Entity Set Expansion on Social Media: A Study for Newly-presented Entity Classes

He Zhao[1], Chong Feng[1*], Zhunchen Luo[2], and Yuxia Pei[3]

[1]School of Computer Science and Technology,
Beijing Institute of Technology, Beijing 100081, China
zhaohe1995@outlook.com, fengchong@bit.edu.cn
[2]China Defense Science and Technology Information Center, Beijing 100142, China
zhunchenluo@gmail.com
[3]State Key Laboratory of Smart Manufacturing for Special Vechicles and Transmission System

Abstract. Online social media yields a large-scale corpora which is fairly informative and sometimes includes many up-to-date entities. The challenging task of expanding entity sets on social media text is to extract more unheard entities with several seeds already in hand. In this paper, we present a novel approach that is able to discover newly-presented objects by doing entity set expansion on social media. From an initial seed set, our method first explores the performance of embedding method to get semantic similarity feature when generating candidate lists, and detects features of connective patterns and prefix rules with specific social media nature. Then a rank model is learned by supervised algorithm to synthetically score each candidate terms on those features and finally give the final ranked set. The experimental results on Twitter text corpus show that our solution is able to achieve high precision on common class sets, and new class sets containing abundant informal and new entities that have not been mentioned in common articles.

Keywords: Social Media Mining; Information Extraction; Entity Set Expansion

1 Introduction

Have you ever expected to find out a TV series as hot as the one that you are obsessed with? Or have you ever wanted to know the newly released product by the brand of which you have been a great fan? As is well known that information in social media, where all these topics are included, always keeps abreast of the times. In order to extract entity instances of a latent semantic class, a set expansion method takes an initial set of seed examples as inputs, expands the set and then outputs more entity instances of that class. For example, one expecting to find all fast-food restaurants may give two or three well-known names like "KFC" and "Taco Bell" as seeds, based on the seed set and the text corpus it can discover other more restaurant names such as "Mcdonald's", "Pizza hut"

and "Dairy Queen", which distinctly belong to that set. The task has been well-studied on news corpus and web-based text in the past decades. However, in recent years the booming user quantity of social media platforms such as Twitter and Facebook provides tremendous amount of noisy and informal short text, simultaneously presents a new challenge of expanding entity sets by usual method and technology. Furthermore, due to the low-barrier of posting and the proliferation with mobile devices, open-domain social media text is often more up-to-date and inclusive than news articles. Conceivably, in informally written text there are many newly created entities that are in abbreviated form or did not exist before. As an example, "BK" usually means "BurgerKing" which is a fast-food chain and "FB" usually means "FaceBook" which is a social media website or an Internet company name. Identifying these entities is profoundly significant to the downstream applications, for instance, constructing entity repositories for social media and delivering precise searching results in social media-oriented query suggestion systems.

It is worth-noting that most semantic classes studied in previous work can be explicitly defined by several entity instances, like given "LALakers" and "Chicago Bulls" as seeds it can wisely infer the intention is to get other entities which are also "NBA teams". Whereas plenty of newly-presented entities have more than one semantic layers, so that we cannot give an absolute definition to a set of entities. A more specific example is, for "Temple Run" and "Subway Surfers", it can be concluded both of them are mobile games, iOS Applications or parkour games. There is no denying that the two instances should be contained into one set in some sense but the semantic class description may be diverse from different point of view. Therefore, unlike previous work, our research purpose is to aggregate the entities into one set just considering their implicit semantic concepts, while not imposing restrictions on the possible class names.

