right to left, and verify its feasibility and effectiveness empirically.

3 METHODOLOGY

In this section, to describe the details of our method more clearly, we first formulate the ESE task and define some key concepts of ESE (Section 3.1). Specially, we will firstly introduce the mechanism of the supervision signal enhancement module (Section 3.2) that we propose for the sparse supervision problem in ESE. Then we will illustrate how we use autoregressive language models to automatically generate the context patterns of entities (Section 3.3). Finally, we will describe the generated patterns guided expansion process (Section 3.4). The overview of our proposed method is shown in Figure 2 and we will summarize our proposed method in Section 3.5. Besides, we will provide a detailed example explanation for the Figure 2 in Section 3.5.

3.1 Problem Formulation

Entity. An entity is something that is distinguishable and exists independently in real world, such as *"China"* and *"United States"*.

Context Pattern. An entity's context pattern contains a **prev-text** and a **next-text**. For example, for the sentence *"We all know that Beijing is the capital of China."* which contains the entity *"Beijing"*, *"We all know that *"* is the prev-text and *"* is the capital of China."* is the next-text of *"Beijing"*.

Semantic Class. A semantic class can classify entities semantically. For instance, *"Japan"* belongs to the `Country` class and *"Microsoft"* belongs to the `Company` class.

Seed Entity Set. A small set of few entities belonging to the same semantic class. In our work, the size of the initial seed entity set is typically 3.

Candidate Entity Set. For a specific ESE dataset, the candidate entity set is a large set containing all entities of all semantic classes in this dataset.

ESE Task. The inputs are a complete set of candidate entities and a small set of seed entities (e.g., {*"United States"*, *"Japan"*, *"Canada"*}) belonging to the same semantic class (i.e., `Country`), the ideal output is a ranked list of other `Country` entities from the candidate set, such as *"Korea"*, *"India"*, and *"Singapore"*.

3.2 Supervision Signal Enhancement

Because the number of initial seed entities is very small (usually only 3-5 entities are initialized as the seed set), while the number of candidate entities is often hundreds or thousands, this naturally poses the *sparse supervision* challenge. Intuitively, if there are more seed entities and fewer candidate entities, the supervision signal will be strengthened. Based on this intuition, we propose the supervision signal enhancement module to automatically increase seed entities and reduce candidate entities.

Given a small initial seed entity set $\mathbb{E}_{\text{seed}}$ and a large candidate entity set $\mathbb{E}_{\text{cand}}$, whether an entity should be added to the seed set or removed from the candidate set is judged according to its similarity to the initial seed entity set. Specifically, the entity similarity score$_e$ between a candidate entity $e_c \in \mathbb{E}_{\text{cand}}$ and $\mathbb{E}_{\text{seed}}$ is measured by computing the cosine distance between entity representations as follow:

$$\text{score}_e(e_c) = \frac{1}{|\mathbb{E}_{\text{seed}}|} \sum_{e_s \in \mathbb{E}_{\text{seed}}} \cos\left(\mathbf{r}_e(e_c), \mathbf{r}_e(e_s)\right), \quad (1)$$

where $|\mathbb{E}_{\text{seed}}|$ is the size of $\mathbb{E}_{\text{seed}}$ and $\cos(x, y) = \frac{x \cdot y}{||x||_2 ||y||_2}$ is cosine distance between two entity representations. The operator $\mathbf{r}_e(e)$ represents the entity representation, which is defined as:

$$\mathbf{r}_e(e) = \text{Glove}(e), \quad (2)$$

where Glove means the pre-trained word vectors. For an entity composed of multiple words, if their concatenation can be directly found in Glove's vocabulary, their concatenated word's vector is used. Otherwise, we average the word vectors of each word as the representation of this entity.

So far, we can judge the similarity between the candidate entity and the seed entity set according to the similarity score (i.e., Equation 1). If the candidate entity has a high similarity score with the seed entity set, it should be added to the seed set to increase the number of seed entities. It is worth noting that under the setting of ESE, it is necessary to keep the seed set and the candidate set without overlapping entities, so we need to remove these entities with high similarity from the candidate set at the same time. Conversely, if the candidate entity has a low similarity score to the seed entity set, it should be removed from the candidate entity set. Practically, we set upper and lower thresholds, i.e., thr$_u$ and thr$_l$, to distinguish different entities. Lager seed entity set $\mathbb{E}^{'}_{\text{seed}}$ and smaller candidate entity set $\mathbb{E}^{'}_{\text{cand}}$ are achieved as:

$$\mathbb{E}^{'}_{\text{seed}} = \mathbb{E}_{\text{seed}} \cup \{e | e \in \mathbb{E}_{\text{cand}},\ \ \text{score}_e(e) \geq \text{thr}_u\}, \quad (3)$$

$$\mathbb{E}^{'}_{\text{cand}} = \{e | e \in \mathbb{E}_{\text{cand}},\ \ \text{thr}_u > \text{score}_e(e) \geq \text{thr}_l\}. \quad (4)$$

By adding more similar entities in the seed set and reducing dissimilar entities from the candidate set, we finally effectively alleviate the effect of sparse supervision in ESE. In addition, a stronger set of seed entities is also beneficial to improve the quality of the generated context patterns later.

