

A Joint Model for Entity Set Expansion and Attribute Extraction from Web Search Queries

Zhenzhong Zhang, Le Sun, Xianpei Han

State Key Laboratory of Computer Sciences, Institute of Software, Chinese Academy of Sciences, China.
{zhenzhong, sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Entity Set Expansion (ESE) and Attribute Extraction (AE) are usually treated as two separate tasks in Information Extraction (IE). However, the two tasks are tightly coupled, and each task can benefit significantly from the other by leveraging the inherent relationship between entities and attributes. That is, 1) an attribute is important if it is shared by many typical entities of a class; 2) an entity is typical if it owns many important attributes of a class. Based on this observation, we propose a joint model for ESE and AE, which models the inherent relationship between entities and attributes as a graph. Then a graph reinforcement algorithm is proposed to jointly mine entities and attributes of a specific class. Experimental results demonstrate the superiority of our method for discovering both new entities and new attributes.

Introduction

Entity Set Expansion (ESE) aims at acquiring new entities of a particular class using a few seed entities. Attribute Extraction (AE) aims at acquiring a set of relevant attributes which can capture the most prominent properties of a given class (e.g., “capital” for the “country” class). ESE and AE are useful in many applications such as dictionary construction (Cohen and Sarawagi, 2004), word sense disambiguation (Pantel and Lin, 2002), query refinement (Hu et al., 2009) and query expansion (Bellare et al., 2007).

Numerous methods have been proposed to ESE (Cucchiarelli and Velardi, 2001; Etzioni et al., 2005; Pasca, 2007a; Riloff and Jones, 1999; Wang and Cohen, 2007; Wang and Cohen, 2008) and AE (Tokunaga et al., 2005; Yoshinaga and Torisawa, 2007; Cui et al., 2009; Pasca and Durme, 2007; Pasca and Durme, 2008). Traditional methods mostly treat ESE and AE as two separate tasks. For example, Pasca and Durme (2008) employed a pipeline

method, which first extracted entities from web documents using two lexical patterns and then leveraged these entities to extract attributes from query log. However, such pipeline architecture often suffers from error propagation.

Moreover, ESE and AE tasks are tightly coupled and they can benefit from the information provided by the other. For instance, on one hand, AE systems often suffer from data sparseness (Alfonseca et al., 2010), which can be mitigated by providing more entities of the given class using an ESE system. On the other hand, ESE systems often suffer from the semantic drift problem, that is, the expansion category may change during the new entity discovery step (e.g., expanding “New York” to the “country” class). The attributes mined by AE systems can provide critical information to resolve the semantic drift problem. For example, if a set of attributes (e.g., “capital”, “president”, and “embassy”) of a given class (e.g., “country”) are learned by an AE system, we can infer that an entity (e.g., “New York”) without these important attributes is unlikely to belong to the given class.

Based on the above observation, this paper proposes a joint model for ESE and AE, which can effectively improve both tasks by exploiting the relationship between entities and attributes using a graph reinforcement algorithm. The intuition behind our approach is that an attribute is important to a given class if it is shared by many typical entities of this class, and an entity is typical if it owns many important attributes of the class. Based on the above intuition, this paper first proposes a graph representation to model the relationship between entities and attributes. To resolve the data sparseness problem, the graph is extended with the relationship between entities and the relationship between attributes. Based on the graph representation, we propose a graph based reinforcement algorithm for better ESE and AE, which can jointly mine entities and attributes by exploiting the relationships captured by the graph.

We conducted large-scale experiments on the AOL search data set (Pass et al., 2006). Experimental results

showed that our approach can achieve competitive performance.

Related Work

In recent years, ESE has received considerable attentions from both research (Cafarella et al., 2005; Pantel and Ravichandran, 2004; Pantel et al., 2009; Pasca, 2007; Wang and Cohen, 2007; Wang and Cohen, 2008) and industry communities (e.g., Google Sets). Due to the limited supervision provided by ESE (in most cases only 3-5 seeds are given), most ESE systems employ bootstrapping techniques. That is, the entity set is iteratively expanded through a pattern generation step and an entity extraction step. These methods use a variety of textual data sources, including web documents (Cafarella et al., 2005; Pantel et al., 2009), encyclopedia (Bing et al., 2013), and web search query log (Pasca, 2007a; Xu et al., 2009).

