Negative-aware Entity Set Expansion

Anonymous ACL submission

Abstract

Entity Set Expansion (ESE) aims to find all entities of one target semantic class with a few seed entities describing it. However, existing ESE methods cannot express what entities we explicitly dislike, and thus hinder its application in real-world scenarios. In this paper, to endow models with the capability of understanding the "dislike" relationship among seed entities, we express the target semantic class with both positive and negative seed entities. To this end, we propose an efficient and learnable negative-aware entity set expansion framework, which is essentially a retrieval model. To facilitate this study, a large-scale Negative-aware ESE Dataset (NED) with more than 1M entities is further collected and annotated. Extensive experiments[1] on NED show that the proposed framework can effectively understand the dislike relations expressed by the negative seeds and expand fewer dislike entities than baseline methods.

1 Introduction

The Entity Set Expansion (ESE) task aims to find all entities of one semantic class with a few seed entities describing it. For example, given {"apple", "banana", "pear"}, ESE tries to find other entities in the target semantic class *Fruit*, such as "orange" and "grape". ESE benefits a variety of downstream NLP and IR applications, such as web search(Chen et al., 2016), question answering (Wang et al., 2008), taxonomy construction(Velardi et al., 2013), and semantic search(Xiong et al., 2017).

Though achieving reasonably good results on existing ESE datasets, current ESE methods can only describe what entities we want, while failing to express what we explicitly don't want, which hinders the application of ESE. For instance, when the user wishes to expand *Snacks* ***without*** *Peanuts* (to prevent food allergy), existing ESE methods cannot express it and tend to return all snack entities

[1]Code and dataset will be available for reproducibility.

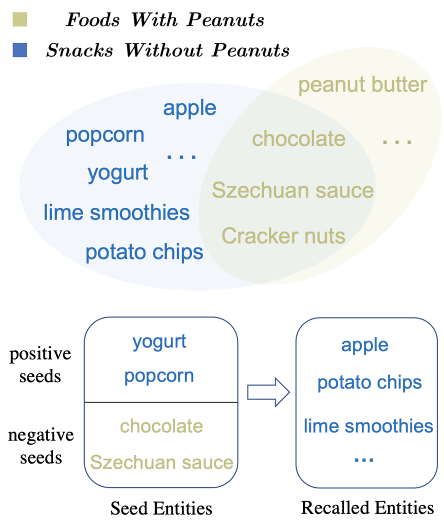

Figure 1: Negative-aware entity set expansion.

from the candidate entity set without filtering those containing peanuts. Obviously, it is natural and necessary to endow ESE models with the capability to understand the "dislike" relationship among seed entities. To achieve this, we express the target semantic class with both positive seeds and negative seeds as shown in Figure 1. Specifically, for semantic class *Snacks without Peanuts*, we may use {"yogurt", "popcorn"} as positive seeds to describe what we want; and use {"chocolate", "Szechuan sauce"} as negative seeds to express what we dislike (foods containing peanuts).

Compared with ESE, negative-aware entity set expansion can better benefit some downstream tasks. For example, in personalized recommendation systems, a user can mark the recommendation results as liked (positive seed entities) or disliked (negative seed entities). Utilizing these liked and disliked results, the system will be able to understand the preference (target negative-aware semantic class) of the user more accurately and consequently give better results.

In this paper, we propose the NegESE, a learnable Negative-aware Entity Set Expansion framework. Given the input of a few positive and negative seeds, NegESE learns to understand the target negative-aware semantic class and find all entities in this semantic class from the candidate entity set. NegESE consists of three modules: (1) The first, *entity encoding module*, maps each entity into its dense representation using BERT(Devlin et al., 2019) and the corpus. (2) The second, *entity set comprehension module*, learns what kind of entities are liked and disliked given positive and negative seeds respectively, and combines these two requirements to model the target semantic class. (3) The third, *entity retrieval module*, retrieves entities from the candidate entity set with the semantic meaning learned by the previous module.

To facilitate the study of negative-aware entity set expansion, we construct the Negative-aware ESE Dataset (NED) based on the English Wikipedia dump[2] and *Harry Potter* series. Specifically, we first extract the candidate entity set from the corpus and identify several coarse-grained semantic classes like *Chemical Elements*, *Sport Leagues* in it. We then annotate several attributes (e.g., <is_radioactive>) for each entity in these classes. Finally, we use these attributes to automatically generate negative-aware samples of different granularities such as *Radioactive Chemical Elements* with attribute value set { <is_radioactive>=True }. The full process is shown in Figure 2 and will be introduced in Section 3.

In summary, our contributions are in three folds:

1. We propose NegESE, a simple and effective learnable framework, to solve the negative-aware entity set expansion task efficiently.

2. We construct NED, a large-scale negative entity set expansion dataset with a great amount of negative-aware semantic samples in different granularities, based on general domain corpus and domain-specific corpus.

3. We conduct comprehensive experiments on NED, which demonstrate the effectiveness and efficiency of NegESE in solving negative-aware entity set expansion task.

