Learning to Rank Entities for Set Expansion from Unstructured Data

Puxuan Yu, Razieh Rahimi, Zhiqi Huang, and James Allan
Center for Intelligent Information Retrieval
University of Massachusetts Amherst
Amherst, MA, 01003
{pxyu,rahimi,zhiqihuang,allan}@cs.umass.edu

ABSTRACT

We propose using learning-to-rank for entity set expansion (ESE) from unstructured data, the task of finding "sibling" entities within a corpus that are from the set characterized by a small set of seed entities. We present a two-channel neural re-ranking model, NESE, that jointly learns exact and semantic matching of entity contexts through entity interaction features. Although entity set expansion has drawn increasing attention in the IR and NLP communities for its various applications, the lack of massive annotated entity sets has hindered the development of neural approaches. We describe DBpedia-Sets, a toolkit that automatically extracts entity sets from a plain text collection, thus providing a large amount of distant supervision data for neural model training. Experiments on real datasets of different scales from different domains show that NESE outperforms state-of-the-art approaches in terms of precision and MAP. Furthermore, evaluation through human annotations shows that the knowledge learned from the training data is generalizable.

KEYWORDS

Set completion; Query by example; Neural networks

ACM Reference Format:
Puxuan Yu, Razieh Rahimi, Zhiqi Huang, and James Allan. 2020. Learning to Rank Entities for Set Expansion from Unstructured Data. In *2020 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '20), September 14–17, 2020, Virtual Event, Norway*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3409256.3409811

1 INTRODUCTION

Corpus-based entity set expansion refers to the task of finding all other entities in a given corpus that belong to the same semantic class as a few seed entities. For example, given the input seed set {Oslo, Amsterdam, Lisbon}, also referred to as a *query*, an ESE algorithm is expected to output other capitals in Europe that are mentioned in a given corpus. Set expansion is broadly useful for a number of downstream applications, such as question answering [34], taxonomy construction [32], relation extraction [17], information extraction [13] and query suggestion [3]. We believe ESE from plain text can provide guidance for knowledge base completion (KBC) [35]. The task is closely related to *example-based search* or *query-by-example* [43], which is a frontier of exploratory search and analysis [19]. It is worth noting that there is a further line of work where entity sets are expanded from structured data (e.g., Web lists and knowledge bases) [14, 25, 41]. Such approaches cannot be used for expansion of entities outside of knowledge bases.

Most corpus-based approaches [10, 20, 26, 27, 29, 31, 38, 39] are based on the assumption of distributional similarity [12], which, in the context of set expansion, can be interpreted in two ways: (1) *exact matching*: textual contexts (e.g., n-grams and skip-grams) are directly considered as features of entities; and (2) *semantic matching*: context information is used to train distributed representations of entities (embeddings) [8, 21–23]. Either exact or semantic matching can be adopted to expand entity sets, though they both have limits. The former finds sibling entities based on extraction of high-quality textual patterns. An entity needs to share exact textual patterns with at least one seed entity to be considered as an expansion candidate. On the other hand, in semantic matching models, entity embeddings generated by different language representation models tend to express different types of similarity, not reflecting only entity sibling relations.

An unsupervised approach CaSE combining exact and semantic matching techniques has shown significant improvements over individual methods [39]. The core intuition is to search for semantically related entities that frequently share important contexts with seed entities. We are interested in capturing such a hybrid process with a neural model. We have two motivations. First, as an unsupervised method, CaSE has made heuristic choices of parameters and conversion functions in the context matching algorithm. Second, we hypothesize that there exists a mapping from a general embedding space to an entity embedded space, where high similarity corresponds to close sibling relation.

We address three key challenges in designing a supervised neural model for the set expansion problem.

Data. There is no publicly available standard dataset of reasonable size for training and evaluation of ESE models. The issues with evaluation of current models include: insufficient number of testing entity sets and queries [15, 31, 38, 42], heavy focus on frequent entities [31, 38] and on entities from selected topics [15, 27, 31, 38, 39, 42]. We describe a toolkit, DBpedia-sets, to extract potential training and test entity sets from any free text corpus using the DBpedia knowledge base [1][1]. DBpedia-sets automatically annotates entity mentions in plain text with an entity linker [7] and groups entities from the same category together based on structural entity relations in the knowledge base.

ICTIR '20, September 14–17, 2020, Virtual Event, Norway

ACM ISBN 978-1-4503-8067-6/20/09.
https://doi.org/10.1145/3409256.3409811

[1] The toolkit and built datasets are publicly available at https://github.com/PxYu/NESE

Feature space complexity. Current unsupervised methods based on exact matching [27, 31, 39] adopt skip-grams as entity features, which makes a sparse and high-dimensional feature space. We revisit the *explicit vector space representation* [18], which means using unigrams in an entity's contexts as entity features. The significant advantage of unigrams over skip-grams for exact context matching is that the entity feature dimension is greatly reduced, while expansion effectiveness is not affected.

Generalizable embedding learning. For semantic matching, the essence is to learn effective and generalizable entity representations without resorting to entity knowledge outside a given corpus (e.g., encyclopedia). It is not feasible to train ESE-specific entity embeddings from scratch for lack of supervision data. Simply fine-tuning pre-trained embeddings cannot update representations of entities outside the training data. We propose using neural networks to learn a "projection" of a pre-trained entity embedding space into an optimal embedding space for the ESE task with limited annotated data. The intuition is that an ESE-specific semantic space better exhibits the sibling relation between entities.

We cast the ESE task as a learning-to-rank problem and propose a two-channel neural re-ranking ESE model, called NESE (**N**eural **E**ntity **S**et **E**xpansion). A semantic matching channel and an exact context matching channel are jointly trained to predict the probability of set membership of an entity given seed entities. Extensive experiments (§6) on three different corpora show that NESE achieves statistically significant improvements over baseline methods. NESE improves the expansion effectiveness in terms of MAP up to 18% over the state-of-the-art models.