Technologies on AE have been developed in Information Extraction (IE). A variety of attribute extraction methods mine web documents to extract and rank a list of attributes for a given class (Tokunaga et al., 2005; Yoshinaga and Torisawa, 2007; Cafarella et al., 2008). For example, the method presented in (Tokunaga et al., 2005) uses manually-created lexicon syntactic patterns on web documents to extract candidate attributes for given classes. The candidate attributes are ranked according to several frequency statistics. As an alternative to web documents, human compiled encyclopedia (e.g., Wikipedia) can also be exploited as sources for attribute extraction (Suchanek et al., 2007; Nastase and Strube, 2008; Wu et al., 2008). Recently, web search queries have also been considered as a textual data source for attribute extraction. For example, the method in (Pasca and Durme, 2007) uses lexical syntactic patterns to extract attributes from search queries. Pasca (2007b) presented a method that employs seed attributes to guide the extraction.

There are also some methods for simultaneous extraction of entities and attributes. For example, the method described in (Pasca and Durme, 2008) employs pipeline architecture for the simultaneous extraction of entities and attributes. The method first extracts entities by applying a few extraction patterns to web documents while guiding the extraction based on the contents of query log. Then, it extracts attributes by mining query logs while guiding the extraction based on a few seed attributes. The semi-supervised learning method presented in (Bing et al., 2013) mines the semi-structured data records on the web to achieve the goal of new entity discovery and attribute extraction. This method takes a few Wikipedia entities as seed input and explores their attribute infoboxes to obtain clues for the discovery of more entities and attributes. Different from their solution, we present an unsupervised ap-

proach to jointly perform ESE and AE using web search queries. In particular, we represent candidate entities and attributes as a bipartite graph where the edges capture the relatedness between candidate entities and attributes. Then we extend the graph by constructing edges between candidate entities/attributes, which capture the relatedness between candidate entities/attributes. We weight and rank candidate entities and attributes according to their importance by deploying graph based reinforcement techniques.

The Joint Model for Entity Set Expansion and Attribute Extraction

In this section, we propose a graph based method for joint ESE and AE. Given a few seed entities of a particular class, our method jointly extracts a ranked list of entities and attributes for the given class. We first propose a graph representation which can capture the relationship between entities and attributes; then we present how to construct the graph from web search queries. Finally, we propose an unsupervised graph based reinforcement algorithm which can effectively mine new entities and attributes.

Graph Representation

In this section, we propose a graph representation, which can: 1) capture the relationship between entities and attributes for joint ESE and AE; and 2) further capture the relationship between neighboring entities (and attributes) to solve the data sparseness problem.

Given a particular class c , we use $E=\{e_1, \dots, e_N\}$ to denote its candidate entity set and $A=\{a_1, \dots, a_M\}$ to denote its candidate attribute set. The relationship between these candidate entities and attributes are modeled as a bipartite graph, as illustrated in Figure 1. Each candidate entity (and attribute) is represented as a node in the graph. The relationship between entity e_i and attribute a_j is represented as a weighted edge between e_i and a_j . The weight $w_{i,j}$ is a positive real number, indicating the strength of the relatedness between e_i and a_j .

Based on the above bipartite graph representation, graph based mutual reinforcement techniques (e.g., HITS algorithm (Kleinberg, 1998)) can be utilized to weight and rank candidate entities and attributes. However, data sparseness may affect the importance scores of long-tail entities and attributes, i.e., entities (or attributes) linking few attributes (or entities) in the graph. In order to alleviate the negative impact of data sparseness, we leverage nearest neighbors of a candidate entity to smooth its importance score. Specifically, we add edges between candidate entities to capture their relationship, whose weights indicate the similarity between them. The importance score of a candidate entity can then propagates to its neighbors along the edges between them. Similarly, we also add edges between

neighboring candidate attributes. For demonstration, Figure 2 gives an example graph.