Corpus-based method has been applied to solve entity set expansion problem on general-purpose web source data and promising results have been reported in the past years. Wang and Cohen [1, 2, 3] develop the SEAL system on semi-structured documents from web. He and Xin [4] propose SEISA system using lists from the HTML web pages and the search query logs. A more latest work by Dalvi et al. [5] relies on clustering terms found in HTML tables and mainly focused on tabular data in web pages. Other than the web corpus, limited in 140 characters in most cases the text from social media does not comprise distinct structured tables or lists. And that due to the short text length, there's often not enough maximal context to build reliable wrappers for each seed example mentions. Hence obviously those off-the-shelf web-based methods show ineffective when applied to solve the entity set expansion task on social media corpus. Another prominent line of work using bootstrapping approaches like that by Thelen and Riloff [6], the ASIA system [7] and KnowItAll [8]. They extract context patterns to capture information about the behavior of a term or utilize hyponym patterns (e.g.,"fruits such as apples, bananas and oranges") to find instances based on the *"is-a"* relation. But with informal writing style and unconventional grammar, social media text often lacks signal words like "such

as" to make up hyponym patterns. And since the oral expression characteristics differ depending on individual users, context around an entity mention is rarely to be coincident to compose patterns.

In this paper, we present a novel entity set expansion method which especially orients to social media text and focus on some newfangled semantic classes that have not been sufficiently studied before. Our method uses a two-phrase architecture of extracting and ranking. In the first phase, to extract and generate candidate lists we adopt an embedding-based strategy to get scores on the feature of *semantic similarity*. Through the observation of various social media text, we find that many homogeneous entities are in appositive construction, connected by some conjunction words or symbols (e.g., ",", "and", "or", et al.). The intuition has been verified in the research of Widdows and Dorow [9] and been also depended on in the entity set expansion algorithm of Sarmento et al. [10]. Another observation is that in one single piece of message the entities, which synchronously have the prefixes "@" (to remind someone to notice) or "#" (to be a hashtag), are very likely in the same class. So here we propose the following hypotheses:

1. If a term is often connected with seed entities by the conjunction symbols that frequently occur between two seed entities in text corpus, the term is more likely to be a candidate instance of that class set.
2. If a term often occurs in the same piece of message with seed entities, both having "@" or "#" as prefix, the term is more likely to be a candidate instance of that class set.

Based on the two hypotheses, for each terms in candidate list, we respectively get scores on the feature of *connective patterns* and *prefix rules*. In the second phase, we present a rank model using a supervised learning method to combine the three features and provide the optimized coordination coefficient vector. We do experiments on outline Twitter text corpus and show that our approach can achieve high precision on some general entity classes and find multiple entities on some newfangled classes.

2 Related Work

Most previous work use two-phases method to expand sets, so as we do, which divides the task into two steps, extracting and ranking. In extracting, some common strategies like corpus-based, search-based, pattern-based and bootstrapping approaches are adopted extensively. And in ranking, most base on probability statistics, distributional similarity and some graph-based structures.

As has been mentioned in the introduction part, the SEAL system by Wang and Cohen [1, 2, 3] gets data source from web pages. In extracting phase they construct maximal context wrappers using given seeds and gather the candidate terms enclosed by these wrappers. Then in ranking phase they use a random walk process and rank by similarity metric on the basis of a graph containing seeds, wrappers and candidate terms.

The KnowItAll system by Etzioni et al. [8] also targets to web-scale data source. In extracting phase they use textual patterns like "colors such as pink, blue" as generic templates to generate candidate lists. And in ranking phase they adopt a bootstrapping method with the PMI measure computed by web search engines to estimate the likelihood of each candidate terms.

Similar to KnowItAll, the ASIA system by Wang and Cohen [4] is another pattern-based method. In extracting phase they also use hyponym patterns to get candidate terms and different from SEAL, they utilize a Bootstrapper to make the process iterative. Then in ranking phase supported by Random Walk with Restart on their graph it can determine the final ranking.

Talukdar et al. [11] propose a context pattern method that firstly find "trigger words" indicating the beginning of a pattern to extract candidate terms. And then they set evaluation mechanism to rank both the patterns and candidates.