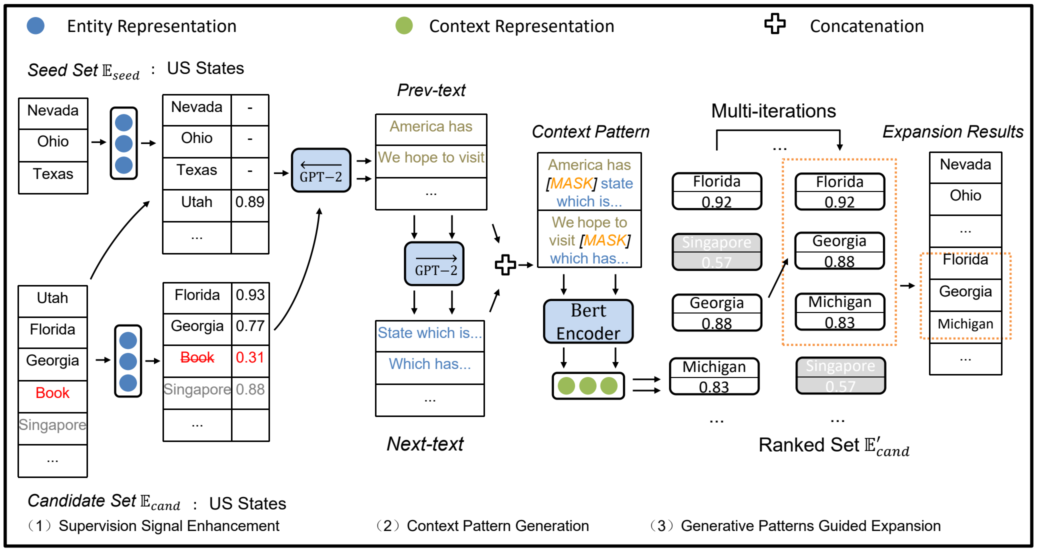


Fig. 2. Overview of our proposed GAPA framework. We update the initial seed/candidate sets according to the similarity of entity representations. Automatically generate the prev-text and next-text of entities respectively through two opposite GPT-2 models, and then concatenate them to get the context patterns. According to the similarity of context representations obtained by BERT, we iteratively select proper entities from the candidate set to add into the seed set, thus resulting in ideal target expansion results.

3.3 Context Pattern Generation

The context pattern generation module inputs seed/candidate sets and generates context patterns for entities. We construct this module by utilizing two separate GPT-2 models in opposite directions to generate the prev-text and next-text of entities.

As mentioned in Section 2, traditional autoregressive language models, such as GPT-2, can only generate the next-text from left to right. Therefore, to efficiently obtain the prev-next of entities, we pre-train a reverse GPT-2 model in a simple training way. Specifically, we pre-process the regular corpus to reverse all the sentences and then use these reverse corpora to pre-train the GPT-2 model. Apart from that, the training strategy and loss function design for training reverse GPT-2 are not different from the training process of regular GPT-2. In other words, we pre-train GPT-2 with the reverse corpus to give it the ability to generate prev-text from right to left. For the convenience, we denote the reverse GPT-2 as $\overleftarrow{\text{GPT-2}}$ and the regular GPT-2 as $\overrightarrow{\text{GPT-2}}$ in our paper.

First, for an entity e, we use e as the guiding text and $\overleftarrow{\text{GPT-2}}$ to generate the reverse text $\overleftarrow{\text{prev}}$:

$$\overleftarrow{\text{prev}} = \overleftarrow{\text{GPT-2}}(e). \tag{5}$$

Note that the word order of $\overleftarrow{\text{prev}}$ is reversed at this time, we can very simply reverse it to $\overrightarrow{\text{prev}}$. Then we use $\overrightarrow{\text{prev}}$ and e as the guiding text for $\overrightarrow{\text{GPT-2}}$ to generate the next-text $\overrightarrow{\text{next}}$:

$$\overrightarrow{\text{next}} = \overrightarrow{\text{GPT-2}}([\overrightarrow{\text{prev}}; e]), \tag{6}$$

where $[\cdot;\cdot]$ denotes concatenation. Finally, combining the prev-text and next-text of e, we get one context pattern of e :

$$\mathbf{c}(e) = [\overrightarrow{\text{prev}}; [\text{MASK}]; \overrightarrow{\text{next}}], \tag{7}$$

where [MASK] is the masked token of BERT [31] which will be used to get the context representation of entities through BERT. In particular, in our framework, a [MASK] represents an entire entity regardless of how many words the entity consists of. The reason we select BERT as our encoder is that BERT is widely used as the encoder in previous advanced ESE work and other NLP tasks.

An important problem worth considering in the above procedure is the choice of GPT-2 decoding strategy. In text generation tasks, GPT-2 often uses decoding strategies such as Greedy Search [40], Beam Search [41], and Top-K Sampling [42]. The typical characteristic of Greedy Search and Beam Search is that when the pre-training parameters of GPT-2 and guiding text are fixed, the generated text of each time is not very different. Therefore, in order to ensure the diversity of the context patterns we generate for every entity, we choose Top-K Sampling as the decoding strategy.

3.4 Generated Patterns Guided Expansion

For every entity in $\mathbb{E}^{'}_{\text{seed}}$ and $\mathbb{E}^{'}_{\text{cand}}$, we automatically generate m context patterns through the context pattern generation module. Supported by the theoretical assumption that "similar entities should have similar contexts", whether a candidate entity should be expanded is judged according to its context similarity to the seed set. Specifically, we measure the context similarity score_c between a candidate entity $e_c \in \mathbb{E}^{'}_{\text{cand}}$ and $\mathbb{E}^{'}_{\text{seed}}$ by computing the cosine distance between context representations as follow:

$$\text{score}_c(e_c) = \frac{1}{|\mathbb{E}^{'}_{\text{seed}}|} \sum_{e_s \in \mathbb{E}^{'}_{\text{seed}}} \cos\left(\mathbf{r}_c(e_c), \mathbf{r}_c(e_s)\right), \tag{8}$$

where $\mathbf{r}_c(e)$ represents the average of the entity's m context representations derived from BERT, which is defined as:

$$\mathbf{r}_c(e) = \frac{1}{m} \sum_{i=0}^{m-1} \text{BERT}(\mathbf{c}_i(e)), \tag{9}$$

where $\mathbf{c}_i(e)$ means i-th context pattern of entity e, which is defined in Equation 7.

After obtaining the context similarity score between each entity in $\mathbb{E}^{'}_{\text{cand}}$ and $\mathbb{E}^{'}_{\text{seed}}$, our model selects the candidate entities with the top-3 scores and add them to $\mathbb{E}^{'}_{\text{seed}}$ at each iteration. Such iterations are continued until the number of entities in $\mathbb{E}^{'}_{\text{seed}}$ reaches the target expansion number. Through this iterative generated patterns guided expansion, we will get the final ideal target entities.

3.5 Summary of Methodology

Starting with a small seed entity set and a large candidate entity set, we first use the Glove as the entity representations, and update the initial seed/candidate sets according to the similarity of entity representations, which alleviates the sparse supervision problem in ESE. Secondly, we automatically generate context patterns for entities using two opposite GPT-2 models, thus making our method less dependent on annotated corpora. Then, we utilize BERT to encode the generated context patterns to obtain context representations. Finally, we iteratively rank the candidate set according to the context similarity score and select top-ranked entities into the seed set, thus achieving the purpose of the ESE task.