2 RELATED WORK

Set Expansion from the Web (Online): Web-based methods for set expansion [5, 34] extract entities from documents retrieved by a search engine with respect to the query built from seed entities. Such methods impose considerable run-time overhead and they assume that top-ranked web pages contain other entities of the set, which is not necessarily true. Most studies, including this one, thus focus on expansion in the offline setting.

Set Expansion from Structured Data: The SEISA model [14] expands a seed set using bipartite graphs built from lists extracted from web pages and Web search query logs in an iterative procedure. The ESER model [41] and the NVSE model [25] both expand query entities on knowledge bases, based on assuming deficiency (e.g., incompleteness and noisiness) of knowledge graphs. Such methods have different applications from plain text based methods because they use structured data as the source of information.

Set Expansion from Unstructured Data: Earlier works [26, 29, 33] are usually based on co-occurrence frequencies of entities. Ghahramani and Heller [10] model set expansion as a Bayesian inference problem. SetExpan [31] is an iterative bootstrapping model where entities and their context features are arranged in a bipartite graph. Two types of features are adopted: skip-grams and coarse-grained entity types from Wikipedia. MCTS-PMSN [38] is another bootstrapping model that learns entity and pattern embeddings for pattern selection in a Monte Carlo tree search, but the learned embeddings are dependent on entity sets. The CaSE model [39] is a non-bootstrapping *one-time ranking* method, and has better scalability with large corpora. To score a candidate with respect to a query, CaSE first constructs a candidate pool, and then leverages candidate-query association strengths directly via embeddings and indirectly via skip-grams. CaSE improves the accuracy of set expansion by combining exact skip-gram matching with entity semantic matching without resorting to any hints of entity relations other than plain text. There are also works which focus on more specific areas in ESE, such as multi-faceted queries [27], semantic drift [15], and set name generation [42].

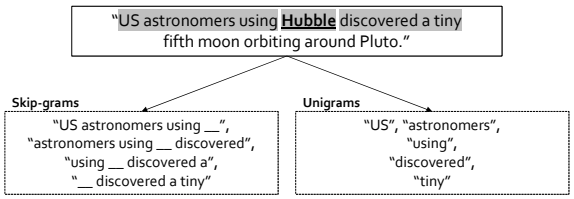

Figure 1: Skip-gram and unigram features extracted for entity "Hubble Space Telescope".

3 THE DBPEDIA-SETS TOOLKIT

We observed multiple problems in the evaluation datasets of existing works on the ESE task:

Human effort. Entity sets and evaluation queries are mainly constructed by human experts [27, 31, 38], an expensive and time-consuming task for large datasets, and one that requires human experts in different domains.

Corpus independence. Entity sets are collected independent of text corpora. Often a large portion of entities in a set are not present in a corpus, which makes the set not suitable for evaluation.

Bias. Entity sets are biased towards the most frequent entities in the corpus, or entities of specific topics. For example, Shen et al. [31] chose 20 queries from the 2,000 most frequent entities in each benchmark corpus to evaluate their ESE algorithm. However, expanding seeds from infrequent entity sets is the main motivation of set expansion from text corpora. The evaluation entity sets used by Rong et al. [27] are mainly related to geo-locations.

To address the above problems, we develop the DBpedia-Sets toolkit to extract entity sets suitable for training and testing of ESE models from a given corpus. DBpedia-Sets requires a plain text corpus and some statistical constraints on desired entity sets as inputs. The output is a series of entity sets that meet the constraints. Each output entity set is essentially a (sub)set of entities in a Wikipedia *category*[2], hence the data is of high quality.

DBpedia-Sets consists of three main steps. First, entity mentions within the text are identified. DBpedia Spotlight [7], a popular DBpedia-based entity linker, annotates the corpus by replacing entity mentions with the distinct surface names of entities to which they are linked. Second, all potential entity sets within the corpus are obtained. Queries written in the SPARQL Query Language [24] are submitted to request entity categories in the knowledge graph via the public SPARQL endpoint over the DBpedia dataset[3]. Each entity can be associated with one or more categories. The union of

[2] For example, https://en.wikipedia.org/wiki/Category:Winter_Olympic_sports
[3] http://dbpedia.org/sparql

all obtained categories for all entities in the corpus constitutes the potential entity sets.

Third, statistical-based filters are applied to select entity sets satisfying the specified constraints, which can be about (1) the upper and lower limits of the size of entity sets; and (2) coverage of set entities in the corpus, where a specified percentage of entities in a set should have corpus frequencies higher than a specified value. For all three corpora in our experiments (§6), the statistical-based filter "all entity sets containing 10 to 100 entities, where at least 90% of the entities appear at least 10 times in the corpus" is applied to extract entity sets. The size constraint is to ensure entity sets with large number of entities do not dominate the training of ESE models. The coverage constraint makes sure that contextual information about entities can be found in the corpus.

Note that the DBpedia knowledge base is only used to build training and evaluation data, and the learned models can be applied without requiring a knowledge base. The process of extracting entity mentions from the corpus does not necessarily require an entity linker, because noun phrases [30] are widely adopted to approximate entity mentions [31, 39]. The proposed sample collection procedure aims to acquire accurate labels from knowledge base to supervise training and evaluation of ESE models. The learned ESE model can also be used for expansion of entity seeds that are not available in a knowledge base, as shown in Section 6.3.

4 ENTITY CONTEXT FEATURE

Context-based ESE methods [27, 31, 38, 39] extract features from entities' contexts (sometimes referred to as "patterns") with the idea that entities sharing similar context features are more likely to be members of the same entity set. Rong et al. [27] proposed to use skip-grams as entity features for set expansion, where a skip-gram is defined as a short span of text around entity mention. They claimed that skip-grams impose strong positional constraints on the context and thus appropriate filtering and sampling based on these features can recover sibling relations more precisely. Subsequent ESE models naturally followed this approach [31, 38, 39]. For example, features in CaSE are extracted by sliding a 4-term window over the 6-term span centered at each entity mention such that four skip-gram features are extracted from each entity mention in the corpus. An example of extracted features is shown in Figure 1.