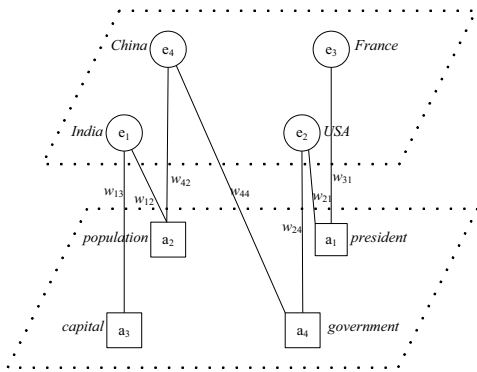


Figure 1: A bipartite graph representation for candidate entities and attributes, where e and a are candidate entities and attributes respectively.

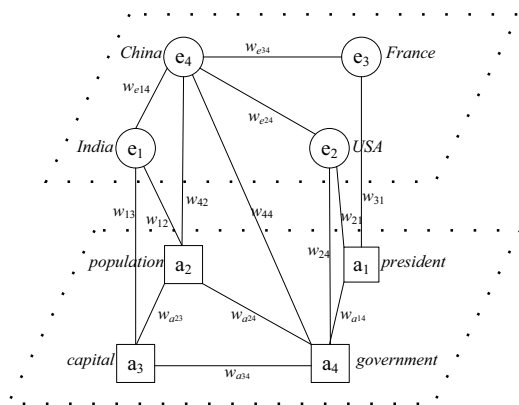


Figure 2: An extended graph representation for candidate entities and attributes, where w_e denote the weights of edges between entities and w_a denote the weights of edges between attributes.

Graph Construction

Our approach constructs the graph representation from web search queries via the following steps.

Step 1: Generate Candidate Entities and Attributes

Given a particular class (e.g., “country”) and a set of seed entities (e.g., “Japan”, “India”, and “Germany”), our approach identifies query patterns by matching seed entities with web search queries, then these query patterns are used to extract new candidate entities from web search queries. A query pattern is generated by matching an entity with queries, and the remainder of matched queries is used as query pattern. For example, we can identify the query pattern “weather in E ” by matching “Germany” with query “weather in Germany”.

The collected candidate entities are then used to extract candidate attributes. In this study we employ the method in (Pasca and Durme, 2007) for candidate attribute extraction

from web search queries. We also record the frequency of an entity-attribute pair in a query log corpus, denoted as $fre(e_i, a_j)$, for attribute filtering.

Step 2: Filter Candidates

The above step will generate many noisy candidate entities and attributes, which makes the graph unnecessarily large. To reduce the graph size, we filter candidate entities using the similarity between candidate entities and seed entities. Inspired by Pasca (2007a), we represent each candidate entity as the query pattern vector, with each dimension corresponding to a query pattern. For example, “Germany” will be represented as a query pattern vector {weather in E , ...}. We weight each query pattern using its frequency. After computing the similarity between candidate entities and seed entities, we keep the top N ($N=500$ in this paper) nearest neighbors for each seed entity. Then we collect candidate attributes from the entity-attribute pairs (e_i, a_j) , where e_i belongs to the top N candidate entities with $fre(e_i, a_j) \geq 5$.

Step 3: Construct Edges

As described in *Graph Representation* section, there are three types of edges in the graph: the edges between candidate entities and attributes, the edges between candidate entities, and the edges between candidate attributes.

For each candidate entity-attribute pair (e_i, a_j) collected by Step 2, we add an edge between e_i and a_j and set its weight as 1.

For each pair of candidate entities, e_i and e_j , our approach adds an edge between them if the cosine similarity between their query pattern vectors exceeds a pre-defined threshold δ ($\delta = 0.6$ in this paper), with the cosine similarity as the edge weight.