To clarify our proposed method more concretely, we now give an example which is illustrated in Figure 2. Given the seed set {*"Nevada"*, *"Ohio"*, *"Texas"*} and a large candidate set, we expect our model to expand more `US States` entities into the initial set. Firstly, GAPA calculates the similarity between candidate entities and seed entities according to the entity representations. Based on the entity similarity score, GAPA updates the initial set and candidate set, such as adding *"Utah"* with higher score to the seed set and removing *"Microsoft"* with lower score from the candidate set. Secondly, we use $\overleftarrow{\text{GPT-2}}$ and $\overrightarrow{\text{GPT-2}}$ to generate various prev-text and next-text for entities. And we concatenate the prev-text and next-text to form the context patterns, such as *"America has [MASK] state which is ..."*. Thirdly, we utilize BERT to encode the context patterns and get the context representations of entities. Furthermore, GAPA measures the similarity of entities by computing the similarity of context, so as to iteratively add more similar entities to the seed set. Taking an iteration shown in the Figure 2 as an example, we select *"Florida"*, *"Georgia"*, *"Michigan"* from the candidate set and add them into the seed set. This iteration continues until the size of the seed set expands to the target number.

4 Experiments

4.1 Datasets

To ensure fairness, in addition to two widely used datasets (i.e., Wiki/APR) in previous works [24], [27], [43], we also choose another larger and more challenging SE2 dataset [27]. Following previous works which used Wiki/APR datasets, in our experiments, each semantic class has 5 seed sets and each seed set has 3 entity queries. The details of our used datasets are as follows:

1) **Wiki** [24], which is from a subset of English Wikipedia. It contains 8 semantic classes, including `China Provinces`, `Companies`, `Countries`, `Diseases`, `Parties`, `Sports Leagues`, `TV Channels` and `US States`.
2) **APR** [24], which consists of all the 2015 year's news articles published by Associated Press and Reuters. It contains 3 semantic classes, including `Countries`, `Parties` and `US States`.
3) **SE2** [27], the largest ESE benchmark to date, which has 60 semantic classes and 1200 seed entity queries.

Additionally, we use the Wikipedia 20171201 dump[1] as the pre-training corpus (we denote this part of data as Reverse-Wikipedia in our paper) for the reverse GPT-2 model (i.e., $\overleftarrow{\text{GPT-2}}$) in the context pattern generation module. It is worth noting that the Reverse-Wikipedia is the only training data we use, which itself source is a widely used unsupervised corpus for pre-training, does not contain any entity annotations related to ESE tasks. Furthermore, we use this part of data in reverse to train $\overleftarrow{\text{GPT-2}}$. Therefore, the existence of the Reverse-Wikipedia does not affect our claim that our GAPA is a corpus-independent ESE method.

4.2 Compared Methods

We compare our method with the following ESE methods.

1) Egoset [12]: This method uses context features and word embeddings to generate a sparse word ego network centered on seed entity. The network is used to capture words with similar semantics.
2) SetExpan [43]: This method calculates the distribution similarity to select context features from the corpus iteratively. Then SetExpan expands the entity set based on its proposed rank ensemble mechanism.
3) SetExpander [13]: This method utilizes a specific classifier to predict whether the candidate entity belongs to the seed set. The input of the classifier is the context features.
4) CaSE [23]: This method combines Skip-Gram context feature and embedding feature to sort all candidate entities in the corpus and then select entities to expand the entity set.
5) Set-CoExpan [14]: This method automatically generates auxiliary sets which contain negative entities to explicitly alleviate the semantic drift problem.
6) CGExpan [24]: This method uses BERT to generate the class name, with the help of Hearst patterns [44]. With the constraint of positive class name, the new entity and seed entity are in the same category, avoiding semantic drift problem.
7) SynSetExpan [27]: This method proposes to better leverages the limited supervision signals in seeds by utilizing the synonym information.
8) ProbExpan [26]: Current state-of-the-art method on Wiki/APR/SE2 datasets. It designs an entity-level masked language model and conducts contrastive learning on the corpus with entity annotations.

4.3 Evaluation Metric

The task objective of ESE is to expand a ranked list of entities that belong to the same semantic class as seed entities.

1. The raw corpus we use for pre-training the reverse GPT-2 can be directly downloaded from https://archive.org/details/enwiki-20171201.

TABLE 1
MAP@K(10/20/50) of different methods. All baseline results are directly from other published papers. We underline the previous state-of-the-art performance on three datasets for convenient comparison.

Methods	Wiki			APR			SE2		
	MAP@10	MAP@20	MAP@50	MAP@ 10	MAP@20	MAP@50	MAP@10	MAP@20	MAP@50
Egoset [12]	0.904	0.877	0.745	0.758	0.710	0.570	0.583	0.533	0.433
SetExpan [43]	0.944	0.921	0.720	0.789	0.763	0.639	0.473	0.418	0.341
SetExpander [13]	0.499	0.439	0.321	0.287	0.208	0.120	0.520	0.475	0.397
CaSE [23]	0.897	0.806	0.588	0.619	0.494	0.330	0.534	0.497	0.420
Set-CoExpan [14]	0.976	0.964	0.905	0.933	0.915	0.830	-	-	-
CGExpan [24]	<u>0.995</u>	0.978	0.902	0.992	<u>0.990</u>	0.955	0.601	0.543	0.438
SynSetExpan [27]	0.991	0.978	0.904	0.985	<u>0.990</u>	<u>0.960</u>	0.628	0.584	0.502
ProbExpan [26]	<u>0.995</u>	<u>0.982</u>	<u>0.926</u>	<u>0.993</u>	<u>0.990</u>	0.934	<u>0.683</u>	<u>0.633</u>	<u>0.541</u>
GAPA (BERT-base)	**1.000**	**1.000**	0.971	**1.000**	**1.000**	0.985	**0.688**	0.640	0.540
GAPA (BERT-wwm)	**1.000**	**1.000**	**0.974**	**1.000**	**1.000**	**0.990**	**0.688**	**0.641**	**0.542**

TABLE 2
The performance (MAP@50) comparison between CGExpan and GAPA under different classes.

Semantic Class	CGExpan	GAPA
`US States`	0.950	0.947 ↓
`China Provinces`	0.662	0.868 ↑
`Countries`	0.965	1.000 ↑
`Diseases`	1.000	1.000 –
`Sports Leagues`	0.942	1.000 ↑
`TV Channels`	0.925	1.000 ↑
`Parties`	0.937	1.000 ↑
`Companies`	0.867	0.976 ↑

Therefore, previous works widely use the Mean Average Precision at different top K positions (i.e., MAP@K) as the evaluation metric. Specifically, MAP@K is calculated as:

$$\text{MAP@K} = \frac{1}{|Q|}\sum_{q\in Q}\text{AP}_K(R_q, G_q), \tag{10}$$

where Q is all the seed entity sets. And for each set q, the $\text{AP}_K(R_q, G_q)$ denotes the average precision at position K with the ranked list R_q and ground-truth list G_q. To ensure the fairness of the experiments, we set the exactly same evaluation metric used by the compared methods. Besides, the choices of K values are exactly following the previous works [24], [27], [43], i.e., MAP@K for K=10,20,50. All the results of our methods reported in the paper are the average results after running the model three times.