2 Related Work

2.1 Methods of ESE

ESE is a weakly supervised task, which is typically given seed entities as supervised signals and expands with new entities from the candidate entity set. Research of ESE in the last decade can be divided into two main categories: (1) One-time ranking methods (Yu et al., 2019a) compute similarity on the basis of semantic features of the seed entities and expand a ranked list of new entities that belong to the same semantic class. (2) Bootstrap-based methods (Huang et al., 2020; Zhang et al., 2020a; Shen et al., 2017a; Shi et al., 2014; He and Xin, 2011; Mamou et al., 2018) iteratively select context patterns around entities, and use extracted patterns to find new entities.

It is worth mentioning that there has been some work to incorporate negative entities (Jindal and Roth, 2011; Gupta and Manning, 2014; Shi et al., 2014; Curran et al., 2007), though the usage of negative entities in these methods is relatively naive. Taking (Shi et al., 2014) as an example, they expand negative entities along with positive entities in a symmetrical manner to improve the expansion performance of positive entities.

We point out that the role of the negative entities used in previous work is fundamentally different from ours. Negative entities in previous work are purely used to help determine the boundary of the target set described by positive seeds. In contrast, negative entities in our model are used to describe the target negative-aware semantic classes that cannot be characterized by positive seeds alone.

2.2 Data Resources of ESE

Common datasets used in ESE including CoNLL (Zupon et al., 2019), OntoNotes(Zupon et al., 2019), Wiki(Ling and Weld, 2012) and APR (Shen et al., 2017a). There are three main limits with these datasets: (1) Existing datasets use solely positive seed entities to describe the target semantic class which is not sufficient if the user explicitly dislikes some entities (e.g. *Snacks Without Peanuts*). (2) The semantic class granularity is coarse. Existing ESE datasets lack fine-grained semantic classes like *USA Software Companies* while only containing coarse-grained semantic classes like *Companies*. (3) Few semantic classes. The most used two datasets Wiki and APR contain only 3 and 8 classes respectively, which is too small to be representative for real scenarios.

3 NED Dataset

To facilitate the study of negative-aware entity set expansion, we need a dataset that contains both

[2] https://meta.wikimedia.org/wiki/Data_dump_torrents#English_Wikipedia

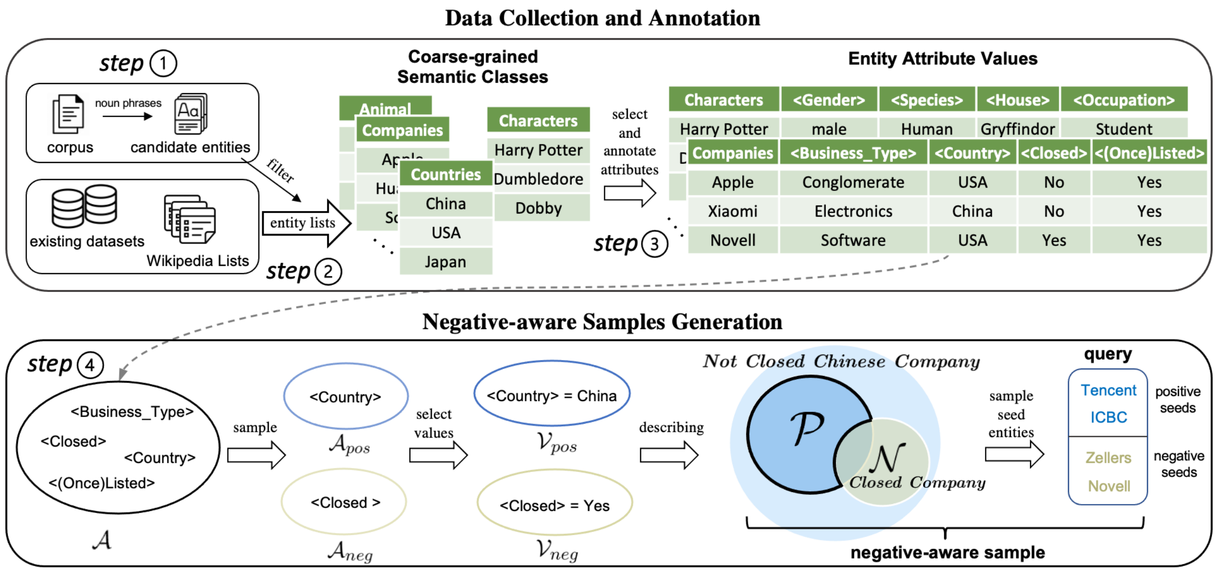

Figure 2: Illustration of the NED dataset construction process. The whole process composes the *Data Collection and Annotation* stage and the *Negative-aware Samples Generation* stage, and can be further divided into four steps.

positive and negative seed entities. However, to the best of our knowledge, there is no such public benchmark dataset. Therefore, we build the NED, a large ESE dataset containing both positive and negative seed entities involving negative-aware semantic classes of different granularities.

To examine the universality of NegESE in different scenarios, NED contains two sub-datasets with entities from two different sources: Wikipedia and *Harry Potter* series. We name the former one as NED-wiki, and the latter one as NED-hp.

3.1 Dataset Construction

We first select corpora of NED and extract the candidate entities. Then we identify several coarse-grained semantic classes based on existing ESE datasets and Wikipedia Lists[3]. Finally, we annotate a few attributes for each entity in these classes and use the attribute values to generate fine-grained negative-aware samples. We will introduce the detail of the construction process in this section, and Figure 2 is an illustration of it.

