Learning from Negative Examples in Set-Expansion

Prateek Jindal
Dept. of Computer Science
UIUC
Urbana, IL, USA
jindal2@illinois.edu

Dan Roth
Dept. of Computer Science
UIUC
Urbana, IL, USA
danr@illinois.edu

***Abstract*—This paper addresses the task of set-expansion on free text. Set-expansion has been viewed as a problem of generating an extensive list of instances of a concept of interest, given a few examples of the concept as input. Our key contribution is that we show that the concept definition can be significantly improved by specifying some negative examples in the input, along with the positive examples. The state-of-the art centroid-based approach to set-expansion doesn't readily admit the negative examples. We develop an inference-based approach to set-expansion which naturally allows for negative examples and show that it performs significantly better than a strong baseline.**

I. Introduction

This paper addresses the task of set-expansion on free text. Set-expansion has been viewed as a problem of generating an extensive list of instances of a concept of interest, given a few examples of the concept as input. For example, if the seed-set is {*Steffi Graf*, *Martina Hingis*, *Serena Williams*}, the system should output an extensive list of female tennis players.

We focus on set-expansion from free text, as opposed to web-based approaches that build on existing lists. The key method used for set-expansion from free text is distributional similarity. For example, the state-of-the-art systems [1], [2] use a centroid-based approach wherein they first find the centroid of the entities in the seed-set and then find the entities that are similar to the centroid. Most of the work on set-expansion has focussed on taking only positive examples. For example, as discussed above, to produce a list of female tennis players, a few names of female tennis players are given as input to the system. However, just specifying a few female tennis players doesn't define the concept precisely enough. The set-expansion systems tend to output some male tennis players along with female tennis players. Specifying a few names of male tennis players as negative examples defines the concept more precisely.

Table I compares the state-of-the-art approach for set-expansion on free text with the approach presented in this paper. The table shows only a small portion of the lists generated by the system. We used 7 positive examples for both approaches, and only 1 negative example was used for the proposed approach. The errors have been underlined and italicized. We see that the output in the 1^{st} column is corrupted by male tennis players. Adding only 1 negative example to the seed-set improves the list-quality significantly. The second column contains no errors. *In this paper, we propose ways to learn from negative examples in set-expansion and show significant improvement.*

FEMALE TENNIS PLAYERS	
State-of-the-art	**This Paper**
Monica Seles	Mary Pierce
Steffi Graf	Monica Seles
Martina Hingis	Martina Hingis
Mary Pierce	Lindsay Davenport
Lindsay Davenport	Steffi Graf
Jennifer Capriati	Jennifer Capriati
Kim Clijsters	Kim Clijsters
Mary Joe Fernandez	Karina Habsudova
Nathalie Tauziat	Sandrine Testud
Kimiko Date	Kimiko Date
Conchita Martinez	Chanda Rubin
Anke Huber	Anke Huber
Judith Wiesner	Nathalie Tauziat
Andre Agassi	Jana Novotna
Pete Sampras	Conchita Martinez
Jana Novotna	Nathalie Dechy
Karina Habsudova	Amanda Coetzer
Jim Courier	Barbara Paulus
Justine Henin	Arantxa Sanchez-Vicario
Julie Halard	Amy Frazier
Meredith McGrath	Iva Majoli
Goran Ivanisevic	Magdalena Maleeva
Jelena Dokic	Jelena Dokic
Michael Chang	Julie Halard

Table I: This table compares the state-of-the-art approach for set-expansion on free text with the approach presented in this paper. The bold and italicized entries correspond to male tennis players and are erroneous. The addition of only 1 negative example to the seed-set improves the list-quality significantly. The second column contains no errors.

We present an inference-based approach to set-expansion which doesn't rely on computing the centroid. The new approach naturally allows for both positive and negative examples in the seed-set. We also extended the centroid-based approach so that it can accept negative examples. We used this improved version of centroid-based approach as a baseline system and show in the experiments that the inference-based approach we developed significantly outperforms this baseline.

II. Related Work

The task of *set-expansion* has been addressed in several works. We briefly discuss some of the most significant efforts towards this task. Google Sets and Boowa [3] are web-based set-expansion methods. For set-expansion on free-text ([4], [5], [1], [2]), pattern recognition and distributional similarity have primarily been used. Some works on set-expansion ([6], [7], [8]) have focussed on integrating information across several types of sources such as structured, semi-structured, unstructured text, query logs etc.