4.4 Implementation Details

All the source code of our experiments is implemented using Pytorch [45] based on the Huggingface's implementation of Transformer library[2] [46]. The configuration of the two GPT-2 models in our method is the same as the default settings of Huggingface, which has 12 transformer layers with 12 attention heads and its hidden state size is 768 [38]. The architecture of the BERT encoder we use is the same as the $BERT_{BASE}$ model [31], whose input embedding passes through 12 stacked bidirectional Transformer blocks with 768 hidden dimensions and 12 self-attention heads. For the main results we report, we initialize the BERT encoder with the weights of BERT-wwm model [47]. But for a fair comparison with CGExpan, we also use BERT-base-uncased to conduct experiments. When updating the initial seed/candidate sets (Equation 3 and Equation 4), we choose $\text{thr}_u = 0.65$ and $\text{thr}_l = 0.25$ as we report the results of our proposed method. When automatically generating the context patterns, we set m to 130 in our experiments. As for the selection strategy of these hyper-parameters, practically, we derive a validation set for each dataset and conduct a simple hyper-parameter grid search on it. More specifically, the validation set for a dataset is generated by randomly selecting seed entities from the ground truth of each semantic class. Note that the seed entities in the validation set have no overlap with seed entities in the test set. By evaluating the model performance on the validation set under different values of hyper-parameters, we obtain the optimal hyper-parameter settings. This is standard practice for determining hyper-parameters and can be easily applied to other datasets as well as other application scenarios. Additionally, we will later provide a parameter study on $\text{thr}_u, \text{thr}_l$ in the experiments to analyze how these two hyper-parameters affect the performance of our proposed model.

2. https://github.com/huggingface/transformers

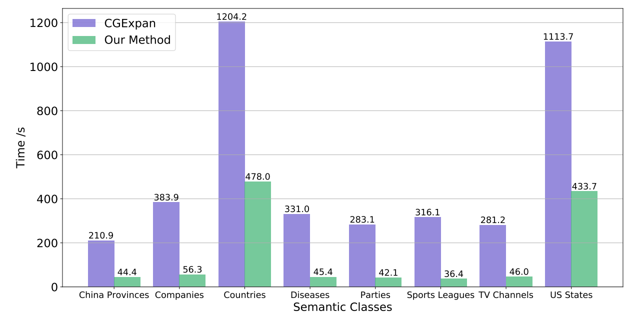


Fig. 3. Running efficiency analysis. For every class, we report the running time consumed by the two models when expanding 50 entities.

4.5 Overall Performance

From Table 1 and Table 2, we can observe that:

1) Our GAPA performs better than all baselines on all datasets and evaluation metrics. Specifically, without the participation of human-annotated corpora, GAPA

TABLE 3
CGExpan-SSE means that we perform supervision signal enhancement (SSE) module on CGExpan. GAPA-NoSSE means the ablation of GAPA without SSE module. "↑" indicates that the corresponding method receives a performance improvement after combing with SSE module.

Methods	Wiki			APR		
	MAP@10	MAP@20	MAP@50	MAP@ 10	MAP@20	MAP@50
CGExpan	0.995	0.978	0.902	0.992	0.990	0.955
CGExpan-SSE	0.996↑	0.980↑	0.916↑	0.993↑	0.992↑	0.960↑
GAPA-NoSSE	0.997	0.996	0.956	0.995	0.992	0.970
GAPA	1.000↑	1.000↑	0.974↑	1.000↑	1.000↑	0.990↑

outperforms the previous state-of-the-art corpus-based ESE models. In addition, while CGExpan uses Hearst patterns to generate class names and SynSetExpan introduces synonym information to enhance ESE, GAPA has better performance than them without any external information.

2) For different values of K in MAP@K, we can see that GAPA performs particularly well under MAP@10 and MAP@20, where it even achieves 100% performance on Wiki and APR. This phenomenon reflects the high quality of the ranked entity list obtained by our method, that is, the ranked list obtained by our method indeed ranks those entities with the correct semantic class preferentially.
3) For fine-grained semantic classes, the comparison results in Table 2 show that GAPA outperforms CGExpan under most classes significantly, which reflects that our method is more generalized than CGExpan for different semantic classes. Especially for `China Provinces/Companies` classes, the performance of the CGExpan is significantly lower than other classes, while our method narrows the performance gap among different classes. We think that the main reason for the large difference in the performance of CGExpan for different semantic classes is that the given corpus it uses has different annotation quality for different classes of entities, which also proves the shortcoming of the corpus-based ESE methods (which is described in Section 1) and reflects the advantage of our proposed corpus-independent ESE.

4.6 Efficiency Analysis

Because our proposed GAPA utilizes autoregressive language models to automatically generate context patterns compared to methods that use manually pre-defined patterns, it is necessary to conduct an in-depth comparative analysis of the efficiency of our method and previous work. For a fair comparison, we run CGExpan and GAPA under the same hardware settings. Our used CPU is the Intel(R) Xeon(R) Gold 5218R CPU @ 2.10GHz, and the GPU is the GeForce RTX 3090 with 24GB memory. We take the time that the model successfully expands 50 entities (right or wrong) as the running time for the final report of the model.

From Figure 3, we can know that GAPA takes significantly less time than CGExpan to obtain the same number of expansion entities, which demonstrates that our method does not suffer from a decrease in efficiency due to our used text generation models such as GPT-2. After comparative analysis at the method design level, we think that there are two reasons for the efficiency difference shown in Figure 3: (1) The first and mainly, in each expansion iteration, CGExpan utilizes BERT to perform the probing operations to automatically generate corresponding semantic class names, which is a very time-consuming process. But GAPA has pre-generated all context patterns with two GPT-2 models before entering the iterative expansion process, and used the BERT encoder to encode these context patterns, so there is no need to run these large-scale models such as GPT-2 and BERT iteratively. (2) The second, kinds of ranking strategies designed in CGExpan are more complex than ours. CGExpan needs to rank both the generated class names and the selected entities. The complex ranking operation increases the time consumed by CGExpan to a certain extent.