Step 1. Corpora Selection and Entity Set Extraction. For NED-wiki, we use the corpus of SE2 (Shen et al., 2020) which is an English Wikipedia dump as it provides enough general domain context information for algorithms to explore. For NED-hp, we use the popular *Harry Potter* series text as our raw corpus, which further improves the domain diversity of our dataset. We extract all noun phrases with frequency above 10 as the candidate entity set for both NED-wiki and NED-hp.

Step 2. Coarse-grained Semantic Class Selection. For NED-wiki, we identify 18 coarse-grained semantic classes based on the APR (Shen et al., 2017a) dataset, Wiki (Ling and Weld, 2012) dataset, and Wikipedia Lists. For classes in existing datasets, we just adopt the origin entity list. For classes we newly introduced, such as *Chemical Elements*, we adopt the corresponding Wikipedia List as its entity list. We filtered entities that cannot be found in the candidate entity set (extracted in Step 1) from these lists. For NED-hp, we select *Characters* as the coarse-grained semantic class and collected character names in the candidate entity set as the entity list. These semantic classes on average contains 198 entities, and the full list of class sizes can be found in Table 8 in appendix.

Step 3. Entity Attribute Annotation. For each coarse-grained semantic class C, we manually select k independent attributes $\mathcal{A} = \{a_1, a_2, \cdots, a_k\}$, which will be used to generate negative-aware samples (introduced in next step). Attributes of each coarse-grained class are shown in Table 9 in appendix. To annotate the attributes for each entity in C, we visit the corresponding Wikipedia page[4] and manually extract the attribute values from the page. Example of entity and attribute values in class *Company* are shown in the top-right of Figure 2.

Step 4. Generation of Negative-aware Samples.

[3] https://en.wikipedia.org/wiki/List_of_lists_of_lists

[4] For the *harry potter* sub-dataset, we use http://magical-menagerie.com/ as data source.

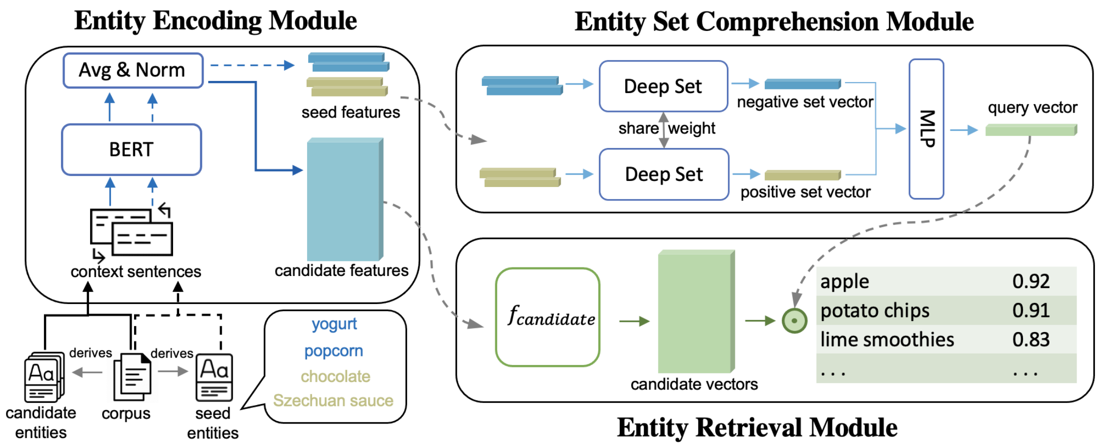

Figure 3: The architecture of the processed NegESE framework, which consists an Entity Encoding Module (EEM), an Entity Set Comprehension Module (ESCM) and an Entity Retrieval Module (ERM). EEM takes the context sentences of candidate entities and seed entities to generate features for them separately. ESCM takes the seed features generated by EEM to learn a query vector representing the semantic meaning of input seed entities (i.e. the query). ERM computes a score for each candidate entity given the candidate features and query vector to generate a ranked list of candidate entities.

We designed an algorithm to automatically generate negative-aware samples of different granularities based on the selected coarse-grained semantic classes,where each sample describes a certain semantic class. The overview of this algorithm is shown in Figure 2.

For each coarse-grained semantic class C and its attributes $\mathcal{A}$, we sample n_{pos} attributes as positive (denoted as $\mathcal{A}_{pos}$) and n_{neg} attributes as negative (denoted as $\mathcal{A}_{neg}$). We then select one value for each attribute from the attribute annotations of entities in C. In this way, we can get a list of positive values $\mathcal{V}_{pos}$ and a list of negative values $\mathcal{V}_{neg}$ describing the "like" and "dislike" requirement respectively. We use $\mathcal{P}$ to represent the entities in the candidate set that satisfy both the "like" and "dislike" requirements, and use $\mathcal{N}$ to represent the explicitly disliked (negative) entities in the set. We then randomly pick five positive seed entities and three negative seed entities from $\mathcal{P}$ and $\mathcal{N}$ respectively to form a query. For instance (shown in Figure 2), using *Company* as the base coarse-grained semantic class, we can generate a negative-aware sample describing *Not Closed Chinese Company* with $\mathcal{V}_{pos}$ = (<Country> = "China") and $\mathcal{V}_{neg}$ = (<Closed> = Yes).