Other existing work specially focuses on ranking the candidate sets. Bayesian Sets by Ghahramani and Heller [12], which is based on Bayesian inference, aims to estimate the probability that a candidate term belongs to a set by learning from a positive set P and an unlabeled candidate set U.

Sarmento et al. [10] propose a distributional similarity method according to a co-occurrence assumption that two elements consistently co-occur tent to be in similar semantic class. They find entity pairs by coordination structures and represent entities in vector space by encoding the co-occurrence frequency to compute the similarity with distance measure.

A PU-Learning method by Li et al. [13] transform set expansion into a two-class classification problem. A seed set is regarded as a set P of positive examples and candidate set is a set U containing hidden positive and negative cases. The task of filtering the candidate set turns to building a classifier to test if each candidate member is positive or not.

Beyond the work listed above, we want to mention another more recent work closed to set expansion on social media text. Qadir et al. [18] propose a novel semantic lexicon induction approach to learn new vocabulary from Twitter. Their method roots on the ability to extract and prioritize N-gram context patterns that are semantically related to the categories and is able to deal with multiword phrases of flexible term length. Starting with a few seed terms of a semantic category, they first explore the context around seeds a corpus, and identify context patterns relevant to that category, which then used to extract candidate terms. They experiment with three commonly discussed semantic categories, which are Food&Drinks, Games&Sports and Vehicles, and show good performance in these topics.

3 Entity Set Expansion

Our task can be defined as follows: given a text corpus T and a set S of seed entities of a hidden class C, we expand S by finding new terms t (we use the word "term" interchangeably with "entity" since textual terms are taken as candidate entities in expanded sets) from T, such that t is also semantically in the class C.

3.1 Entity Class

In this paper, we experimented with 24 semantic classes that are more concerned and widely discussed in social media platforms as is shown in Table 1. To demonstrate that our method can achieve high precision on some common classes and simultaneously find out entities of newfangled classes, we take both the *Common-class* sets containing more common entities with clearer semantics and *New-class* sets, in which most entities are proper and complex in semantic. In Table 1 we give 3 representative seed examples for each sets, but without any straightforward description about the class. And in the actual experiment, we construct initial candidate seed sets for each classes, which contains 9 seed entities on average.

Common-class	***New-class***
Sociology, Chemistry, Biology	Avatar, Transformers, The Dark Knight
lawyer, doctor, manager	iPhone, iPad, MacBook
hat, shirt, sweater	Microsoft, IBM, Apple
Canada, France, Morocco	keyboard, print, mouse
steak, pork, sausage	KFC, Taco Bell, Starbucks
fever, headache, migraine	Temple Run, Warfare, Dota
milk, juice, coffee	Amazon, eBay, Google
brother, sister, dad	Super Junior, BIGBANG, CNBlue
head, ear, mouth	mcflurry, frappe, smoothie
football, baseball, tennis	Twitter, Facebook, Instagram
piano,guitar,drum	Justin Bieber, Taylor Swift, Katy Perry
gym,store,library	Skype, FaceTime, kik

Table 1. Semantic classes and some representative seed examples

3.2 Corpus Description & Pre-processing

In this research, we use Twitter text to build our corpus dataset. From online Twitter platform, each tweet is a short message with a maximum length of 140 characters, in the nature of informal grammar, abbreviated expressions and misspellings. We collect 9,582,314 English tweets published in January, 2013, removing other redundant information and only keeping textual content left.

For pre-processing, we utilize Twitter NLP pipeline contributed by Ritter et al. [14] to chunk each tweet. For identified noun phrases, we delete the space between two words to integrate multiple words into a whole (e.g.,"Los Angeles Lakers" to "LosAngelesLakers"). The purpose of this is in vectorization procedure the phrases can be represented with single word embedding. Another important cause is that we observe and discover many Twitter users do like the integrated expression instead of typing more spaces, so it can exactly make the two forms refer to the same semantic instance.