The alternative entity feature is the *explicit vector space representation* [18], that is, to simply use unigrams around entity mentions as features. This approach has the advantage of a lower-dimensional feature space, since there are far fewer distinct words than distinct skip-grams in a corpus. A comparison of the number of skip-gram and unigram features for different datasets is presented in Table 1.

Set expansion models based on skip-gram features are biased towards sets of highly frequent entities, because exact matching based on multi-term sequences is much more restrictive than that based on single terms. Using unigram context features increases the chance of entities being matched so that recall is improved. We show in Section 6.2 that using unigram features can significantly reduce the number of queries for which no correct entities are retrieved at top-100 position (recall@100=0). On the other hand, terms matched with skip-gram features usually have the same part-of-speech, which is particularly useful in the task of *term set expansion* [20] and building a thesaurus from plain text [11, 28]. However, ESE models can approximate entity mentions with noun-phrases. We demonstrate that there is no statistically significant difference in terms of MAP performance of CaSE with either skip-gram or unigram features. This finding shows that unigram features can be as effective as skip-gram features for the ESE task. Due to the aforementioned advantages of unigram features over skip-grams, we extract unigram features from entity contexts for our ESE model.

5 NEURAL SET EXPANSION MODEL

5.1 Task Formulation

Entity set $E = \{e_1, e_2, \cdots, e_{|E|}\}$ refers to a complete set of entities extracted from a corpus that can be categorized under one semantic class. An example in the AP89 corpus is {Lithuania, Norway, Estonia, Iceland, Sweden, Denmark, Finland, Latvia, Russia} from the entity set "Northern European countries". Query q refers to the initial set of n seed entities sampled from an entity set as input. Given a query q with n entities sampled from entity set E and a text corpus, the goal of an ESE model is to accurately retrieve other entities in the corpus that belong to E (i.e., $E-q$). Instead of considering all entities in the corpus as candidates, our proposed model re-ranks the top-100 results from an unsupervised candidate generator method.[4]

5.2 Input Representation

We encode each query or candidate entity with two features: a pre-trained entity embedding and a unigram feature vector. Entity embeddings are continuous vectors of length l_{eb} that embed all entities in a latent space. As discussed in Section 4, we extract unigram features from contexts of entity mentions in text and build a unigram feature vector for each entity occurring in the corpus. Unigram feature vectors are continuous vectors of length l_{uf}, equalling the size of unigram vocabulary. We choose a window of 6-term span centered at an entity mention as its context (Figure 1). Formally, for all in-corpus entities CE and the unigram vocabulary U, we calculate matrix $S^{|CE|\times|U|}$ in which S_{ij} corresponds the association strength between entity e_i and unigram u_j. The association strength can be estimated in different ways. Intuitively, S_{ij} should increase as the frequency of u_j in the context of e_i increases, and should decrease as the number of entities that co-occur with unigram u_j increases. For this purpose, we choose the *positive pointwise mutual information* (PPMI) metric [2, 6, 18]:

$$S_{ij} = \max(\text{PMI}(e_i, u_j), 0),$$
$$\text{PMI}(e, u) = \log \frac{P(e,u)}{P(e)P(u)} = \log \frac{\text{freq}(e,u)|\text{corpus}|}{\text{freq}(e)\text{freq}(u)}, \quad (1)$$

where $|\text{corpus}|$ is the total number of words in the corpus, $\text{freq}(e, u)$ is the frequency of u occurring in the context of e, and $\text{freq}(e)$ and $\text{freq}(u)$ are the corpus frequencies of e and u, respectively.

Padding and random permutation of inputs. We assume a maximum number of entities in queries, because in practical settings, it is very unlikely that users will provide numerous entity seeds to be completed. The maximum number is set to 5 in the experiments of this study, though our model can be trained for any maximum query length. Queries with less than 5 entities are first

[4] Pilot experiments regarding the first-step ranker are described in Section 6.1.

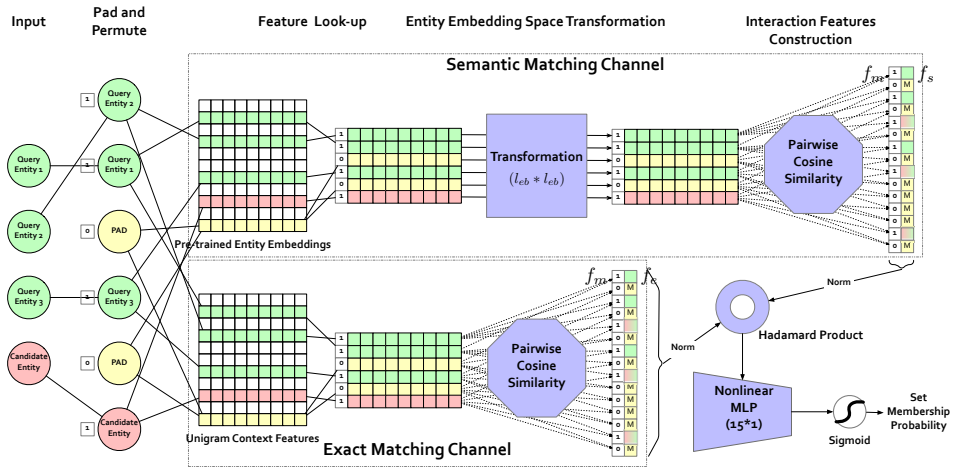

Figure 2: The NESE re-ranker that jointly learns semantic and exact context matching from data.

padded. Then, five query and possibly pad entities are randomly permuted. A candidate entity is then appended to the end.