We add edges between candidate attributes in a similar way of constructing the edges between entities. If the similarity between two candidate attributes exceeds a pre-defined threshold θ ($\theta = 0.2$ in this paper), we add an edge between them with the similarity as its weight.

Importance Propagation

In this section, we propose an unsupervised graph based reinforcement algorithm, which allows the information from ESE to be used for AE, and vice versa. Specifically, our algorithm jointly ranks candidate entities and attributes based on the relationships in the above mentioned graph representation. The assumption of our method is:

Hypothesis 1: The entities linked by many important attributes tend to be typical and the attributes linked by many typical entities tend to be important.

Based on the above hypothesis, we weight candidate entities and attributes in a mutual recursion way. Let $s_i(e_i)^k$ be the importance score of entity e_i at the k -th iteration, $s_i(a_j)^k$

be the importance score of attribute a_j at the k -th iteration, $a_m - e_i$ be the edge between a_m and e_i , and $w(a_m - e_i)$ be the weight of the edge between e_i and a_m . Here the subscript of $s_i(\cdot)$ denotes that it is computed based on *Hypothesis 1*. Then the importance scores of e_i and a_j at the $(k+1)$ -th iteration are calculated as

$$s_1(e_i)^{k+1} = \sum_{\forall m: a_m - e_i} s_1(a_m)^k \frac{w(a_m - e_i)}{\sum_{\forall n: a_m - e_n} w(a_m - e_n)} \quad (1)$$

$$s_1(a_j)^{k+1} = \sum_{\forall i: e_i - a_j} s_1(e_i)^k \frac{w(e_i - a_j)}{\sum_{\forall n: e_i - a_n} w(e_i - a_n)} \quad (2)$$

However, due to data sparseness the above graph based mutual reinforcement approach may underestimate the importance scores of long-tail entities (and attributes). To alleviate this problem, our approach leverages the relationship between entities (and attributes) based on the following hypothesis:

Hypothesis 2: The entities linked by many typical entities tend to be typical. The attributes linked by many important attributes tend to be important.

According to *Hypothesis 2*, the importance scores of e_i and a_j at the $(k+1)$ -th iteration are calculated as

$$s_2(e_i)^{k+1} = (1-\beta) \times s_2(e_i)^k + \beta \times \sum_{\forall j: e_j - e_i} s_2(e_j)^k \frac{w(e_j - e_i)}{\sum_{\forall n: e_j - e_n} w(e_j - e_n)} \quad (3)$$

$$s_2(a_j)^{k+1} = (1-\beta) \times s_2(a_j)^k + \beta \times \sum_{\forall i: e_i - a_j} s_2(e_i)^k \frac{w(e_i - a_j)}{\sum_{\forall n: e_i - a_n} w(e_i - a_n)} \quad (4)$$

where β is a parameter ($0 \leq \beta \leq 1$). $s_2(e_i)^k$ is the importance score of e_i at the k -th iteration. Equations (3) and (4) consist of two terms. The first term is the importance score from the node itself, ensuring that its importance does not deviate too much from the previous iteration. The second term is the incoming importance score from neighboring nodes.

Finally, our approach weights each node by combining the above two clues using a linear interpolation method:

$$s(e_i)^{k+1} = (1-\alpha) \times s_1(e_i)^{k+1} + \alpha \times s_2(e_i)^{k+1} \quad (5)$$

$$s(a_j)^{k+1} = (1-\alpha) \times s_1(a_j)^{k+1} + \alpha \times s_2(a_j)^{k+1} \quad (6)$$

Our approach iteratively updates the importance scores of entities and attributes using equation (5) and (6). After each iteration, the importance score of each entity (attribute) is normalized such that the sum importance scores of all entities (attributes) is 1. The iteration stops when one of two conditions is met – either the change of importance score is minimal or the number of iteration exceeds a predefined threshold. Then the candidate entities and attributes are ranked according to their importance scores, respectively.