There has also been some work on the use of negative examples in set-expansion. Thelen and Riloff [9] and Lin et al. [10] present a framework to simultaneously learn several semantic classes. In this framework, instances which have been accepted by one semantic class serve as negative examples for all other semantic classes. This approach is limited because it necessitates the simultaneous learning of several semantic classes. Moreover, negative examples are not useful if the semantic classes addressed are not related to one another. Lin et al. note that it is not easy to acquire good negative examples. The approach presented here, on the other hand, allows the use of negative examples even when there is only one semantic class.

The focus of this work is on set-expansion from free text. Thus, we do not compare our system with systems which use textual sources other than free text (e.g. semi-structured web pages or query logs). The works of Sarmento et al. [1] and Pantel et al. [2] are the state-of-the-art works that are most related to our approach and, therefore, in our experiments, we compare their centroid-based approach with the approach developed here.

III. Centroid-Based Approach to Set-Expansion

A. Feature Vector Generation

The input to our set-expansion system consists of free text. To extract the relevant entities from the text, we preprocess the corpus with a state-of-the-art Named Entity Recognition tool developed by Ratinov and Roth[1] [11]. Our experiments were done only for entities of type *PER*, and we denote the set of all distinct entities recognized in the corpus by E. e_i represents the i^{th} entity. The features of an entity e_i are composed of the words appearing in a window of size W centered on each mention of the entity e_i. We use discounted PMI [12] to measure the association of a feature with the entity. Table II gives some of the features for two different entities as generated from the corpus. The numbers along with the features indicate the absolute frequency of the feature.

B. List Generation

We compute the similarity between any two entities using the cosine coefficient. Given an entity e_i, we compute the similarity between e_i and all other entities in the entity set E. We then sort all the entities in E based on this similarity score in decreasing order. The resulting ranked list has the property that entities with lower rank are more similar to e_i than entities with higher rank. We call this list the set of neighbors of e_i, denoted as *NBRLIST(e_i)*.

In the centroid-based approach, first of all, centroid ($\mathcal{C}$) is computed by averaging the frequency vectors of entities in the seed-set ($\mathcal{S}$) and then computing the discounted PMI of the resulting frequency vector. Next, *NBRLIST* of the centroid is computed as described above and the system outputs the first M members of *NBRLIST*.

IV. Learning from Negative Examples in Centroid-Based Approach

The centroid-based approach to set-expansion doesn't easily allow learning from negative examples. In this section, we present a novel framework which allows the incorporation of negative examples in a centroid-based approach.

The active features of any entity are those features which have non-zero frequency. The active features of the centroid are the union of the active features of the entities in the seed-set. The active features of the centroid are not equally important. To incorporate this knowledge into set-expansion, we associate a weight term with each entry in the vocabulary. Higher weight would mean that a particular word is more relevant to the underlying concept. By incorporating these weights into the cosine similarity metric, the new formula to compute the similarity between entities e_i and e_j becomes:

$$wsim_{ij} = \frac{\sum_k w_k dpmi_{ik} dpmi_{jk}}{\sqrt{\sum_k dpmi_{ik}^2}\sqrt{\sum_k dpmi_{jk}^2}} \tag{1}$$

where w_k is the weight associated with the k^{th} word and *dpmi* refers to the discounted PMI values. We wish to learn a weight vector w such that the similarity between the positive examples and the centroid becomes more than a prespecified threshold ζ. Moreover, we want that the similarity between negative examples and the centroid should become less than a prespecified threshold ϑ. We accomplish this objective using the following linear program:

$$\begin{aligned} max \quad & \sum_{e_i \in PositiveExamples} wsim_{\mathcal{C}i} \\ & - \sum_{e_j \in NegativeExamples} wsim_{\mathcal{C}j} \end{aligned} \tag{2}$$

$$\text{s.t.} \sum_k w_k \leq \text{num of non-zero entries in centroid} \tag{3}$$

$$wsim_{\mathcal{C}i} \geq \zeta \quad \forall e_i \in PositiveExamples \tag{4}$$

$$wsim_{\mathcal{C}i} \leq \vartheta \quad \forall e_i \in NegativeExamples \tag{5}$$

$$w_k \geq 0 \qquad \forall k \tag{6}$$

$$w_k \leq \lambda \qquad \forall k \tag{7}$$

[1] http://cogcomp.cs.illinois.edu/page/software

Entity	Examples of Feature Vectors
Bill Clinton	[President, 24912], [administration, 790], [House, 766], [visit, 761], [talks, 742], [announced, 737], [summit, 703], [White, 684], [Republican, 541], [WASHINGTON, 508], [Congress, 490], [Democratic, 318], [budget, 243], [veto, 230], [government, 219], [election, 192], [political, 182], [Hillary, 149]
Pete Sampras	[USA, 323], [World, 254], [champion, 226], [number-one, 191], [defending, 124], [final, 115], [American, 112], [pts, 99], [beat, 86], [round, 81], [tennis, 73], [singles, 65], [Wimbledon, 62], [seeded, 40], [lost, 39], [semi-final, 38], [Grand, 36], [Slam, 36], [tournament, 34], [top-seed, 32], [Tennis, 5]