5.3 NESE Architecture

The proposed NESE model consists of two components, one for matching entities based on their embeddings, and one based on their context features extracted from text. The two components are referred to as *semantic matching* and *exact matching* models respectively. The intuition in the design of both components is that only the similarity of a candidate with seed entities is not fully indicative of set membership. This is because entities in different entity sets have various degrees of closeness, which can be characterized by query-query interactions. Our model is thus designed to operate on both query-query and query-candidate similarities.

The **semantic matching model** expands queries by entities that are most semantically similar. We assume that there exists a linear mapping from the pre-trained entity embedding space to another embedding space, where the similarity of entities from the same semantic class is more apparent. To model linear mapping, we employ a learnable weight matrix $\mathbf{W} \in \mathbb{R}^{l_{\text{eb}} \times l_{\text{eb}}}$ and each entity embedding $\mathbf{x}$ is transformed to $\mathbf{Wx}$. Next, interaction matrix between any pairs of seed and candidate entities is obtained by calculating cosine similarities between transformed entity representations. The interaction matrix is then concatenated into one vector $\mathbf{f_s}$. A masking vector $\mathbf{f_m}$ is built by checking which input elements equal to the padding element, multiplying it by itself, and then flattening it. By taking the Hadamard product of $\mathbf{f_s}$ and $\mathbf{f_m}$, all non-valid similarity values due to the padding are set to zero in the final $\mathbf{f_s}$.

Exact matching model tries to match entities based on the similarity of their contexts in the given corpus. Unigram features are extracted from entities' contexts and constitute the input of this model. Similar to the semantic matching, an interaction matrix between seeds and the candidate entity is built by computing their pairwise cosine similarities, and the output is the masked similarity vector $\mathbf{f_e}$.

Either semantic or exact matching model can be independently applied for the ESE task by adding a classification component on top of their outputs such as a linear layer. However, to fully utilize all features, we combine the two components and jointly train them for the ESE task. Specifically, masked pairwise similarity vectors from two components, $\mathbf{f_s}$ and $\mathbf{f_e}$, are first min-max normalized and are then combined using the Hadamard product. A multi-layer perceptron (MLP) is adopted to predict the set-membership probability of a candidate entity given the combined features. We show that the exact matching model can help the training of semantic matching model which has many more parameters to be learned.

An ESE model should be invariant under permutation of entities in the query. The only order-sensitive part of the NESE model is the vector of pairwise similarities. Theoretically, using a summation or an average operator instead of pairwise similarity makes the model fully invariant to input permutations [40]. However, an inner representation based on summed or averaged query entity representations does not fully capture *interactions* between entities.

5.4 Model Training

Figure 2 illustrates how to calculate the set-membership probability z_i of one candidate c_i given query q. Repeating the process for all candidates $\boldsymbol{c}$ for q, a ranked list of candidate entities is obtained. We have ground-truth labels (relevance judgement) by considering candidate entities that belong to the entity set that query q is sampled from as relevant and others as non-relevant. For training of NESE, we compare the obtained ranking of candidates with ground-truth labels using the ListNet [4] loss function. This loss function is based on estimating a probability distribution for a list of scored entities, indicating the probability of different rankings. The probability distribution can be estimated using permutation or top-1 probabilities. Because of the computational complexity of permutation probabilities, we use top-1 probabilities following the original model.

Table 1: Statistics of training and test data.

Datasets	AP89	WaPo	Wiki
# sentences	1.60M	22.6M	43.3M
# skip-gram features	4.1M	14.2M	57.1M
# unigram features	0.12M	0.57M	2.20M
# kept unigram features	16,225	42,813	86,471
# entity sets	66	121	200
avg. # entities per set	18.3	18.9	19.8
avg. # training queries per fold	8,730	14,430	23,946
# test queries	1,980	3,630	6,000

Table 2: Recall@100 of candidate generation methods.

Dataset	AP89			WaPo			Wiki		
Query length	3	4	5	3	4	5	3	4	5
GloVe	.283	.318	.345	.386	.422	.449	.505	.534	.560
BERT	**.611**	**.618**	**.631**	**.510**	**.516**	**.532**	.444	.463	.472
CaSE-skip	.421	.432	.445	.495	.505	.517	.551	.568	.583
CaSE-uni	.418	.428	.444	.476	.484	.498	**.569**	**.583**	**.597**

6 EXPERIMENTS

6.1 Experimental Design

Benchmark datasets. We use three text collections for evaluation: (1) **AP89** is a TREC collection of 84,678 news articles published by the Associated Press in 1989. (2) **WaPo** is the Washington Post Corpus by TREC, which contains 608,180 news articles and blog posts from January 2012 to August 2017. (3) **Wiki** is the English Wikipedia dump of June 2019, where non-article pages such as "list of", redirect, and disambiguation pages are removed. The DBpedia-sets toolkit is applied on each corpus to extract and select entity sets. The toolkit generates 4,127 sets for Wiki corpus, and we randomly sampled 200 of them for efficiency. All generated entity sets from AP89 and WaPo are kept.

From each entity set E with m entities, some training and test queries are sampled. For each query length $n \in \{3, 4, 5\}$, we randomly sample n entities from the set E to form a training query for $\min\{\binom{m}{n}, 100\}$ times. The reason for setting an upper limit on the number of samples per set is to prevent large entity sets from dominating the training data. Entities in E which are not sampled for a query are labeled as 1 in training samples. For negative samples, we randomly choose 40% of incorrect entities retrieved by the initial ESE model that generates candidate entities. The reason for not using all negative samples is discussed in Section 6.4. Similarly, for each query length $n \in \{3, 4, 5\}$, $\min\{\binom{m}{n}, 10\}$ test queries are sampled from the set E. Fewer number of test queries are sampled from each entity set due to the quite long run-time of some baseline models. It also prevents evaluation metrics averaged on the query level from being biased towards large entity sets.