Experiments

Data Set

We use the AOL search data (Pass et al., 2006) as the mining corpus, which contains about 20 million web search queries collected from 650 thousand users over three months.

Target Classes

We evaluate the performance of our method on the following categories (seed entities are within {}):

- Country: {china, france, canada, russia, germany};
- American City: {new york, chicago, houston, boston, phoenix};
- Company: {walmart, kmart, staples, sears, pfizer};
- Disease: {allergy, asthma, lupus, kidney cancer, hiv};
- Book: {mice and men, cold sassy tree, the yellow wallpaper, moby dick, the great gatsby}.

Parameter Setting

The parameters α and β from *Importance Propagation* section are empirically set to 0.2 and 0.1 respectively. And we initialize the weight of an entity to 1 if it is a seed entity and 0 otherwise.

Entity Set Expansion

For entity set expansion, we compare our method with three baselines. The first baseline (referred to as *Bootstrap*) employs traditional bootstrapping method to extract entities. The second baseline (*Nearest Neighbors*) is one of the state-of-the-art methods (Pasca, 2007a), which extracts top N candidate entities nearest with seed entities. The third baseline (*GraphMethod_Entity*) is our approach but only leverages the relationship between entities. We use precision at top K ($P@K$ and $K=10, 20, 30$) to evaluate the ranked entities.

The overall entity expansion results are given in Table 1. From Table 1, we can see that

1) Our approach has the best average precisions at all top K . This result verifies that the joint method for entity and attribute extraction is beneficial to entity extraction;

2) Compared with *GraphMethod_Entity*, our approach achieves 16%, 11%, and 12% precision improvement at $P@10$, $P@20$, and $P@30$, respectively. These precision improvements show that attributes are helpful for entity expansion;

3) Our approach has better performance than *Bootstrap*, especially at $P@20$ and $P@30$. We believe this is because *Bootstrap* suffers from the semantic drift problem; in contrast our approach leverages attributes to reduce the semantic drift in each iteration;

Class	<i>Bootstrap</i>			<i>Nearest Neighbors</i>			<i>GraphMethod Entity</i>			Our approach		
	p@10	p@20	p@30	p@10	p@20	p@30	p@10	p@20	p@30	p@10	p@20	p@30
Country	0.90	0.70	0.60	1.00	0.70	0.60	0.70	0.60	0.53	1.00	0.80	0.73
USA city	1.00	0.95	0.77	1.00	0.95	0.83	0.80	0.60	0.63	0.80	0.65	0.60
Disease	0.70	0.75	0.70	0.80	0.70	0.67	0.80	0.80	0.70	1.00	0.90	0.90
Company	0.90	0.65	0.47	0.90	0.85	0.75	0.70	0.65	0.53	0.90	0.75	0.63
Book	0.60	0.55	0.43	0.50	0.55	0.53	0.70	0.55	0.40	0.80	0.65	0.53
average	0.82	0.72	0.59	0.84	0.75	0.67	0.74	0.64	0.56	0.90	0.75	0.68

Table 1: The results of entity set expansion of different approaches.