3.3 Generating Candidate Sets

To extract candidate terms for each sets of seeds, we first train a word embedding model on the corpus T using Word2Vec, which is a text deep distributional representation approach proposed by Mikolov et al. [19]. Then the semantic similarity between two terms turns to be computed by cosine distance in the vector space. For each seed term $s \in S$, we select the top n semantically similar terms add to the candidate set D and remove the duplicates. After that for each candidate term $item \in D$, we calculate mean similarity with all seed terms in S as its score on the feature of *Semantic Similarity*.

$$SC_{sim}(item) = \frac{\sum_{s \in S} Similar(s, item)}{|S|} \tag{1}$$

3.4 Connective Patterns and Prefix Rules

By observing tweets text we conclude a set CS of 24 conjunction symbols that are generally used to connect two congeneric terms as is showed in Table 2. For each conjunction symbol $c \in CS$, and a pair of seed terms $(s_1, s_2) \in S$,

and	or	,	&	+	-
x	X	/	$\star$	>	<
|	vs	VS	.	:	//
·	\	\\	=	$\ll$	$\gg$

Table 2. Conjunction symbols set CS

we respectively construct two reversed strings "s_1 c s_2" and "s_2 c s_1" that we called *searching patterns* (e.g., "KFC & Starbucks"). We count frequencies of these two *searching patterns* in corpus text and sum them to get the frequency value $f_{(c,pair)}$ of c in regard to pair (s_1, s_2). The sum of the values computed by all possible seed pairs in S indicates the weight of that symbol c.

$$Weight(c) = \frac{\sum_{pair \in S} f_{(c,pair)}}{\sum_{c \in CS} \sum_{pair \in S} f_{(c,pair)}} \tag{2}$$

Then for each candidate term $item \in D$ and each seed term $s \in S$, again we use symbol c to construct two reversed strings "s c $item$" and "$item$ c s" that we called *matching patterns* (e.g. "KFC & Mcdonald's", "Mcdonald's & KFC"). Counting and summing the frequencies of the *matching patterns*, and multiplying the weight of c, we get the frequency value $f_{(item,c,s)}$of $item$ weighted by c. So the sum of the values computed by all symbols in CS indicates the score of $item$ in feature of *Connective Patterns*.

$$SC_{con}(item) = \sum_{c \in CS} \sum_{s \in S} Weight(c) \times f_{(item,c,s)} \tag{3}$$

We use similar process to get the score in feature of *Prefix Rules*. First we conclude a set PS containing 4 groups of different prefix collocations by "@" and "#" as is showed in Table 3. For each prefix collocation $(p_1, p_2) \in PS$, and a pair of seed terms $(s_1, s_2) \in S$, we respectively construct a group of *searching*

Prefix Collocations	Seed Terms	Candidate Terms
(@,@)	@s	@$item$
(@,#)	@s	#$item$
(#,@)	#s	@$item$
(#,#)	#s	#$item$

Table 3. Prefix Collocations set PS

patterns which are "$p_1 s_1$" and "$p_2 s_2$" (e.g., "@KFC" and "@Starbucks"). We count frequencies of the two patterns in corpus text and sum them to get the frequency value $f_{(p,s)}$ of prefix collocation p in regard to seed s. The sum of the values computed by all seeds in S indicates the weight of that collocation p.

$$Weight(p) = \frac{\sum_{s \in S} f_{(p,s)}}{\sum_{p \in PS} \sum_{s \in S} f_{(p,s)}} \tag{4}$$

Then for each candidate term $item \in D$ and each seed term $s \in S$, we use prefix collocation p to construct two strings "$p_1 item$" and "$p_2 s$" that are also named as *matching patterns* (e.g., "@KFC" and "@Mcdonald's"). Counting and summing the frequencies of the *matching patterns*, and multiplying the weight of p, we get the frequency value $f_{(p,item,s)}$ of $item$ weighted by p. So the sum of the values computed by all collocations in PS indicates the score of $item$ in feature of *Prefix Rules*.