For each corpus, obtained entity sets are randomly divided into 5 folds. In each run, training queries in three folds are used to train the model, test queries in a fourth fold for monitoring training and early stopping, and test queries in the final fold for testing the performance of the learned model. Note that this means that a given entity set E is never used for both training and testing. Five runs generate results for test queries sampled from all entity sets. Test queries for baselines that do not need training are the combination of test queries sampled from all folds. Statistics of datasets is reported in Table 1.

Baselines. We divide comparable corpus-based ESE models into three categories, and select the most effective methods in each category as baselines for evaluation. **1) Semantic matching approaches.** We acquire **GloVe** and **BERT** embeddings of all entities in each corpus annotated by DBpedia-Sets, where each entity is regarded as a word. GloVe[5] embeddings are trained by setting the window size and maximum number of training iterations to 10 and 30, respectively. The output embedding vectors are of dimension 100. A BERT embedding of each entity is obtained by averaging the contextualized representations of all its occurrences in a corpus. We use "bert-base-uncased" version of pre-trained BERT from the Transformers library [36]. BERT embeddings are 768-dimensional. A k-NN classifier based on cosine similarity of candidates and average of seeds' embeddings is then used to rank the candidates. **2) Exact matching approach.** We use the released implementation of SetExpan [31] for evaluation[6]. **3) Hybrid approach.** We use the released implementation of CaSE for evaluation[7]. The original work uses skip-gram features for exact matching. Here, we experiment with both skip-gram features (**CaSE-skip**) and unigram features (**CaSE-uni**), following the discussion in Section 4. For the distributed entity representations, we adopt locally trained GloVe embeddings. **4) Feature-based learning-to-rank.** We also report the results of AdaRank [37] for the ESE task where the input consists of human-engineered features. To have a fair comparison, we build features by computing cosine similarities between each pair of entities in a given query and candidate sample, where entities are represented by pre-trained embeddings (BERT on AP89 and WaPo, GloVe on Wiki), unigram context feature vectors, or both. We respectively refer to these models as **AdaRank-emb**, **AdaRank-uni**, and **AdaRank-cmb**. We train separate models for each length of input query as the model does not support inputs of various lengths. AdaRank is a listwise learning-to-rank framework that is capable of directly optimizing IR metrics. In this re-ranking task, we optimize MAP@100.

Candidate generation. The criteria for selecting candidate generation model among baseline methods include high recall value and run-time efficiency. SetExpan is excluded from consideration because of slow runtime. Recall@100 values of other baseline methods for test queries on different datasets are listed in Table 2. Given the results, we adopt BERT as candidate generator for AP89 and WaPo, and CaSE-uni for Wiki.

Evaluation metrics. The baseline and proposed ESE models retrieve a ranked list of entities with respect to a query. The queries in test data are evaluated using Mean Average Precision at top 100 entities (MAP@100), and precision at top-20 entities (P@20). Statistical significant tests are performed using the two-tailed paired t-test at the 0.05 level.

[5] https://github.com/stanfordnlp/GloVe [6] https://github.com/mickeystroller/SetExpan
[7] https://github.com/PxYu/entity-expansion

Table 3: Performance of different ESE models on different corpora. Strongest baselines on each dataset are underlined (AP89, WaPo: BERT, Wiki: CaSE-uni). †: statistically significant (95% confidence interval) improvements compared to the strongest baseline. Δ: NESE's relative improvement over the strongest baseline.