A large number of samples can be automatically generated by changing the attributes or the values of attributes. We can also produce samples of different granularities by changing the size of $\mathcal{A}_{pos}$ and $\mathcal{A}_{neg}$. Note to mention that $\mathcal{A}_{neg}$ can be empty to represent the case in which no "dislike" requirement is needed. By changing the seed entities picked from $\mathcal{P}$ and $\mathcal{N}$, we can also generate multiple queries describing the same negative-aware semantic class, which consequently forms multiple samples describing the same semantic class.

Dataset	#Classes	#Samples
APR (Shen et al., 2017a).	3	15
Wiki (Ling and Weld, 2012)	8	40
SE2 (Zhang et al., 2020b)	60	1,200
NED-wiki	1,473	34,721
NED-hp	35	890

Table 1: Number of semantic classes and samples in NED and other ESE datasets

3.2 Dataset Analysis

We analyze some properties of the NED dataset from the following aspects.

Dataset Scale. NED-wiki corpus provides 149 million sentences and 1.57 million candidate entities. NED-hp provides 66,518 sentences and 1,967 candidate entities. We annotate 3,628 entities with 12,894 attribute values for 18 coarse-grained semantic classes in NED-wiki and 122 entities with 488 attribute values for the coarse-grained semantic class in NED-hp.

Number of Samples and Semantic Classes. For NED-wiki, we generated 34,721 negative-aware samples describing 1,473 distinct semantic classes. These semantic classes consist on average 76 entities in $\mathcal{P}$ and 41 entities in $\mathcal{N}$. For NED-hp, we generated 890 negative-aware samples describing 35 distinct semantic classes, which consist on average 55 entities in $\mathcal{P}$ and 28.6 entities in $\mathcal{N}$. We compare the number of samples and semantic classes of NED and existing ESE datasets in Table 1. For NED-wiki, we split 34,429, 128, 164 samples into the train, val and test split respectively (representing 1,294, 101, 78 distinct semantic classes without overlap). For NED-hp, we split 726, 80, 84 samples into the train, val and test split respectively (representing 20, 5, 10 distinct semantic classes without overlap).

Semantic Class Granularity. By changing the size of A_{neg} and A_{pos}, we can get samples of different granularities. The detail distribution of numbers of distinct semantic classes in different granularity is shown in Figure 5 and Figure 6 in appendix.

4 NegESE

In this section, we introduce the proposed NegESE framework in detail. First, we present the entity encoding module in Section 4.1. Then, we discuss our entity set comprehension module and entity retrieval module in Section 4.2 and Section 4.3 respectively. Finally, we introduce the training and inference process in Section 4.4.

4.1 Entity Encoding Module

We apply a BERT model (Devlin et al., 2019) with the pre-trained weight in (Wolf et al., 2020) to extract entity features from the corpus. Specifically, for each entity e_i, we first sample at most k_s sentences containing e_i from the corpus as its context features. For each sentence $s_j, j \in [1, k_s]$, we pass the text into the BERT model to get one feature vector v_j using the outputs of e_i. We then take the normalized average of all vectors of e_i as the feature $\mathbf{x}_i \in \mathrm{R}^d$, where d is the output size of BERT.

4.2 Entity Set Comprehension Module

The seed comprehension module inputs a small collection of positive (negative) seed entities S_{pos} (S_{neg}) and output the vector representation for the positive (negative) seed set.

We first define a basic encoding operation f as:

$$f_\theta(\mathbf{v}) = \mathrm{MLP}_\theta(\mathbf{v}) + \mathbf{v} \tag{1}$$

where MLP_θ is a two layer perception with dropout and non-linear activation. For seed set S, we generate its set representation using a Deep Set model (Zaheer et al., 2018):

$$\mathbf{S} = f_{\text{set}}(\sum_{e_i \in S} f_{\text{ele}}(W_1 \mathbf{x}_i + \mathbf{b}_1)) \tag{2}$$

where $W_1 \in \mathbb{R}^{H \times d}$ is a learnable matrix, $\mathbf{b}_1 \in \mathbb{R}^H$ is a bias vector, H is the hidden size. f_{set} and f_{ele} are two different trainable module using the structure defined in Equation (1). We then further normalize $\mathbf{S}_{pos}$ and $\mathbf{S}_{neg}$ using a layer normalization module (Ba et al., 2016) to help the model coverage faster.

As mentioned before, the input positive and negative seed entities describe a negative-aware semantic class. Next, we explore the semantic meaning of this class. We first concatenate the positive set representation $\mathbf{S}_{pos}$ and the negative set representation $\mathbf{S}_{neg}$, and then embed the result into $\mathbf{q} \in \mathbb{R}^H$ use a simple two-layer MLP introduced in Equation (1):

$$\mathbf{q} = \mathbf{MLP}([\mathbf{S}_{pos}; \mathbf{S}_{neg}]) \tag{3}$$

4.3 Entity Retrieval

For each entity e_j in the candidate entity set $\mathcal{E}$, we encode it using the module defined in Equation(1):

$$\mathbf{e}_j = f_{\text{candidate}}(\mathbf{x}_j) \tag{4}$$

With $\mathbf{e}_j$ and query representation $\mathbf{q}$, we can calculate score for each entity e_j using the cosine similarity between $\mathbf{e}_j$ and $\mathbf{q}$:

$$s_j = < \mathbf{q}, \mathbf{e}_j > \tag{5}$$

4.4 Training and Inference

In general, the performance of a model will benefit from enough negative entities during the training stage. Therefore, besides the negative candidate entities in $\mathcal{N}$, we also further sample k_e entities from the candidate set for each sample to provide additional supervision. We use $\mathcal{N}'$ to represent the union of sampled entities and entities in $\mathcal{N}$. Given the positive entity set $\mathcal{P}$, the negative entity set $\mathcal{N}'$ and score for each entity in these sets, we adopt a binary cross-entropy loss to formulate our training objective L as:

$$L = -\sum_{e_j \in \mathcal{P}} \log s_j - \sum_{e_j \in \mathcal{N}'} \log(1 - s_j) \tag{6}$$

During the inference stage, we compute s_j for all entities in the candidate entity set $\mathcal{E}$, and sort them in descending manner to get the ranked list L_q of all candidate entities.

Method	NED-wiki MAP@k↑				NED-hp MAP@k↑			
	k=10	k20	k=50	k=100	k=10	k20	k=50	k=100
SetExpan (Shen et al., 2017b)	12.53	10.04	5.93	4.19	19.97	11.34	7.04	6.97
CaSE (Yu et al., 2019b)	26.29	18.45	12.30	8.30	26.71	18.26	13.02	9.84
CGExpan (Zhang et al., 2020b)	37.58	31.68	23.14	-	15.31	14.61	13.15	12.84
NegESE	**40.67**	**38.54**	**33.79**	**26.41**	**67.88**	**68.60**	**70.35**	**67.85**

Table 2: Overall performance comparison with state-of-the-arts on NED

Method	NED-wiki		NED-hp	
	MeanNeg@100↓	MeanNeg@200↓	MeanNeg@20↓	MeanNeg@50↓
SetExpan (Shen et al., 2017b)	24.53	34.91	42.50	57.50
CaSE (Yu et al., 2019b)	22.43	25.23	32.50	67.50
CGExpan (Zhang et al., 2020b)	42.05*	-	27.50	100.0
NegESE	**8.33**	**11.11**	**20.00**	**40.00**

Table 3: Negative candidate entity intrusion result comparison with state-of-the-arts on NED. Since CGExpan only expand 50 entities in our experiments, we use the MeanNeg@50 score which is strictly smaller than or equals to the expected MeanNeg@100 score in this table.

5 Experiments

In this section, we first compare the performance of NegESE with existing state-of-the-art models on NED. We then analyze the effectiveness of negative seed entities, expansion performance on different-grained semantic classes, and the efficiency of NegESE. Finally, we present a real case produced by NegESE on NED-wiki. All results shown in this section is conducted on NED-wiki test split and NED-hp test split.

5.1 Compared Methods

We compare NegESE with the following methods:

(1) **SetExpan** (Shen et al., 2017b) is a bootstrap-based method that utilize a rank ensemble mechanism to select entities in each bootstrap iteration.

(2) **CaSE** (Yu et al., 2019b) is an efficient one-time ranking method based on SetExpan utilizing lexical features as well as pretrained distributed representations of entities.

(3) **CGExpan** (Zhang et al., 2020b) is one of the state-of-the-art models for ESE. It queries BERT (Devlin et al., 2019) to generate the name of the target semantic class and iteratively extends the set of entities employing class names. Note to mention that CGExpan is too slow and costs tremendous running memory, so we only expand 50 and 100 entities for samples in NED-wiki and NED-hp respectively to make the running time affordable. We also reduce the size of candidate entity set $\mathcal{E}$ to be one-third for NED-wiki to save running memory and make it runnable on a machine with 256G RAM. When reducing the candidate entity set $\mathcal{E}$, we keep all entities in any $\mathcal{P}$, and consequently only make the expansion easier for CGExpan.

5.2 Evaluation Metrics

Following previous work, we use $\mathrm{MAP@k}$ (Mean Average Precision) to measure the performance of the retrieved top-k entities:

$$\mathrm{MAP@k} = 100 \times \frac{1}{|Q|} \sum_{q \in Q} \mathrm{AP}_k \left(L_q, \mathcal{P}\right) \quad (7)$$

where Q stands for the set of all queries. For each query q, we use $\mathrm{AP}_k \left(L_q, \mathcal{P}\right)$ to denote the traditional average precision at position k given the ranked list of candidate entities and a ground-truth set of target semantic class $\mathcal{P}$. Methods that recall candidate entities in $\mathcal{P}$ more accurately will get higher $\mathrm{MAP@k}$ scores.