Class	Method	Top ten entities
Country	<i>Bootstrap</i>	china, france, canada, russia, germany, mexico, spain, italy, ireland, africa
	<i>Nearest Neighbors</i>	canada, india, china, germany, france, mexico, spain, italy, ireland, new zealand
	<i>GraphMethod_Entity</i>	canada, germany, france, russia, china, mexico, europe, missouri, burlington wa, new zealand
	Our method	china, germany, canada, france, russia, mexico, italy, ireland, spain, india
American city	<i>Bootstrap</i>	new york, chicago, houston, boston, phoenix, philadelphia, dallas, ny, nyc, seattle
	<i>Nearest Neighbors</i>	new york, chicago, miami, houston, denver, phoenix, san diego, boston, philadelphia, los angeles
	<i>GraphMethod_Entity</i>	boston, chicago, phoenix, houston, new york, atlanta, portland, georgia, pennsylvania, south carolina
	Our method	new york, chicago, boston, houston, phoenix, ny, georgia, atlanta, virginia
Disease	<i>Bootstrap</i>	allergy, asthma, lupus, kidney cancer, hiv, bipolar, thyroid, ms, fibromyalgia, anxiety
	<i>Nearest Neighbors</i>	ms, hiv, prostate cancer, lupus, allergy, menopause, asthma, lyme disease, chlamydia, uti
	<i>GraphMethod_Entity</i>	asthma, hiv, allergy, lupus, kidney cancer, early pregnancy, menopause, flu, std, stroke
	Our method	hiv, lupus, asthma, allergy, kidney cancer, stroke, breast cancer, stomach cancer, high blood pressure, hpv
Company	<i>Bootstrap</i>	walmart, kmart, staples, sears, pfizer, home depot, wal-mart, wal mart, costco, sports
	<i>Nearest Neighbors</i>	walmart, sears, kmart, wal-mart, home depot, staples, sports, wallmart, costco, k-mart
	<i>GraphMethod_Entity</i>	walmart, kmart, sears, staples, pfizer, sports, nba, blockbuster, wal-mart, san antonio
	Our method	walmart, sears, kmart, pfizer, staples, home depot, wal mart, kroger, medical, wal-mart
Book	<i>Bootstrap</i>	mice and men, cold sassy tree, the yellow wallpaper, moby dick, the great gatsby, tom sawyer, the great gatsby, california, florida, the movie
	<i>Nearest Neighbors</i>	cold sassy tree, a seperate peace, the yellow wallpaper, facts, the great gatsby, characters, mice and men, the movie , supreme court, resume
	<i>GraphMethod_Entity</i>	mice and men, the yellow wallpaper, cold sassy tree, moby dick, the great gatsby, the movie, the great gatsby, a house, king lear, green
	Our method	mice and men, the yellow wallpaper, the great gatsby, cold sassy tree, moby dick, pride and prejudice, frankenstein, poetry, the aeneid, animal farm

Table 2: Top ten ranked entities for each class.

4) Our approach has similar performance as *Nearest Neighbors* at P@20 and P@30, but has better performance at P@10 with 6% precision improvement, which shows our approach can achieve competitive performance. We found that our method achieves better performance than *Nearest Neighbors* in all target classes except “City”. This is because many attributes extracted for “City” are also the attributes of “State”, e.g., “population”, “map”, “history” and so on. Hence our method expanded some states into “City” class.

For demonstration, we also list the top ten entities returned by different approaches in Table 2. We can see that our method extracts more accurate entities than baselines in all target classes except “City”. Our method expands Virginia and Georgia into “City” class. This is because many attributes extracted for “City” are also the attributes of “State”. In this situation, attributes cannot guide the entity extraction process well.

Class	Pipeline method			Our approach		
	p@10	p@20	p@30	p@10	p@20	p@30
Country	0.70	0.68	0.70	0.95	0.85	0.83
USA city	0.60	0.55	0.47	0.65	0.60	0.47
Disease	0.80	0.78	0.72	0.80	0.80	0.75
Company	0.55	0.48	0.40	0.70	0.63	0.62
Book	0.35	0.38	0.35	0.60	0.53	0.42
average	0.60	0.57	0.52	0.74	0.68	0.62

Table 3: The results of attribute extraction of different approaches.

Class	Method	Top ten attributes
Country	Pipeline method	maps, history, pictures, capital, population, flag, picture, country, a map, photos
	Our method	maps, pictures, history, capital, people, president, leader, flag, population, government
American city	Pipeline method	map, pictures, city, university, history, zip code, archdiocese, best, population, maps
	Our method	map, pictures, zip code, maps, population, diocese, history, address, art institute, children's museum
Disease	Pipeline method	symptoms, treatment, signs, pictures, causes, symptoms, symptoms, home remedies, cause, herbs
	Our method	symptoms, pictures, signs, symptoms, treatment, risk, symptoms, survival rate, causes, herbs
Company	Pipeline method	coupons, location, address, history, list, pictures, phone number, owner, jobs, ceo
	Our method	phone number, ceo, address, mission statement, owner, job applications, photo, apply, employees, current ceo
Book	Pipeline method	summary, pictures, picture, pics, author, cast, causes, list, outline, effects
	Our method	spark notes, plot summary, criticism, summary, explanation, copy, author, analysis, picture, citing

Table 4: Top ten ranked attributes for pipeline based baseline and our approach.