Dataset	**AP89** (34.8M tokens)						**WaPo** (395M tokens)						**Wiki** (928M tokens)					
Metrics	MAP@100			P@20			MAP@100			P@20			MAP@100			P@20		
Query length	3	4	5	3	4	5	3	4	5	3	4	5	3	4	5	3	4	5
GloVe	.110	.123	.128	.101	.108	.104	.167	.179	.183	.161	.160	.157	.231	.250	.259	.214	.216	.210
BERT	.262	.270	.267	.227	.212	.208	.235	.234	.239	.211	.203	.196	.180	.186	.187	.174	.169	.161
SetExpan	.154	.153	.153	.120	.121	.119	.171	.172	.165	.172	.168	.162	.220	.217	.217	.201	. 195	.188
CaSE-skip	.174	.183	.183	.148	.147	.143	.206	.196	.196	.205	.188	.184	.249	.248	.248	.227	.216	.205
CaSE-uni	.168	.181	.179	.152	.153	.146	.204	.195	.195	.200	.185	.180	.254	.253	.254	.231	.219	.208
AdaRank-uni	.223	.245	.256	.217	$.220^{\dagger}$	$.217^{\dagger}$	.238	.240	$.247^{\dagger}$	$.226^{\dagger}$	$.223^{\dagger}$	$.218^{\dagger}$	.213	$.264^{\dagger}$	$.267^{\dagger}$	.206	$.237^{\dagger}$	$.230^{\dagger}$
AdaRank-emb	.227	.245	.259	.217	.210	.203	.235	.232	.238	.211	.202	.197	.260	$.273^{\dagger}$	$.282^{\dagger}$	$.239^{\dagger}$	$.237^{\dagger}$	$.232^{\dagger}$
AdaRank-cmb	.227	.246	.256	.218	$.220^{\dagger}$	.215	.238	$.242^{\dagger}$	$.247^{\dagger}$	$.226^{\dagger}$	$.225^{\dagger}$	$.219^{\dagger}$	.259	$.270^{\dagger}$	$.280^{\dagger}$	$.239^{\dagger}$	$.236^{\dagger}$	$.230^{\dagger}$
NESE-uni	.241	.252	.256	.230	$.227^{\dagger}$	$.220^{\dagger}$	.242	$.246^{\dagger}$	$.248^{\dagger}$	$.232^{\dagger}$	$.228^{\dagger}$	$.218^{\dagger}$	.249	$.264^{\dagger}$	$.268^{\dagger}$	$.240^{\dagger}$	$.237^{\dagger}$	$.231^{\dagger}$
NESE-emb-nt	.236	.250	.261	.207	.200	.201	.225	.230	.236	.202	.197	.193	.261	$.273^{\dagger}$	$.281^{\dagger}$	$.239^{\dagger}$	$.238^{\dagger}$	$.230^{\dagger}$
NESE-emb	.206	.206	.212	.192	.182	.178	.217	.217	.222	.201	.196	.192	.217	.228	.235	.213	.210	.203
NESE-nt	.244	.253	$.277^{\dagger}$	.231	.226	$.224^{\dagger}$	$.246^{\dagger}$	$.248^{\dagger}$	$.266^{\dagger}$	$.234^{\dagger}$	$.229^{\dagger}$	$.224^{\dagger}$	.260	$.270^{\dagger}$	$.281^{\dagger}$	$.239^{\dagger}$	$.240^{\dagger}$	$.232^{\dagger}$
NESE	$\mathbf{.273}^{\dagger}$	$\mathbf{.283}^{\dagger}$	$\mathbf{.291}^{\dagger}$	$\mathbf{.240}^{\dagger}$	$\mathbf{.237}^{\dagger}$	$\mathbf{.231}^{\dagger}$	$\mathbf{.264}^{\dagger}$	$\mathbf{.268}^{\dagger}$	$\mathbf{.282}^{\dagger}$	$\mathbf{.253}^{\dagger}$	$\mathbf{.247}^{\dagger}$	$\mathbf{.240}^{\dagger}$	$\mathbf{.272}^{\dagger}$	$\mathbf{.288}^{\dagger}$	$\mathbf{.293}^{\dagger}$	$\mathbf{.252}^{\dagger}$	$\mathbf{.246}^{\dagger}$	$\mathbf{.239}^{\dagger}$
Δ	+4.2%	+4.8%	+9.0%	+5.7%	+11.8%	+11.1%	+12.3%	+14.5%	+18.0%	+19.9%	+21.7%	+22.4%	+7.1%	+12.8%	+15.4%	+9.1%	+ 12.3%	+14.9%

NESE settings. We removed unigrams that co-occur with less than 5 entities when building the unigram context features. This greatly improves storage and run-time efficiency without compromising performance. NESE is implemented with the PyTorch Framework. On AP89 and WaPo, we use BERT as pre-trained embeddings ($l_{\text{eb}} = 768$), and on Wiki we use GloVe as pre-trained embeddings ($l_{\text{eb}} = 100$). Two hidden layers (6 and 3 nodes) are applied with 20% probability of dropout in the prediction MLP. Stochastic optimization method Adam [16] is applied with learning rate 0.001 for mini-batch style training. Batch size is set to 64. The maximum number of training epochs is set to 20.

Ablation study. We perform an ablation study by leaving out one of the main components of the NESE model at each time. The obtained variants of the model are (1) **NESE-uni**: the NESE model using only the exact matching component based on unigram context features; (2) **NESE-emb**: the NESE model using only the semantic matching component based on linearly transformed entity embeddings; (3) **NESE-emb-nt**: the NESE-emb variant minus the linear transformation module; and (4) **NESE-nt**: the full NESE model without the linear transformation module.

6.2 Results and Analysis

Table 3 summarizes the performance of baseline and proposed models for the ESE task on each dataset using different evaluation metrics. Following, we discuss the reported results in details.

***k*-NN in embedding space.** The performance of GloVe improves as the corpus size grows. This behavior is expected because the larger the corpus, the more reliable the estimation of entity embeddings by GloVe. In contrast, the expansion performance based on BERT embeddings decreases as the size of text corpus increases, which is consistent with previously reported results [39]. As BERT is trained on massive text corpus, it stores large amount of world knowledge such that it compliments the lack of entity contextual information on smaller corpora. On the other hand, the Wiki corpus covers multiple domains and by averaging different representations of an entity from its different contexts, we lose contextual entity representations especially for ambiguous entities. This can cause the low performance of BERT on the Wiki corpus.

Exact matching models. SetExpan, AdaRank-uni, and NESE-uni expand seed entities based on exact matching of entities' textual context. Comparing their results in Table 3 shows the latter two significantly outperform SetExpan which is a strong unsupervised model. NESE-uni performs better than AdaRank-uni especially for queries of length 3 which are harder for every model to expand.

Skip-gram v.s. unigram context features. Results of CaSE-uni and CaSE-skip show that there is no statistically significant difference between the performance of using unigram or skip-gram features based on any evaluation metric. However, by using unigram instead of skip-gram as entity context feature, the percentage of test queries with zero recall@100 decreases from 6.76% to 5.19% on AP89, from 0.94% to 0.03% on WaPo, and from 2.13% to 0.82% on Wiki. This observation shows that unigram features are more suitable for expansion of queries with infrequent entities.

Analysis of the NESE model. As shown in Table 3, NESE constantly outperforms strong baselines over all corpora in terms of all evaluation metrics. In terms of MAP@100 (P@20), we obtained up to 9.0%, 18.0%, and 15.4% (11.1%, 22.4%, and 14.9%) improvements over the strongest baseline on AP89, WaPo and Wiki, respectively.

Comparing the results of NESE-emb-nt and NESE-emb models, we can observe that the linear transformation in the former overfits the training data and does not generalize as well as NESE-emb-nt. However, when we combine semantic matching with exact context matching, the model with linear transformation (NESE) generalizes better. Therefore, we can infer the following: (1) there does exist a linear mapping from a general entity embedding space to one that better exhibits the sibling relation among entities, given that

Table 4: MAP@30 for the human evaluation experiment.