In addition, to measure the intrusion of negative candidate entities in $\mathcal{N}$ which are obviously disliked by the user, we design the $\mathrm{MeanNeg@k}$ metric based on the implementation of Top-k metric in (Karpukhin et al., 2020):

$$\mathrm{MeanNeg@k} = 100 \times \frac{1}{|Q|} \sum_{q \in Q} \mathrm{Neg_k}(L_q, \mathcal{N}) \quad (8)$$

Method	NED-wiki MAP@k↑				NED-hp MAP@k↑			
	k=10	k20	k=50	k=100	k=10	k20	k=50	k=100
NegESE(w.o. neg)	33.28	29.02	25.26	20.16	65.14	67.05	67.50	65.47
NegESE	**40.67**	**38.54**	**33.79**	**26.41**	**67.88**	**68.60**	**70.35**	**67.85**

Table 4: Comparing entity set expansion performance of NegESE(w.o. neg) and NegESE on NED

Method	NED-wiki		NED-hp	
	k=100	k=200	k=20	k=50
NegESE(w.o. neg)	13.89	18.52	25.00	50.00
NegESE	**8.33**	**11.11**	**20.00**	**40.00**

Table 5: Comparing negative candidate entity intrusion result of NegESE(w.o. neg) and NegESE on NED. The metric used in this table is MeanNeg@k.

where $\mathrm{Neg_k}$ returns 1 if top-k entities in L_q contains at least one entity in $\mathcal{N}$, otherwise 0. Higher $\mathrm{NegMean@k}$ scores mean the method tends to recall "disliked" entities for more queries (samples).

5.3 Implementation Details

To ensure fairness of comparison, we report experiment results of $k = 10, 20, 50, 100$ for $\mathrm{MAP@k}$ as previous works and use $k = 20, 50, 100, 200$ for $\mathrm{NegMean@k}$. More details are shown in Table 10.

5.4 Main Results

We evaluate our proposed NegESE framework and existing state-of-the-art methods on NED and the overall results are recorded in Table 2.

Overall Analysis: Surprisingly, we observe that on both datasets, NegESE outperforms the state-of-the-art ESE method with a large margin across all metrics. We attribute this improvement to the utilization of negative seeds and the learnable framework of NegESE since it performs much better than another BERT-based method CGExpan.

Domain Adaptability: When coming to the domain-specific scenario (i.e. on NED-hp), the advantages of the learnable framework become more prominent. Since the pre-trained BERT contains less knowledge about the *Harry Potter* series, the performance of CGExpan drops significantly from NED-wiki to NED-hp, and even worse than CaSE. On the contrary, NegESE performs very well in this domain-specific scenario, demonstrating that a learnable framework is more preferable when ESE needs to be applied to a specific domain.

Negative Entity Intrusion: We also investigate the intrusion of negative entities, which is the main problem in this paper. The results are recorded in Table 3, higher $\mathrm{NegMean@k}$ scores means the method tends to recall disliked entities described the negative seed entities for more samples. We find that NegESE gives the lowest scores compared to other ESE methods on both datasets. We attribute this to the utilization of negative seed entities, as existing methods lack the capability to utilize the negative seed entities conveying the "dislike" requirement and consequently give higher $\mathrm{NegMean@k}$ scores. For example, though achieves decent $\mathrm{MAP@k}$ performance, CGExpan gets poor $\mathrm{MeanNeg@k}$ scores, because it ignores the negative seed entities.

5.5 Further Analysis

Effectiveness of Negative Seed Entities. We develop a baseline model NegESE(w.o. neg) to find out the importance of negative seeds in expansion, which has the same architecture with NegESE while do not use the negative seed entities during expansion. To be specific, we replace the *negative set vector* in Figure 3 with a zero-filled vector. The detailed experiment results are shown in Table 4 and Table 5. We find that removing negative seeds from inputs leads to performance drops in all metrics, showing the necessity of negative seeds, and NegESE has the capability to utilize the "dislike" requirement conveyed by negative seeds to improve the expansion quality and mitigate negative entity intrusion problem.

Expansion Performance on Different-grained Semantic Classes. We further analyze the performance of our method on samples of different granularities, the full results are recorded in Table 7. For samples with certain size of $\mathcal{A}_{neg}$, we can get finer-grained samples by adding more attributes to the $\mathcal{A}_{pos}$, which consequently reduce the size of $\mathcal{P}$ in Figure 2. For samples with certain size of $\mathcal{A}_{pos}$, we can also get samples of different granularities by changing the size of $\mathcal{A}_{neg}$, which consequently changes the size of $\mathcal{N}$. Note to mention, if

Seed Entity Set	Semantic Class	CGExpan		NegESE	
Positive: { "Enron", "Motown", "France Telecom" "British Aerospace", "Northrop Grumman" } *Negative*: {"Toyota", "Namco", "Mazda" }	***Companies Not In Japan***	1	IBM	1	"Halliburton Energy Services"
		2	"BAE Systems"	2	"Texas Instruments"
			...		...
		18	"Microsoft"	18	"McDonald's"
		19	"Philips"	19	"Northrop Grumman Corporatoin"
		20	"Sony"	20	"American Express"
		21	"Compaq"	21	"Freescale"
		22	"SAP"	22	"General Dynamics Corp"
		23	"Fujitsu"	23	"Hasbro Inc."

Table 6: Expanded entity set of a sample from NED , entities in the negative candidate set $\mathcal{N}$ are colored in red.