4.7 Ablation Studies

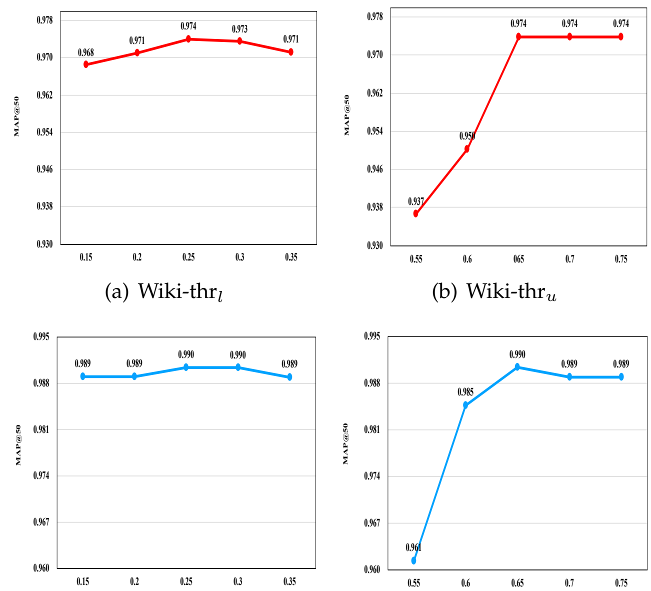
4.7.1 *Effectiveness of Supervision Signal Enhancement Module*

To verify the effectiveness of our designed supervision signal enhancement module, we analyze the performance impact of the supervision signal enhancement module on GAPA and CGExpan. From Table 3 we can see that after combining with the supervision signal enhancement module, all evaluation metrics of CGExpan and GAPA-NoSSE on the two datasets have been improved. These steady performance improvements prove that our proposed supervision signal enhancement module is helpful to alleviate the sparse supervision problem in ESE. Furthermore, the performance advantage of GAPA-NoSSE over CGExpan shows that even without the participation of the supervision signal enhancement module, our model still has a competitive performance.

4.7.2 *Parameter Sensitivity Analysis*

In our proposed GAPA framework, there are two key hyper-parameters (i.e., thr_u and thr_l) for updating the initial seed/candidate sets to enhance the supervision signal for our model. Therefore, we further analyze how these two hyper-parameters affect the overall performance of our model. Specifically, we report the model performance with various values of thr_u, thr_l on different datasets. Note that because our method performs very stable on MAP@10 and MAP@20, the performance remains at 1.000 for almost every hyper-parameter value, so we only use MAP@50 results for demonstration. As shown in Figure 4, the performance of GAPA in all cases is higher than the previous state-of-the-art performance, which reflects the stable competitive performance of our proposed model.

Figure 4(a) and Figure 4(c) show that as the thr_l changes, the model performance does not change very significantly,



(a) Wiki-thr$_l$ (b) Wiki-thr$_u$

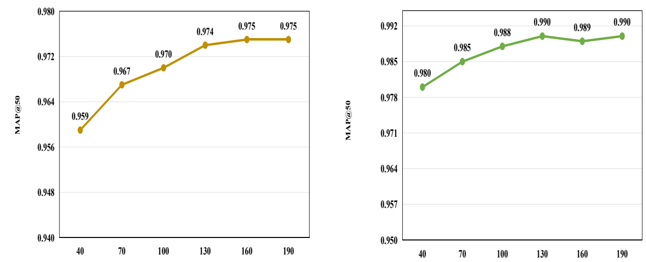
(c) APR-thr$_l$ (d) APR-thr$_u$

Fig. 4. Parameter sensitivity analysis of thr$_u$, thr$_l$ in GAPA. The MAP@50 of the state-of-the-art models on Wiki and APR are 0.926 and 0.960.

usually just to three or four decimal places. In GAPA, the direct reason why we want to use the thr$_l$ is that we can use thr$_l$ as a threshold to reduce the candidate set size, thereby reducing the search space in the expansion process. The second advantage of using thr$_l$ is that the number of expansion iterations can be decreased, thereby reducing the computational consumption and running time. Therefore, when the value of thr$_l$ changes in a relatively small interval (e.g., $[0.15, 0.35]$), it will only cause the candidate set size to change, because the entities involved in the values of these smaller thr$_l$ must be entities that are far apart in semantic space from the seed entities, the presence or absence of these entities has little effect on the model performance.

From Figure 4(b) and Figure 4(d), we can see that as the thr$_u$ increases, the performance of the model first increases and then tends to remain unchanged. As described in Section 3.2, we use thr$_u$ as a threshold to select entities from the initial candidate set to expand the initial seed set. Therefore, when the value of thr$_u$ is too small, some entities with low similarity to the seed entities will be wrongly introduced into the seed set, thereby introducing noise for the subsequent expansion process, which naturally reduces the expansion performance. Conversely, when the value of thr$_u$ increases to a certain value, the overall expansion performance of the model no longer changes significantly, which indicates that when we set thr$_u$ as a value within a suitable interval, the model performance will be not sensitive for the value of thr$_u$. This phenomenon is consistent with the design motivation that we originally want to select entities with high similarity from the candidate set to expand the initial seed set, thereby enhancing the supervision signal.

Additionally, we also analyze how the context patterns number (i.e., m) affects the performance. Figure 5(a) and Figure 5(b) show that as the m increases, there is a slight boost in performance. Benefiting from the strategy of increasing supervised signals, the performance has already reached competitive results with m of 40. Note that under various values of m, the performance of GAPA is better than the previous state-of-the-art performance, which proves that our method is not very sensitive to the value of m. To achieve a balance between efficiency and effectiveness, we recommend a setting of 130 for m in practice.



(a) Wiki-m (b) APR-m

Fig. 5. Parameter sensitivity analysis of m in GAPA. The MAP@50 of the state-of-the-art models on Wiki and APR are 0.926 and 0.960.

TABLE 4
Rouge score between generated context patterns and various corpora.

Dataset	Rouge-1	Rouge-2	Rouge-L
Wiki	0.368	0.203	0.361
SE2	0.413	0.279	0.482
Reverse-Wikipedia	0.431	0.302	0.534

TABLE 5
Perplexity and Distinct n-gram (Dist-n) of generated contexts and reference contexts.