$$SC_{pre}(item) = \sum_{p \in PS} \sum_{s \in S} Weight(p) \times f_{(p,item,s)} \tag{5}$$

3.5 Candidate Terms Ranking

With the scores of each candidate terms in feature of *Semantic Similarity*, *Connective Patterns* and *Prefix Rules*, now we can respectively rank the candidate set according to the three features, and get three ranking lists that are expressed as R_{sim}, R_{con} and R_{pre}. We present a synthetic ranking model R, learning a set of coordination coefficient vector $\boldsymbol{W} = (\alpha_1, \alpha_2, \alpha_3)$ to combine the each item's positions, which are $R_{sim}(item)$, $R_{con}(item)$ and $R_{pre}(item)$ in the three ranking lists.

$$R(\boldsymbol{W}) = \alpha_1 R_{sim}(item) + \alpha_2 R_{con}(item) + \alpha_3 R_{pre}(item) \tag{6}$$

First we give some notations and explanations using those defined in AdaRank by Xu and Li [20] as reference, as is showed in Table 4.

Here we use MAP(Mean Average Precision) as performance measure. For a given semantic class c_i, rank of candidate lists Y_i and a permutation π_i on candidate set D_i, average precision for c_i is defined as:

$$AvgP_i = \frac{\sum_{j=1}^{n(c_i)} P_i(j) \cdot y_{ij}}{\sum_{j=1}^{n(c_i)} y_{ij}}, \tag{7}$$

Notations	Explanations
$c_i \in C$	i^{th} hidden semantic class
$D_i = \{item_{i1}, \cdots, item_{i,n(c_i)}\}$	Candidate Set of Class c_i
$y_{ij} \in \{r_1, r_2, \cdots, r_l\}$	Rank of $item_{ij}$ w.r.t. c_i
$Y_i = \{y_{i1}, y_{i2}, \cdots, y_{i,n(c_i)}\}$	List of ranks for c_i
$S = \{(c_i, D_i, Y_i)\}_{i=1}^m$	Training Set
$\boldsymbol{W} = (\alpha_1, \alpha_2, \alpha_3)$	The coordination coefficient vector
$f, R(\boldsymbol{W}), R_{sim}, R_{con}, R_{pre} \in \mathcal{R}$	Ranking models
$\pi(c_i, D_i, f)$	Permutation for c_i, D_i, and f
$E(\pi(c_i, D_i, f), Y_i) \in [-1, +1]$	Performance measure function

Table 4. Notations and explainations

where y_{ij} takes on 1 and 0 as values, which presents $item_{ij}$ is positive or negative instance and $P_i(j)$ is defined as precision at the position of d_{ij}:

$$P_i(j) = \frac{\sum_{k:\pi_i(k) \leq \pi_i(j) y_{ij}}}{\pi_i(j)}, \tag{8}$$

where $\pi_i(j)$ presents the position of $item_{ij}$.

To learn coordinate coefficient for each rank model, to start with we assign the same value to α_1, α_2 and α_3. With a training set $S = \{(c_i, D_i, Y_i)\}_{i=1}^m$ as input, the algorithm takes the performance measure function E and runs at most T rounds. At each round, it creates a new rank model f by linearly combining the three feature rankers with $\boldsymbol{W}$. Each coefficient α_i will be updated to increase if the corresponding feature rank model outperforms model f, but decrease the opposite. Then we test if the rank model combined with updated $\boldsymbol{W}$ is better than f. If not, the algorithm reaches the maximum point at f, the loop ends and f is returned. The pseudocode of the whole process is shown in Algorithm 1, where Z_m is a standardization factor, to make $\sum \boldsymbol{W} = 1$:

$$Z_m = \sum_{k=1}^{3} \alpha_k \cdot exp\{\frac{\sum_{i=1}^m E(\pi(c_i, D_i, R_k), Y_i) - \sum_{i=1}^m E(\pi(c_i, D_i, f_t), Y_i)}{m}\} \tag{9}$$