$\|\mathcal{A}_{pos}\|$	$\|\mathcal{A}_{neg}\|$	MAP@k			
		k=10	k=20	k=50	k=100
0	**0**	59.24	59.40	55.11	44.32
1	**0**	28.63	25.55	24.20	19.67
2	**0**	16.83	11.95	12.79	17.03
0	**1**	38.17	40.38	29.68	21.84
0	**2**	49.32	47.06	40.38	29.68
0	$+\infty$	59.24	59.40	55.11	44.32

Table 7: Expansion performance of NegESEon samples of different granularities in NED-wiki

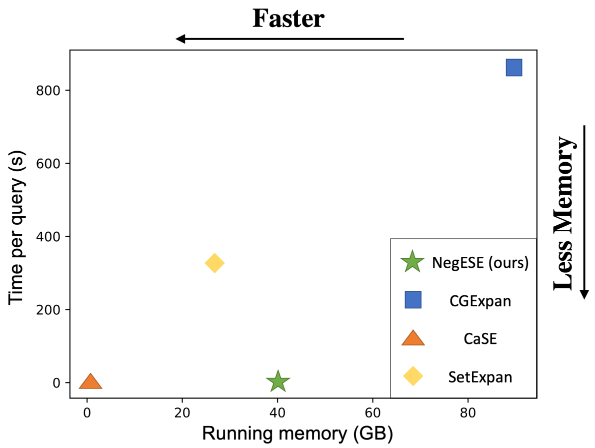

Figure 4: Time the memory efficiency of NegESE and state-of-the-art methods

$|\mathcal{A}_{neg}| = 0$, the samples express that no entities are disliked, this can also be written as $|\mathcal{A}_{neg}| = +\infty$ which means the disliked entities satisfy infinite requirements.

We conduct experiments on samples of different granularities (i.e. different $(|\mathcal{A}_{pos}|, |\mathcal{A}_{neg}|)$) from NED-wiki. We spot that: (1) Given $|\mathcal{A}_{neg}| = 0$ (The first 3 rows in Table 7), the performance drops consistently when adding more attributes to $\mathcal{A}_{pos}$. This is reasonable since the size of $\mathcal{P}$ shrinks if we select more positive attributes, which makes the entities we like hard to be find from the candidate entity set. (2) Given $|\mathcal{A}_{pos}| = 0$ (The last 3 rows in Table 7), the performance improves when we use more negative attributes, since more negative attributes make $\mathcal{N}$ to be smaller. As a result, the corresponding $\mathcal{P}$ becomes larger and makes the expansion process to be easier.

Efficiency Analysis. We analyze the time and space efficiency of our NegESE framework and other ESE methods on NED-wiki test split, the result is shown in Figure 4. We can observe that CGExpan costs much more time and running memory to achieve a MAP@10 of 37.58% which is still lower than NegESE. NegESE take the shortest time and not much running memory to get better expansion performance compared to existing methods.

5.6 Case Study

We show a real expansion result generated by NegESE on NED-wiki in Table 6. Entities marked in red color are entities in the negative candidate entity set $\mathcal{N}$, other entities are correctly recalled and belong to the $\mathcal{P}$ of the corresponding sample. This result demonstrates that our method can effectively prevent the disliked entities to be recalled.

6 Conclusion

To solve the entity set expansion involving both "like" and "dislike" requirements, we propose to express the target negative-aware semantic class with both positive and negative seed entities and design a learnable negative-aware entity set expansion framework, NegESE to solve this problem. Since existing ESE datasets only contain positive seed entities, we also construct the NED dataset to facilitate the study of the negative-aware entity set expansion task. Experiments demonstrate that the proposed method have the ability to utilize the negative seed entities to improve the set expansion performance and mitigate the negative candidate entity intrusion problem.

Acknowledgements

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Zhe Chen, Michael Cafarella, and H. V. Jagadish. 2016. Long-tail vocabulary dictionary extraction from the web. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, page 625–634, New York, NY, USA. Association for Computing Machinery.

James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, volume 6, pages 172–180. Citeseer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Sonal Gupta and Christopher D Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 98–108.

Yeye He and Dong Xin. 2011. Seisa: set expansion by iterative similarity aggregation. In *Proceedings of the 20th international conference on World wide web*, pages 427–436.

Jiaxin Huang, Yiqing Xie, Yu Meng, Jiaming Shen, Yunyi Zhang, and Jiawei Han. 2020. Guiding corpus-based set expansion by auxiliary sets generation and co-expansion. In *Proceedings of The Web Conference 2020*, WWW '20, page 2188–2198, New York, NY, USA. Association for Computing Machinery.

Prateek Jindal and Dan Roth. 2011. Learning from negative examples in set-expansion. In *2011 IEEE 11th International Conference on Data Mining*, pages 1110–1115. IEEE.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, page 94–100. AAAI Press.

Jonathan Mamou, Oren Pereg, Moshe Wasserblat, Ido Dagan, Yoav Goldberg, Alon Eirew, Yael Green, Shira Guskin, Peter Izsak, and Daniel Korat. 2018. Setexpander: End-to-end term set expansion based on multi-context term embeddings. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 58–62.

Jiaming Shen, Wenda Qiu, Jingbo Shang, Michelle Vanni, Xiang Ren, and Jiawei Han. 2020. SynSetExpan: An Iterative Framework for Joint Entity Set Expansion and Synonym Discovery. page 16.

Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017a. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 288–304. Springer.

Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017b. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *ECML/PKDD*.