	Perplexity↓	Dist-1↑	Dist-2↑	Dist-3↑
Generated	38.921	0.247	0.547	0.705
Wikipedia	36.263	0.171	0.485	0.653

4.7.3 Context patterns Analysis

To verify the quality of context patterns, we analyze the Rouge score between generated context patterns and various related corpus. For each initial seed entity set, we get the sentences that contain the seed entity from the corpus and regard these sentences as the reference. At the same time, generated context patterns of the seed entity are obtained via our context pattern generation module. To measure the overlap between generated context patterns and reference sentences, we calculate the average Rouge-1, Rouge-2, and Rouge-L scores between them. Specifically, for each generated context containing an entity, Rouge scores are computed between the context and reference contexts containing the same entity, and the highest Rouge scores among various reference contexts represent the similarity scores between the generated context and the corpus. Similarity scores for all generated contexts are averaged as the entire overlap between generated contexts and reference contexts.

From the results of Wiki and SE2 in Table 4, the low Rouge scores indicate that our generated context patterns do not have a high degree of coincidence with the corpus used by traditional corpus-based ESE methods, so our method

TABLE 6
Context patterns generated for corresponding entities. The 1st and 3rd columns are the prev-text and next-text automatically generated by $\overleftarrow{\text{GPT-2}}$ and $\overrightarrow{\text{GPT-2}}$, respectively.

Generated Prev-Text	Entity	Generated Next-Text
... were used to act counter-clockwise at Nanchong , or at ,	*"Sichuan"*	, China . The first two of these are the most common ...
... a football competition in	*"England"*	, which was held from 1892 to 1896 ...
... a useful clinical sign in distinguishing malaria from other causes of	*"fever"*	, but it is not a reliable diagnostic test for malaria ...
... He was purchased by the Rajasthan Royals for the 2008 and 2009 seasons of	*"Indian Premier League"*	(IPL) and was a part of the team that won the title ...
... Alfonso Foods was a fast food hamburger drive-in restaurant chain based in	*"California"*	. It was founded in 1946 by Alfredo ...
... after creating her Facebook page, Bean also joined	*"Twitter"*	and Instagram . She is a regular contributor ...
... the company design websites for many clients including,	*"Microsoft"*	, Apple, Google, Facebook , Amazon, eBay ...
... a huge area stretching from Yunnan , a southwest into	*"China"*	's far west , to the Tibetan plateau ...
... HeTe was selected as a member and joined the	*"Reform party"*	in 1983 . He was a founding member ...
... who is best known for starring in a drama with	*"TVB"*	and the Hongkong Film Production ...

does not have an information leakage problem. And the low Rouge scores of Reverse-Wikipedia reflect that GAPA generates diverse contexts related to entities, rather than simply copying what it has seen in the pre-training corpus.

Furthermore, we compare the quality of the generated contexts to those of Wikipedia, evaluating them based on fluency and diversity. We employ perplexity [48] to measure the fluency and Distinct n-gram (Dist-n) [49] to measure the diversity. GPT-Neo-1.3B [50] is utilized for computing perplexity. For fair comparison in Dist-n, we sample the same number of contexts in Wikipedia as generated contexts. The results presented in Table 5 demonstrate that GAPA's generated contexts are not only similar in fluency to those in Wikipedia, but also exhibit higher diversity. When it comes to long-tail entities, Wikipedia only contains a limited number of contexts available, making it challenging to expand uncommon entities for the ESE task. Meanwhile, GAPA is capable of generating a comparable number of contexts for each entity with high diversity, making it easier to expand uncommon but semantically relevant entities.

4.8 Case Studies

4.8.1 Context Pattern Generation

Table 6 shows some cases of our context pattern generation module for several entities from different semantic classes. We can see that the texts generated by the two GPT-2 models in opposite directions are of high quality, both in terms of relevance and fluency. Especially for the prev-text generated by $\overleftarrow{\text{GPT-2}}$, please kindly note that in practice $\overleftarrow{\text{GPT-2}}$ generates the prev-text from right to left under the guidance of the entity, which fully proves that our pre-trained reverse GPT-2 model is able to generate reverse text. To the best of our knowledge, our work firstly demonstrates that it is completely feasible to train GPT-2 with reversed corpus so that it has the ability of reverse generation.

Moreover, as can be seen from the cases in Table 6, the context patterns generated by our model inadvertently introduce many related entities, which strengthens the representation of semantic classes. We believe this is one of the reasons why our automatically generated context patterns can greatly improve the performance of the model.

4.8.2 Generated Patterns Guided Expansion

From the results of class `China Provinces` and `Companies` shown in Table 7, it can be seen that our model effectively avoids some entities with very similar semantic class when expanding. It is because CGExpan uses pre-defined patterns to generate class names, and these pre-defined patterns cannot distinguish fine-grained semantic classes (e.g., *"Anhui"* and *"Nanjing"* both belong to the `Location` class, but *"Nanjing"* belongs to `China City` and *"Anhui"* belongs to `China Province` on a more granular level). The comparison of these two semantic classes further illustrates the advantage of our proposed corpus-independent ESE paradigm, that is our context pattern generation module can perceive fine-grained semantic classes and automatically generate high-quality patterns accordingly.

In addition, it is also necessary to analyze in which case GAPA does not perform well, so as to find the direction of further improvement in the future. From Table 2, we can see that our model does not perform as well as CGExpan when faced with the `US States` class, so we further compare and analyze this semantic class in our case study. Interestingly, we find that GAPA sometimes incorrectly expands some nested entities (e.g., *"Kansas State University"* and *"Mississippi River"*). We think this phenomenon may be related to the way we automatically generate context patterns. If we use *"Mississippi"* in the seed set as the guide text to generate context patterns, then $\overrightarrow{\text{GPT-2}}$ is likely to generate the word "river" as the next-text, this will lead to GAPA wrongly judging that *"Mississippi River"* is an entity with high similarity with the seed entity set. Therefore, it is a worthwhile research direction to study how to avoid introducing noise in the context pattern generation process. We suggest that it is possible to control the context patterns generated by generative language models by optimizing decoding strategies that more closely match ESE scenarios.

TABLE 7
According to Table 2, we select the `China Provinces` and `Companies` classes because they lead to poor CGExpan's performance, and we present the results of the `US States` class to analysis in which case will GAPA have poor performance. We mark the wrong entities in red.