4 Experiments

4.1 Baselines

To compare the performance of our set expansion algorithm with previous work, we use two pattern-based approaches. One is a **C**ontext **P**attern **I**nduction **M**ethod by Talukdar et al. [11], which we briefly name as **CPIM**, and another is a**T**witter-oriented **S**emantic **L**exicon **I**nduction **M**ethod by Qadir et al. [18], briefly named as **TSLIM**. Due to that many previous works target to structured text data source from web and utilize tables, lists and markup tags to solve the problem, obviously they are not fit for our Twitter text. Hence we choose approaches based on context patterns that don't like that impose restrictions on specific corpus. Besides, to the best of our knowledge very few works study entity

Algorithm 1 The rank model algorithm

Input:

Training set $S = \{(c_i, D_i, Y_i)\}_{i=1}^{m}$

Performance measure function $E(\pi(c_i, D_i, f), Y_i)$

Max iterations T

Initialize:

for α_k *in* $\boldsymbol{W}$: $\alpha_k = \frac{1}{|\boldsymbol{W}|}$

$R_1 = R_{sim}, R_2 = R_{con}, R_3 = R_{pre}$

Algorithm:

for $t = 1$ to T **do**
  $f_t = \sum_{k=1}^{3} \alpha_k \cdot R_k$
  for α_k in $\boldsymbol{W}$ **do**
    $\alpha_k = \alpha_k \frac{exp\{\frac{\sum_{i=1}^{m} E(\pi(c_i, D_i, R_k), Y_i) - \sum_{i=1}^{m} E(\pi(c_i, D_i, f_t), Y_i)}{m}\}}{Z_m}$,
  end for

  if $\sum_{i=1}^{m} E(\pi(c_i, D_i, \sum_{k=1}^{3} \alpha_k \cdot R_k), Y_i) < \sum_{i=1}^{m} E(\pi(c_i, D_i, f_t), Y_i)$ **then**
    return f_t
  end if
end for

Output:

$R(\boldsymbol{W}) = f_t$

set expansion problem on social media, so we unavoidably take Qadir's lexicon induction method as baseline, which is more or less distinguishing compared with our task.

Furthermore, to demonstrate our synthetic ranking model is effective, we also compare the ranking results with that only using semantic similarity from Word2Vec, and show the increase in precision.

4.2 Evaluation

Since the output of all these approaches are ranked lists, we adopt *Rank Precision@N*, which is commonly used for evaluation of entity set expansion techniques [13] and defined as follow:

Precision@N : The percentage of correct entities among the top N entities in the ranked list.

For each entity class, we collected the top 100 terms in the ranked lists generated by each of the approaches, and provided the sets to three individual annotators. The annotators were given only the initial sets of seed examples as guidelines, without any redundant definitions or descriptions of semantic class, and were asked to determine if each term is a positive or negative instance of that set. At last we assign class membership to each term follow the majority voting rule, that is, a term will be judged as positive only if more than two annotators deem it is.

4.3 Results

In our experiment, we use a word embedding model of 200 dimensions, and set the window size to 5, minimum word count to 5. To train our synthetical rank model, we select 10 semantic class sets, being more generic and commonly-used in set expansion tasks, which are completely not same as the sets that are intentionally used to perform our entity extraction method. Finally we get the set of coordination coefficient vector as: $\boldsymbol{W} = (0.49578957, 0.26961497, 0.23459546)$.

First the number of initial seed entities is fixed to 3, and for four methods, we present the precisions at the top 5-, 10-, 20-, 50- and 100-ranked positions (i.e., precisions@5, 10, 20, 50 and 100). The detailed experimental results are shown in Table 5 for both the *Common-class* and the *New-class*.