Bei Shi, Zhengzhong Zhang, Le Sun, and Xianpei Han. 2014. A probabilistic co-bootstrapping method for entity set expansion.

Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707.

Richard C. Wang, Nico Schlaefer, William W. Cohen, and Eric Nyberg. 2008. Automatic set expansion for list question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, page 947–954, USA. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 1271–1279, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Puxuan Yu, Zhiqi Huang, Razieh Rahimi, and James Allan. 2019a. 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*, pages 1153–1156.

Puxuan Yu, Zhiqi Huang, Razieh Rahimi, and James Allan. 2019b. 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*, pages 1153–1156. ACM.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. 2018. Deep Sets.

Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020a. Empower entity set expansion via language model probing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8151–8160, Online. Association for Computational Linguistics.

Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020b. Empower Entity Set Expansion via Language Model Probing.

Andrew Zupon, Maria Alexeeva, Marco Valenzuela-Escárcega, Ajay Nagesh, and Mihai Surdeanu. 2019. Lightly-supervised representation learning with global interpretability. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 18–28, Minneapolis, Minnesota. Association for Computational Linguistics.

Semantic Class	Size
Animal	155
Books	204
Buildings	217
Chemical Elements	118
Cities	379
Companies	341
Consumer Electronics	56
Countries and Dependencies	234
Diseases	111
Institutions	258
Locations	264
Language	190
Person	203
Provinces and States	179
Sports Leagues	141
TV Channels	144
Video Games	234
Universities	200
Harry Potter Characters	111

Table 8: Number of entities in each coarse-grained semantic class

A Example Appendix

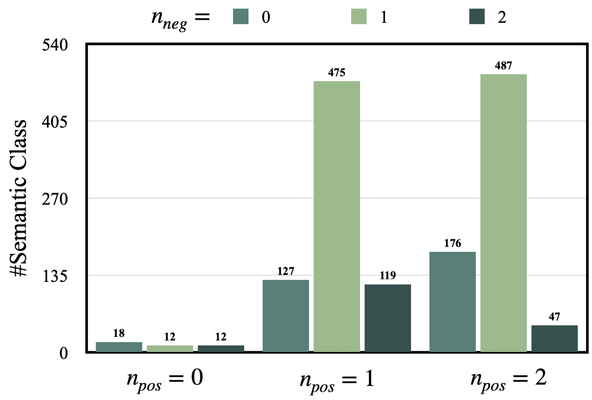

Figure 5: Number of semantic classes in different granularities in NED-wiki. n_{pos} and n_{neg} represents the number of positive attributes and negative attributes respectively.

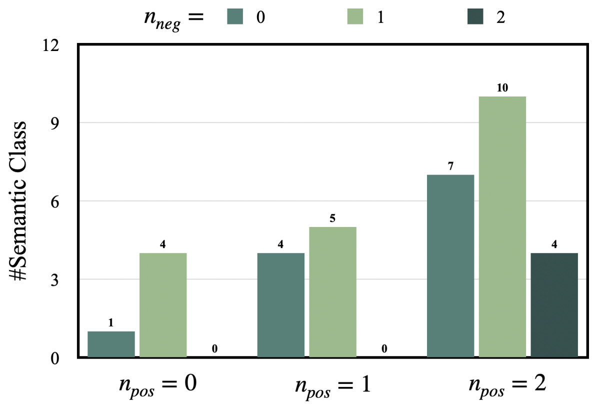

Figure 6: Number of semantic classes in different granularities in NED-hp. n_{pos} and n_{neg} represents the number of positive attributes and negative attributes respectively.

Semantic Class	**Attributes**
Animals	<Is_Vertebrate>, <Living_Environment>, <Diet>
Books	<Time_Published>, <Language>, <Genre>
Buildings	<Country>, <Function>, <Time_Built>, <In_Use>
Chemical Elements	<Element Group>, <Discovery_Country>, <Is_Radioactive>, <Symbol_Origin>
Cities	<Country>, <Is_Capital>, <Location>
Companies	<Business_Type>, <Country>, <Closed>, <(Once)_Listed>
Consumer Electronics	<Company>, <Type>, <Release_Time>
Countries and Dependencies	<Region>, <Is_UN_Member>, <Economic_Outlook>
Diseases	<Surgical>, <Chronic>, <Curable>, <Infectious>, <Part>
Institutions	<Type>, <Religion>, <Level>
Locations	<Region>, <Type>, <Climate>
Languages	<Language_Family>, <Is_Official_Language>, <In_Use>, <Origin>
Person	<Occupation>, <Born_Place>, <Gender>, <Birth_Time>
Provinces and States	<Country>, <Location>, <Economic_Outlook>
Sports Leagues	<Country>, <Sport>, <Closed>, <Time_Found>
TV Channels	<Type>, <Country>, <Closed>, <Language>
Video Games	<Publisher>, <Platforms>, <Time>, <Genre>
Universities	<Region>, <Public/Private>, <Comprehensive/Specialist>, <Time_Found>
Harry Potter Characters	<Gender>, <House>, <Occupation>, <Species>

Table 9: All coarse-grained semantic classes and corresponding attributes of each class in NED .

Parameter	**Value**
BERT	bert-base-cased
k_s	50
k_e	4000
non-linear activation	ReLU
dropout	0.2
H	768

Table 10: Parameters setting