Seed Entity Set		CGExpan		GAPA
`China Provinces` {*“Anhui”,“Fujian”,“Hunan”*}	1	*“Hubei”*	1	*“Henan”*
	2	*“Guangdong”*	2	*“Shandong”*
	3	*“Jiangxi”*	3	*“Guangdong province”*
		...		...
	29	*“Nanjing”*	29	*“Yunnan”*
	30	*“Wuhan”*	30	*“Ningxia”*
	31	*“Inner Mongolia”*	31	*“Jiangxi”*
	32	*“Hangzhou”*	32	*“Henan province”*
	33	*“Guangzhou”*	33	*“Liaoning province”*
		...		...
`Companies` {“Myspace”,“Twitter”,“Youtube”}	1	*“Flickr”*	1	*“facebook”*
	2	*“wikipedia”*	2	*“Livejournal”*
	3	*“AOL”*	3	*“Discogs”*
		...		...
	29	*“Safari”*	29	*“AIM”*
	30	*“CNet”*	30	*“Lego”*
	31	*“Second Life”*	31	*“Mtv”*
	32	*“Microsoft Outlook”*	32	*“Macromedia”*
	33	*“Mozilla Thunderbird”*	33	*“Microsoft”*
		...		...
`US States` {“Ohio”,“Mississippi”,“Indiana”}	1	*“Illinois”*	1	*“Missouri”*
	2	*“Arizona”*	2	*“Arkansas”*
	3	*“California”*	3	*“Iowa”*
		...		...
	46	*“Alaska”*	46	*“Nevada”*
	47	*“Vermont”*	47	*“Kansas State University”*
	48	*“Kansas State”*	48	*“Orange County”*
	49	*“Hawaii”*	49	*“Mississippi River”*
	50	*“Mexico”*	50	*“Louisvile”*

5 Conclusion and Future Work

In this paper, we design a simple yet effective strategy for updating the initial seed/candidate sets and apply it to the supervision signal enhancement module to alleviate the long-standing problem of sparse supervision in ESE. More importantly, we propose to promote the ESE task by automatically generating context patterns for entities with the help of two separate GPT-2 models in opposite directions. This pioneering attempt frees our model from the dependence on manually annotated corpus and brings the ESE task into the new corpus-independent paradigm. Empirical results on three widely used benchmarks show the competitive performance of our method. In the future, we will study how to automatically measure the quality of the generated context patterns to further enhance our method.

Acknowledgments

Yinghui Li, Shulin Huang, and Xinwei Zhang realize the development methodology and the creation of models. Qingyu Zhou, Yangning Li, and Ruiyang Liu are responsible for the investigation process, experimental verification, and the writing and revision of the paper. Yunbo Cao and Hai-Tao Zheng are responsible for the management of the team, the planning of the scientific research progress, and the financial support. Ying Shen is responsible for checking the correctness and innovation of the proposed method, as well as the writing and revision of the manuscript.

References

[1] B. Shi, Z. Zhang, L. Sun, and X. Han, “A probabilistic co-bootstrapping method for entity set expansion,” 2014.

[2] X. Zhang, Y. Chen, J. Chen, X. Du, K. Wang, and J.-R. Wen, “Entity set expansion via knowledge graphs,” in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017, pp. 1101–1104.

[3] H. Zhao, C. Feng, Z. Luo, and C. Tian, “Entity set expansion from twitter,” in *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval*, 2018, pp. 155–162.

[4] Z. Chen, M. Cafarella, and H. Jagadish, “Long-tail vocabulary dictionary extraction from the web,” in *Proceedings of the Ninth ACM international conference on Web search and data mining*, 2016, pp. 625–634.

[5] C. Shi, J. Ding, X. Cao, L. Hu, B. Wu, and X. Li, “Entity set expansion in knowledge graph: a heterogeneous information network perspective,” *Frontiers of Computer Science*, vol. 15, no. 1, pp. 1–12, 2021.

[6] R. C. Wang, N. Schlaefer, W. W. Cohen, and E. Nyberg, “Automatic set expansion for list question answering,” in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Honolulu, Hawaii: Association for Computational Linguistics, Oct. 2008, pp. 947–954. [Online]. Available: https://aclanthology.org/D08-1099

[7] H. Zhao, C. Feng, Z. Luo, and Y. Pei, “Entity set expansion on social media: A study for newly-presented entity classes,” in *Chinese National Conference on Social Media Processing*. Springer, 2017, pp. 116–128.

[8] L. Yan, X. Han, L. Sun, and B. He, “Learning to bootstrap for entity set expansion,” 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)*, 2019, pp. 292–301.

[9] L. Yan, X. Han, B. He, and L. Sun, “End-to-end bootstrapping neural network for entity set expansion,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 9402–9409.

[10] L. Yan, X. Han, and L. Sun, “Progressive adversarial learning for bootstrapping: A case study on entity set expansion,” in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 9673–9682. [Online]. Available:

https://aclanthology.org/2021.emnlp-main.762
[11] P. Velardi, S. Faralli, and R. Navigli, "OntoLearn reloaded: A graph-based algorithm for taxonomy induction," *Computational Linguistics*, vol. 39, no. 3, pp. 665–707, Sep. 2013. [Online]. Available: https://aclanthology.org/J13-3007
[12] X. Rong, Z. Chen, Q. Mei, and E. Adar, "Egoset: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion," in *Proceedings of the Ninth ACM international conference on Web search and data mining*, 2016, pp. 645–654.
[13] J. Mamou, O. Pereg, M. Wasserblat, A. Eirew, Y. Green, S. Guskin, P. Izsak, and D. Korat, "Term set expansion based NLP architect by Intel AI lab," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 19–24. [Online]. Available: https://aclanthology.org/D18-2004
[14] J. Huang, Y. Xie, Y. Meng, J. Shen, Y. Zhang, and J. Han, "Guiding corpus-based set expansion by auxiliary sets generation and co-expansion," in *Proceedings of The Web Conference 2020*, 2020, pp. 2188–2198.
[15] C. Zhang, Y. Li, N. Du, W. Fan, and P. S. Yu, "Entity synonym discovery via multipiece bilateral context matching," *arXiv preprint arXiv:1901.00056*, 2018.
[16] A. Barrena, A. Soroa, and E. Agirre, "Alleviating poor context with background knowledge for named entity disambiguation," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1903–1912. [Online]. Available: https://aclanthology.org/P16-1179
[17] P. Pantel, E. Crestan, A. Borkovsky, A.-M. Popescu, and V. Vyas, "Web-scale distributional similarity and entity set expansion," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 2009, pp. 938–947.
[18] J. Shen, Z. Shen, C. Xiong, C. Wang, K. Wang, and J. Han, "Taxoexpan: Self-supervised taxonomy expansion with position-enhanced graph neural network," in *Proceedings of The Web Conference 2020*, 2020, pp. 486–497.
[19] P. Yu, R. Rahimi, Z. Huang, and J. Allan, "Learning to rank entities for set expansion from unstructured data," in *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*, 2020, pp. 21–28.
[20] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *Seventh IEEE international conference on data mining (ICDM 2007)*. IEEE, 2007, pp. 342–350.
[21] S. Tong and J. Dean, "System and methods for automatically creating lists," Mar. 25 2008, uS Patent 7,350,187.
[22] G. Kushilevitz, S. Markovitch, and Y. Goldberg, "A two-stage masked LM method for term set expansion," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 6829–6835. [Online]. Available: https://aclanthology.org/2020.acl-main.610
[23] P. Yu, Z. Huang, R. Rahimi, and J. Allan, "Corpus-based set expansion with lexical features and distributed representations," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 1153–1156.
[24] Y. Zhang, J. Shen, J. Shang, and J. Han, "Empower entity set expansion via language model probing," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 8151–8160. [Online]. Available: https://aclanthology.org/2020.acl-main.725
[25] A. Zupon, M. Alexeeva, M. Valenzuela-Escárcega, A. Nagesh, and M. Surdeanu, "Lightly-supervised representation learning with global interpretability," in *Proceedings of the Third Workshop on Structured Prediction for NLP*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 18–28. [Online]. Available: https://aclanthology.org/W19-1504
[26] Y. Li, Y. Li, Y. He, T. Yu, Y. Shen, and H.-T. Zheng, "Contrastive learning with hard negative entities for entity set expansion," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 1077–1086. [Online]. Available: https://doi.org/10.1145/3477495.3531954
[27] J. Shen, W. Qiu, J. Shang, M. Vanni, X. Ren, and J. Han, "SynSetExpan: An iterative framework for joint entity set expansion and synonym discovery," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 8292–8307. [Online]. Available: https://aclanthology.org/2020.emnlp-main.666
[28] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," *Science China Technological Sciences*, pp. 1–26, 2020.
[29] C. Dong, Y. Li, H. Gong, M. Chen, J. Li, Y. Shen, and M. Yang, "A survey of natural language generation," *arXiv preprint arXiv:2112.11739*, 2021.
[30] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heinz, and D. Roth, "Recent advances in natural language processing via large pre-trained language models: A survey," *arXiv preprint arXiv:2111.01243*, 2021.
[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423
[32] 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," *arXiv preprint arXiv:1910.10683*, 2019.
[33] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
[34] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon, "Unified language model pre-training for natural language understanding and generation," *arXiv preprint arXiv:1905.03197*, 2019.
[35] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.
[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
[37] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.
[38] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
[39] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.
[40] U. Germann, "Greedy decoding for statistical machine translation in almost linear time," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003, pp. 72–79.
[41] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, and D. Batra, "Diverse beam search: Decoding diverse solutions from neural sequence models," *arXiv preprint arXiv:1610.02424*, 2016.
[42] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," *arXiv preprint arXiv:1805.04833*, 2018.
[43] J. Shen, Z. Wu, D. Lei, J. Shang, X. Ren, and J. Han, "Setexpan: Corpus-based set expansion via context feature selection and rank ensemble," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 288–304.
[44] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*, 1992. [Online]. Available: https://aclanthology.org/C92-2082
[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.
[46] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural*

Language Processing: System Demonstrations. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[47] Y. Cui, W. Che, T. Liu, B. Qin, Z. Yang, S. Wang, and G. Hu, "Pre-training with whole word masking for chinese bert," *arXiv preprint arXiv:1906.08101*, 2019.

[48] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 2122–2132.

[49] J. Li, M. Galley, C. Brockett, J. Gao, and W. B. Dolan, "A diversity-promoting objective function for neural conversation models," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 110–119.

[50] S. Black, L. Gao, P. Wang, C. Leahy, and S. Biderman, "Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow, 2021," *URL: https://doi. org/10.5281/zenodo*, vol. 5297715, 2022.

Yinghui Li received the BEng degree from the Department of Computer Science and Technology, Tsinghua University, in 2020. He is currently working toward the PhD degree with the Tsinghua Shenzhen International Graduate School, Tsinghua University. His research interests include natural language processing and deep learning.

Shulin Huang received the BEng degree from the Department of Computer Science and Engineering, Northeastern University, in 2021. She is currently working toward Master's degree with the Tsinghua Shenzhen International Graduate School, Tsinghua University and mainly focusing on natural language processing.

Xinwei Zhang received the BEng degree from the Department of Intelligence Science and Technology, Sun Yat-sen University, in 2021. His research interests include natural language processing and deep learning.

Qingyu Zhou received his PhD degree in a joint program of Harbin Institute of Technology and Microsoft Research Asia. He is currently a Senior Researcher at Tencent, Beijing, China. His research interests include text summarization, natural language generation, dialogue system and educational applications of NLP.

Yangning Li received the BEng degree from the Department of Computer Science and Technology, Huazhong University of Science and Technology, in 2020. He is currently working toward a Master's degree with the Tsinghua Shenzhen International Graduate School, Tsinghua University. His research interests include natural language processing and data mining.

Ruiyang Liu received the BEng degree from the Department of Computer Science and Technology, Tsinghua University, in 2020. He is currently working toward Master's degree with the Department of Computer Science and Technology, Tsinghua University and mainly focusing on deep learning, computer vision, and discrete differential equation.

Yunbo Cao received his PhD degree in computer science from the Shanghai Jiao Tong University, in 2011. He is currently the head of the NLP Center of Tencent Cloud Xiaowei and mainly focuses on various applications of NLP.

Hai-Tao Zheng received the bachelor's and master's degrees in computer science from the Sun Yat-Sen University, China, and the PhD degree in medical informatics from Seoul National University, South Korea. He is currently an associate professor with the Shenzhen International Graduate School, Tsinghua University, and also with Peng Cheng Laboratory. His research interests include web science, semantic web, information retrieval, and machine learning.

Ying Shen received the PhD degree in computer science from the University of Paris Ouest Nanterre La Défense, France and the Erasmus Mundus master's degree in natural language processing from the University of Franche-Comt, France and the University of Wolverhampton, U.K. She is currently an associate professor with the School of Intelligent Systems Engineering, Sun Yat-Sen University. Her research interests include natural language processing and deep learning.