Sets	NBA teams		TV channels		European capitals	
Query	q1	q2	q3	q4	q5	q6
GloVe	.625	.673	.059	.125	.050	.313
CaSE-uni	.656	.647	.178	.254	.524	.454
NESE	**.733**	**.733**	**.254**	**.313**	**.551**	**.524**

NESE shows statistically significant improvements over NESE-nt consistently; and (2) the exact matching channel plays the role of regularization and restricts the learning process of transformation matrix $\mathbf{W}$ to a more generalizable direction.

We also examine two pairs of models [AdaRank-uni & NESE-uni] and [AdaRank-emb & NESE-emb-nt]. In each pair, two models take the same inputs but are trained with different algorithms. NESE-based methods consistently perform on par or even better across different query lengths, with the additional benefit of handling variable-length queries compared to AdaRank-based models. In particular, NESE-uni significantly outperforms AdaRank-uni on short queries. We believe that this is because NESE-uni is trained using queries of different lengths which results in more training samples than those for AdaRank-uni.

Effect of query length. Intuitively, longer queries lead to more accurate ranked lists because of less ambiguous input. In terms of MAP, the performance of embedding-based models (GloVe and BERT) and supervised models (AdaRank and NESE) improve as query gets longer, while the performance of context matching models (SetExpan and CaSE) stays stable. Therefore, NESE aligns better with intuition and achieves higher relative improvements over baselines on longer queries.

Finally, we conclude that the reasons for the superior performance of NESE are: (1) learning a transformation layer that maps pre-trained GloVe entity embeddings to another space for ESE; (2) using unigram context features which allows better generalization of the transformation layer; (3) diversified matching patterns from variable-length queries ensures stronger generalization; and (4) DBpedia-sets generates high-quality supervision.

6.3 Ranking Noun Phrases

We also study how the trained model generalizes to entities outside a knowledge base. AutoPhrase [30] is first used to extract noun phrases from the Wiki corpus as an approximation of entity mentions. Since noun phrases have pre-trained embeddings and unigram features from their mentions in text similar to entity mentions obtained by a knowledge base, the trained NESE can be directly applied to rank noun phrases given noun phrase queries. Noun phrases are hard for automatic evaluation, because an entity can be expressed in different noun phrase forms. Therefore, we perform human evaluation in this experiment.

We first select three concept sets that are not in the training data of Wiki. The sets we choose are NBA teams, TV channels and European capitals. Two queries are formulated in each concept set. We gather the top-30 retrieved noun phrases for each query from GloVe, CaSE-uni and NESE. We present the sample queries to three volunteers, who are familiar with those three topics, and let them judge if the retrieved noun phrases belong to the topic. To avoid confusion, we also present the topic names, and allow volunteers to look for external information from the Web. The Fleiss' multi-rater agreement measure [9] κ are 0.96, 0.60 and 0.71 for the three sets, respectively. We regard noun phrases with two or three votes as correct answers and measure MAP@30. Finally, the results are shown in Table 4. This shows that NESE generalizes to entities out of KB and yields competitive performance.

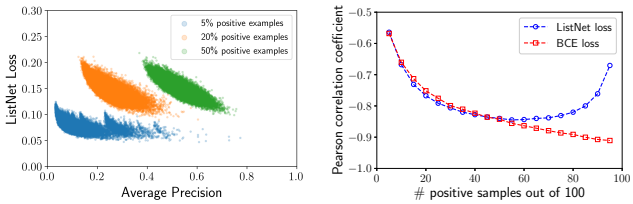

Figure 3: Left: distribution of ListNet loss and AP under different percentages of positive samples. Right: PCC between different losses and AP varies with the percentage of positive samples.

6.4 Impact of Unbalanced Training Set

In theory, ranking loss and ranking metrics should be negatively correlated. However, in our experiments, we sometimes observe contradictory phenomena, which brings unwanted randomness to model training. One challenging property of our data is that the ratio of relevant entities to the number of candidate entities is small. Based on this observation, we hypothesize that the correlation of listwise ranking loss to AP is directly influenced by the ratio of positive and negative samples in ranked lists. To validate our hypothesis, we conduct the following simulation experiment. With a truth list of l ones and $(100 - l)$ zeros, 100 real numbers in the $[0, 1]$ interval are randomly generated as probabilities of relevance, and the ranking loss and AP of this prediction list are calculated. By repeating this step 50,000 times, we are able to acquire a statistically significant relationship between AP and ranking loss by calculating the Pearson correlation coefficient (PCC). The above two steps are collectively referred to as one "run". We sweep l, the number of correct entities in the 100 ground-truth data for each query, from 5 to 95 in increments of 5, and execute one run for each l value. Therefore, we get the PCC of ranking loss and ranking performance under different ratios of positive and negative samples.

We also experiment with weighted binary cross entropy loss (BCE loss), which is often applied when ranking with binary relevance judgments is considered as binary classification:

$$\mathcal{L}(y^{(q)}, z^{(q)}; \theta) = -\sum_i \left(w_p y_i^{(q)} \log z_i^{(q)} + (1 - y_i^{(q)}) \log(1 - z_i^{(q)}) \right),$$

where w_p is the *positive weight*, making it possible to trade off recall and precision by adding weights to positive examples. We set w_p empirically to the reciprocal of the positive-negative ratio, which balances the number of positive and negative samples in the data. The experimental results are shown in Figure 3. PCC with larger *absolute* value indicates stronger correlation.

We conclude from the results that ListNet loss and weighted BCE loss have a strong correlation with AP when the percentage of positive samples in candidate entities is between 20% and 80%. The two losses behave similarly when the candidate list is dominated by negative samples and very differently when the candidate list is dominated by positive examples. After verifying our hypothesis, we chose to randomly discard a subset of negative samples for each query in the training data, so that the proportion of positive samples in the data lies in the desired [20%,80%] interval. This sampling of negative candidate entities is performed when training the AdaRank and the NESE model, but not during inference time.