Attribute Extraction

For AE, we compare our approach with a pipeline method, which uses the output of an entity expansion system as input, and then leverages the method in (Pasca and Durme, 2007) to extract attributes. In our experiments, we use the entities outputted by *Nearest Neighbors*.

We employ the metric in (Pasca and Durme, 2007) to measure the performance of each approach, where each attribute is manually assigned a correctness label (including *vital*, *okay*, and *wrong*). The precision score over a ranked list of attributes is computed by converting the correctness labels to numeric values (*vital*=1, *okay*=0.5, and *wrong*=0). Then P@K in a given list is measured as the average of the assigned values of the first K attributes.

The overall results of attribute extraction are given in Table 3. From Table 3, we can see that our approach has better performance than the pipeline method with 14%, 11%, 10% improvement at the average P@10, P@20, P@30, respectively. We believe this is because the pipeline method suffers from the error propagation problem. Com-

pared with the pipeline method, our approach can leverage attributes to guide entity extraction and then these accurate entities can in turn help the attribute extraction. For demonstration, we also list the top ten attributes returned by different approaches in Table 4. Our method extracts more prominent attributes than the pipeline method. The attributes extracted for “City” class contain “population”, “map”, “history” and so on. These attributes also propagate their importance scores to the “State” entities (e.g., Georgia), which causes our method expands some “State” entities into “City” class.

Conclusion

In this paper, we propose a joint model for Entity Set Expansion and Attribute Extraction. Our approach first mines a large set of candidate entities and extracts a large set of candidate attributes, and then ranks the candidate entities and attributes together based on graph reinforcement. The graph reinforcement algorithm leverages the relatedness

between candidate entities and attributes, the relatedness between candidate entities, and the relatedness between candidate attributes. Experiment results demonstrate the propose approach's superiority for discovering new entities and extracting attributes. In our future work, we plan to incorporate more resources (e.g., Wikipedia) to capture the relationship between entities and attributes, entities and entities, and attributes and attributes.

Acknowledgements

This work is supported by the National High Technology Development 863 Program of China under Grants no. 2015AA015405, and the National Natural Science Foundation of China under Grants no. 61433015, 61272324 and 61572477. Moreover, we sincerely thank the reviewers for their valuable comments.