Method	***Common-class***					***New-class***				
	5	**10**	**20**	**50**	**100**	**5**	**10**	**20**	**50**	**100**
CPIM	0.35	0.35	0.37	0.34	0.36	0.48	0.44	0.36	0.30	0.26
TSLIM	0.74	0.65	0.60	0.45	0.40	0.61	0.50	0.42	0.39	0.37
Word2Vec	1.00	0.99	0.96	0.86	0.73	1.00	0.99	0.92	0.80	0.67
Our method	**1.00**	**0.99**	**0.97**	**0.90**	**0.76**	**1.00**	**0.99**	**0.95**	**0.84**	**0.74**

Table 5. Precision @ top N for four methods (with 3 seeds).

From Table 5, we observe that in *Common-class*, on average our method outperforms CPIM by about $40-65\%$, TSLIM by about $26-45\%$ and Word2Vec by about $1-4\%$. And in *new-class* our method outperforms more compared with all these three approaches, which is to $48-59\%$, $37-52\%$ and $3-6\%$. It indicates that our method is more effective when dealing with newly-presented semantic class sets.

To test the sensitivity of initial seeds, we also vary the strategies of selecting seed entities from candidate seed sets. We adopt three different manners:

- *Random Manner:* Randomly select 3 entities from candidate seed set.
- *Maximum Similarity:* Select the most 3 semantically similar entities as seeds.
- *Maximum Frequency:* Select the most 3 frequent entities occurred in corpus as seeds.

Then we use different size of corpus to analyze the corpus effects on performance. Due to space constrains, we present only average results of *Common-class* sets and *New-class* sets. All that results are shown in Table 6. From Table 6 we

Seed Select	**Corpus size**	**5**	**10**	**20**	**50**	**100**
Random	**1,398,511**	0.91	0.85	0.77	0.64	0.51
MaxSim	**1,398,511**	0.99	0.95	0.86	0.73	0.58
MaxFreq	**1,398,511**	1.00	0.96	0.86	0.73	0.59
MaxFreq	**4,080,031**	1.00	0.97	0.93	0.85	0.73
MaxFreq	**9,582,314**	**1.00**	**0.99**	**0.96**	**0.86**	**0.74**

Table 6. Precision @ top N when varying seed sets and corpus size (with 3 seeds).

can see that using MaxFreq to select seeds performs better. We speculate that the reason is, whether seed entities are common in corpus has a great influence on the precision result. A seed more frequent in corpus is able to bring more information in constructing patterns and rules, for example, "BurgerKing" and "KFC" as seeds are better than "WhataBurger". Hence uncommon and infrequent entities should be avoided when generating initial seed sets. In addition, we speculate that the relevancy in semantic among seed entities also affects much. It is probably because more similar seeds are often more representative for their class, and may prevent the semantic center drift. Such as the seed set {"ukulele", "violin", "guitar"} is better than {"piano", "violin", "guitar"}, where the former is more likely to have the center in "string instruments". Not surprisingly, enlarging corpus size have a significant impact on set expansion performance. One side the chunk parsing and word embedding model training process work better on larger corpus, and on the other side, most terms occur more times with the corpus growing, which greatly improves the entity sparseness problem.

5 Conclusion

We present a novel entity set expansion method that is able to expand newfangled semantic class on social media text. We first adopt a word embedding model to generate candidate sets and get scores of semantic similarity. And the other two scores are acquired by using some connective patterns and prefix rules which are specific in social media text. Then we propose a synthetical rank model to integrate the feature scores and show evident improvement promotion in some newly-presented entity class.

The demarcation and definition of "newly-presented" entities need further investigation. And as further work direction, we will take advantage of more features from accessible social media platform, such as userinfos, geographical location information and published time attributes.

6 Acknowledgements

This work was supported by the National High-tech Research and Development Program (863 Program) (No. 2014AA015105) and National Natural Science Foundation of China (No. 61602490).