7 CONCLUSION AND FUTURE WORK

In this study, we consider the ESE task as a listwise learning to rank retrieval problem, and propose a two-channel neural re-ranking model, NESE. The semantic and exact matching channels generate pairwise similarities of entity embeddings using learned mapping and explicit entity representations, respectively. NESE is trained and evaluated with queries sampled from entities set obtained by DBpedia-Sets toolkit from three corpora. Extensive experiments show that NESE achieves statistically significant improvement over state-of-the-art baseline methods. Human annotation experiment confirms that NESE generalizes to entities outside knowledge base.

For future work we would like to perform sentence selection for generating query-aware BERT representation for entities.

ACKNOWLEDGMENT

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #IIS-1813662, and in part by NSF grant #IIS-1617408. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor. We thank Florian Gwechenberger for discussions about the use of BERT as a candidate generation method. The first author also thanks Zhipeng Tang for discussions about model efficiency.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.
[2] John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods* 39, 3 (2007), 510–526.
[3] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *SIGKDD*. ACM, 875–883.
[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *ICML*. ACM, 129–136.
[5] Zhe Chen, Michael Cafarella, and HV Jagadish. 2016. Long-tail vocabulary dictionary extraction from the web. In *WSDM*. ACM, 625–634.
[6] Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics* 16, 1 (1990), 22–29.
[7] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N. Mendes. 2013. Improving Efficiency and Accuracy in Multilingual Entity Extraction. In *I-Semantics*.
[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.
[9] Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76, 5 (1971), 378.
[10] Zoubin Ghahramani and Katherine A Heller. 2006. Bayesian sets. In *NIPS*. 435–442.
[11] James Gorman and James R Curran. 2006. Scaling distributional similarity to large corpora. In *COLING*. 361–368.
[12] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.
[13] Yifan He and Ralph Grishman. 2015. Ice: Rapid information extraction customization for nlp novices. In *NAACL*. 31–35.
[14] Yeye He and Dong Xin. 2011. Seisa: set expansion by iterative similarity aggregation. In *WWW*. ACM, 427–436.
[15] 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 *WWW*. 2188–2198.
[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[17] Joel Lang and James Henderson. 2013. Graph-Based seed set expansion for relation extraction using random walk hitting times. In *NAACL-HLT*.
[18] Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *CoNLL*. 171–180.
[19] Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. 2019. Example-based Search: a New Frontier for Exploratory Search. In *SIGIR*. ACM.
[20] 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 *COLING*. 58–62.
[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*. 3111–3119.
[22] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. 1532–1543.
[23] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL-HLT*. 2227–2237.
[24] Bastian Quilitz and Ulf Leser. 2008. Querying distributed RDF data sources with SPARQL. In *ESWC*. Springer, 524–538.
[25] Pushpendre Rastogi, Adam Poliak, Vince Lyzinski, and Benjamin Van Durme. 2019. Neural variational entity set expansion for automatically populated knowledge graphs. *Information Retrieval Journal* 22, 3-4 (2019), 232–255.
[26] Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semiautomatic semantic lexicon construction. In *COLING*. 1110–1116.
[27] Xin Rong, Zhe Chen, Qiaozhu Mei, and Eytan Adar. 2016. Egoset: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion. In *WSDM*. ACM, 645–654.
[28] Pavel Rychlỳ and Adam Kilgarriff. 2007. An efficient algorithm for building a distributional thesaurus (and other Sketch Engine developments). In *ACL*. 41–44.
[29] Luis Sarmento, Valentin Jijkuon, Maarten De Rijke, and Eugenio Oliveira. 2007. More like these: growing entity classes from seeds. In *CIKM*. ACM, 959–962.
[30] Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *TKDE* 30, 10 (2018), 1825–1837.
[31] Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. 2017. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *ECML-PKDD*.
[32] Jiaming Shen, Zeqiu Wu, Dongming Lei, Chao Zhang, Xiang Ren, Michelle T Vanni, Brian M Sadler, and Jiawei Han. 2018. HiExpan: Task-Guided Taxonomy Construction by Hierarchical Tree Expansion. In *SIGKDD*. ACM, 2180–2189.
[33] Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *EMNLP*. 214–221.
[34] Richard C Wang, Nico Schlaefer, William W Cohen, and Eric Nyberg. 2008. Automatic set expansion for list question answering. In *EMNLP*. 947–954.
[35] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge base completion via search-based question answering. In *WWW*. ACM, 515–526.
[36] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).
[37] Jun Xu and Hang Li. 2007. Adarank: a boosting algorithm for information retrieval. In *SIGIR*. ACM, 391–398.
[38] Lingyong Yan, Xianpei Han, Le Sun, and Ben He. 2019. Learning to Bootstrap for Entity Set Expansion. In *EMNLP-IJCNLP*. 292–301.
[39] Puxuan Yu, Zhiqi Huang, Razieh Rahimi, and James Allan. 2019. Corpus-based Set Expansion with Lexical Features and Distributed Representations. In *SIGIR*. 1153–1156.
[40] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan R Salakhutdinov, and Alexander J Smola. 2017. Deep sets. In *NIPS*. 3391–3401.
[41] Xiangling Zhang, Yueguo Chen, Jun Chen, Xiaoyong Du, Ke Wang, and Ji-Rong Wen. 2017. Entity set expansion via knowledge graphs. In *SIGIR*. ACM, 1101–1104.
[42] Yunyi Zhang, Jiaming Shen, Jingbo Shang, and Jiawei Han. 2020. Empower Entity Set Expansion via Language Model Probing. In *ACL*. 8151–8160.
[43] Mingzhu Zhu and Yi-Fang Brook Wu. 2014. Search by multiple examples. In *WSDM*. ACM, 667–672.