References

- Alfonseca, E.; Pasca, M.; and Robledo-Arnumcio, E. 2010. Acquisition of Instance Attributes via Labeled and Related Instances. In *Proceedings of SIGIR'10*, pages 58-65.
- Bellare, K. et al. 2007. Lightly-Supervised Attribute Extraction. In *NIPS Workshop on Machine Learning for Web Search*.
- Bing, L.; Lam, W.; and Wong, T. 2013. Wikipedia Entity Expansion and Attribute Extraction from the Web Using Semi-supervised Learning. In *Proceedings of WSDM'13*, pages 567-576.
- Cafarella, M. J.; Downey, D.; Soderland, S.; and Etzioni, O. 2005. KnowItNow: Fast, Scalable Information Extraction from the Web. In *Proceedings of EMNLP'05*, pages 563-570.
- Cafarella, M. J.; Halevy, A.; Wang, D. Z.; Wu, E.; and Zhang, Y. 2008. Webtables: Exploring the Power of Tables on the Web. In *Proceedings of the VLDB Endowment archive*, pages 538-549, volume 1.
- Cohen, W. W. and Sarawagi, S. 2004. Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods. In *Proceedings of KDD'04*, pages 89-98.
- Cucchiarelli, A. and Velardi, P. 2001. Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence. In *Proceedings of Computational Linguistics*, pages 123-131, volume 27.
- Cui, G.; Lu, Q.; Li, W.; and Chen, Y. 2009. Automatic Acquisition of Attributes for Ontology Construction. In *Proceedings of the 22nd International Conference on Computer Processing of Oriental Languages. Language Technology for the Knowledge-based Economy*, pages 248-259.
- Etzioni, O. et al. 2005. Unsupervised Named-Entity Extraction from the Web: An Experimental Study. In *Proceedings of Artificial Intelligence*, pages 91-134, volume 165.
- Hu, J.; Wang, G.; Lochovsky, F.; Sun, J. and Chen, Z. 2009. Understanding User's Query Intent with Wikipedia. In *Proceedings of WWW'09*, pages 471-480.
- Kleinberg, J. 1998. Authoritative Sources in a Hyperlinked Environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*.
- Nastase, V. and Strube, M. 2008. Decoding Wikipedia Categories for Knowledge Acquisition. In *Proceedings of AAAI'08*, pages 1219-1224.
- Pantel, P. and Lin, D. 2002. Discovering Word Senses from Text. In *Proceedings of KDD'02*, pages 613-619.
- Pantel, P. and Ravichandran, D. 2004. Automatically Labeling Semantic Classes. In *Proceedings of HLT/NAACL*, pages 321-328, volume 4.
- Pantel, P. et al. 2009. Web-Scale Distributional Similarity and Entity Set Expansion. In *Proceedings of EMNLP'09*, pages 938-947.
- Pasca, M. 2007a. Weakly-supervised Discovery of Named Entities Using Web Search Queries. In *Proceedings of CIKM'07*, pages 683-690.
- Pasca, M. 2007b. Turning Web Text and Search Queries into Factual Knowledge: Hierarchical Class Attribute Extraction. In *Proceedings of AAAI'08*, pages 1225-1230.
- Pasca, M. and Durme, B. V. 2007. What You Seek Is What You Get: Extraction of Class Attributes from Query Logs. In *Proceedings of IJCAI'07*, pages 2832-2837.
- Pasca, M. and Durme, B. V. 2008. Weakly-Supervised Acquisition of Open-Domain Classes and Class Attributes from Web Documents and Query Logs. In *Proceedings of ACL'08*, pages 19-27.
- Pass, G.; Chowdhury, A.; and Torgeson, C. 2006. A picture of search. In *INFOSCALE*, paper 1.
- Riloff, E. and Jones, R. 1999. Learning Dictionaries for Information Extraction Using Multi-level bootstrapping. In *Proceedings of AAAI'99*, pages 474-479.
- Suchanek, F.; Kasneci, G. and Weikum, G. 2007. Yago: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *Proceedings of WWW'07*, pages 697-706.
- Tokunaga, K.; Kazama, J. and Torisawa, K. 2005. Automatic Discovery of Attribute Words from Web Documents. In *Proceedings of IJCNLP'05*, pages 106-118.
- Wang, R. C. and Cohen, W. W. 2007. Language-Independent Set Expansion of Named Entities using the Web. In *Proceedings of ICDM'07*, pages 342-350.
- Wang, R. C. and Cohen, W. W. 2008. Iterative Set Expansion of Named Entities Using the Web. In *Proceedings of ICDM'08*, pages 1091-1096.
- Wu, F.; Hoffmann, R. and Weld, D. 2008. Information Extraction from Wikipedia: Moving Down the Long Tail. In *Proceedings of KDD'08*, pages 731-739.
- Xu, G.; Yang, S. and Li, H. 2009. Named Entity Mining from Click-Through Data Using Weakly Supervised Latent Dirichlet Allocation. In *Proceedings of KDD'09*, pages 1365-1374.
- Yoshinaga, N. and Torisawa, K. 2007. Open-Domain Attribute-Value Acquisition from Semi-Structured Texts. In *Proceedings of the Workshop on Ontolex*, pages 55-66.