References

1. Wang, Richard C and Cohen, William W: Language-Independent Set Expansion of Named Entities Using the Web. In: IEEE International Conference on Data Mining, pp. 342–350. IEEE (2007)
2. Wang, Richard C and Cohen, William W: Iterative Set Expansion of Named Entities Using the Web. In: Eighth IEEE International Conference on Data Mining, pp. 1091–1096. IEEE (2009)

3. Wang, Richard C and Cohen, William W. SEAL, http://http://rcwang.com/seal
4. He, Yeye and Xin, Dong: SEISA:set expansion by iterative similarity aggregation. In: International Conference on World Wide Web, pp. 427–436. WWW 2011, Hyderabad, India (2011)
5. Dalvi, Bhavana Bharat and Cohen, William W and Callan, Jamie: WebSets: extracting sets of entities from the web using unsupervised information extraction. In: ACM International Conference on Web Search and Data Mining, pp. 243–252. ACM (2012)
6. Thelen, Michael and Riloff, Ellen: A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In: Acl-02 Conference on Empirical Methods in Natural Language Processing, pp. 212–221. ACL (2002)
7. Wang, Richard C. and Cohen, William W: Automatic Set Instance Extraction using the Web. In: ACL 2009, Proceedings of the Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing of the Afnlp, pp. 441–449. ACL, Singapore (2009)
8. Etzioni, Oren and Cafarella, Michael and Downey, Doug and Kok, Stanley and Popescu, Ana Maria and Shaked, Tal and Soderland, Stephen and Weld, Daniel S. and Yates, Alexander: Web-scale information extraction in knowitall. pp. 100-110. WWW (2004)
9. Widdows, Dominic and Dorow, Beate: A graph model for unsupervised lexical acquisition. In: International Conference on Computational Linguistics, pp. 1093–1099. (2002)
10. Sarmento, Luis and Jijkuon, Valentin and De Rijke, Maarten and Oliveira, Eugenio: More like these: growing entity classes from seeds. In: Sixteenth ACM Conference on Conference on Information and Knowledge Management, pp. 959–962. ACM (2007)
11. Talukdar, Partha Pratim and Brants, Thorsten and Liberman, Mark and Pereira, Fernando: A Context Pattern Induction Method for Named Entity Extraction. In:Computational Natural Language Learning, pp. 141–148. CoNLL-X (2006)
12. Ghahramani, Zoubin and Heller, Katherine A: Bayesian Sets. (2005)
13. Li, Xiao Li and Zhang, Lei and Liu, Bing and Ng, See Kiong: Distributional similarity vs. PU learning for entity set expansion. In:ACL 2010 Conference Short Papers, pp. 359–364. ACL (2010)
14. Ritter, Alan and Sam, Clark and Mausam and Etzioni, Oren: Named entity recognition in tweets. (2011)
15. Li, Chenliang and Weng, Jianshu and He, Qi and Yao, Yuxia and Datta, Anwitaman and Sun, Aixin and Lee, Bu Sung: TwiNER: named entity recognition in targeted twitter stream. In:Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, pp. 721–730. (2012)
16. Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A. Greenwood, Diana Maynard, Niraj Aswani: TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. In:Proceedings of the International Conference on Recent Advances in Natural Language Processing. Association for Computational Linguistics. (2013)
17. GATE, https://gate.ac.uk/wiki/twitie.html
18. Qadir, Ashequl and Mendes, Pablo N and Gruhl, Daniel and Lewis, Neal: Semantic lexicon induction from twitter with pattern relatedness and flexible term length. In:Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2432–2439 (2015)
19. Mikolov, Tomas and Chen, Kai and Corrado, Greg and Dean, Jeffrey: Efficient Estimation of Word Representations in Vector Space. In:Computer Science. (2013)
20. Xu, Jun and Li, Hang : AdaRank: a boosting algorithm for information retrieval. In:International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 391–398 (2007)