Table II: This table shows some of the features for two different entities. The numbers along with the features indicate the absolute frequency of the feature appearing with the entity under consideration.

In the above linear program, Equation (2) is the objective of the linear program which aims at

1) maximizing the similarity between positive examples and the centroid and
2) minimizing the similarity between negative examples and the centroid.

Note that $\mathcal{C}$ refers to centroid in Equations (2), (4) and (5).

V. Inference-Based Approach to Set-Expansion

The centroid-based approach to set-expansion has some limitations. In the centroid-based approach, the centroid is supposed to fully represent the underlying concept, and all the similarity scores are computed with respect to the centroid. There is no way to confirm the decisions made with respect to the centroid. There is a lot of information in the individual positive and negative examples which is not exploited in the centroid-based approach. Moreover, as more and more positive examples are added to the seed-set, the number of active features of the centroid keeps increasing and this may lead to over-generalization.

In this section, we present an inference-based method for set-expansion. Unlike Section III, we do not compute the centroid of the positive examples. The new approach is based on the intuition that the positive and negative examples can complement each others' decision to better represent the underlying concept. Each example can be thought of as an expert which provides positive or negative evidence regarding the membership of any entity in the underlying concept. We develop a mechanism to combine the suggestions of such experts.

Algorithm 1 gives the procedure for inference-based set expansion. In steps 1 and 2, we compute the *NBRLIST* of positive and negative examples respectively (see Section III-B). Entities which have high similarity to the positive examples are more likely to belong to the underlying concept, while entities which have high similarity to the negative examples are likely to *not* belong to the underlying concept.

Steps 1 and 2 of Algorithm 1 generate one list corresponding to every positive and negative example. We associate a reward (or penalty) with each entity in these lists based on the rank of the entity. Our reward (or penalty) function is based on the effective length, $\mathcal{L}$, of a list. The entities which have higher rank than the effective length of the list are given a reward (or penalty) of zero.

Algorithm 1: InferenceBasedSetExpansion

Input : E (Entity Set), W (List of positive examples), B (List of negative examples)
Output: L (Ranked List)
begin

1 *Compute NBRLISTs of Positive Examples*
for $j \leftarrow 1$ **to** $|W|$ **do**
  for $i \leftarrow 1$ **to** $|E|$ **do**
    $WSV_j[i] \longleftarrow sim_{iw_j}$
  Sort the entities in E based on WSV_j and store the result in WL_j

2 *Compute NBRLISTs of Negative Examples*
for $j \leftarrow 1$ **to** $|B|$ **do**
  for $i \leftarrow 1$ **to** $|E|$ **do**
    $BSV_j[i] \longleftarrow sim_{ib_j}$
  Sort the entities in E based on BSV_j and store the result in BL_j

3 *Initialize the score for each entity to* 0
for $i \leftarrow 1$ **to** $|E|$ **do**
  $score_i \longleftarrow 0$

4 *Compute the contribution from positive examples*
for $j \leftarrow 1$ **to** $|WL|$ **do**
  for $i \leftarrow 1$ **to** $|E|$ **do**
    $e \longleftarrow WL_j[i]$
    $score_e \longleftarrow$
    $score_e + reward(i, 0) + WSV_j[e]$

5 *Compute the contribution from negative examples*
for $j \leftarrow 1$ **to** $|BL|$ **do**
  for $i \leftarrow 1$ **to** $|E|$ **do**
    $e \longleftarrow BL_j[i]$
    $score_e \longleftarrow score_e + reward(i, 1)$

6 $L \longleftarrow$ List of entities in E sorted by $score$

Effective length, $\mathcal{L}$, of a list is computed by multiplying the required list length (or cut-off) by a list factor, $\mathcal{F}$. If M is the specified cut-off, then $\mathcal{L} = M \times \mathcal{F}$. The reward is calculated according to the following equation:

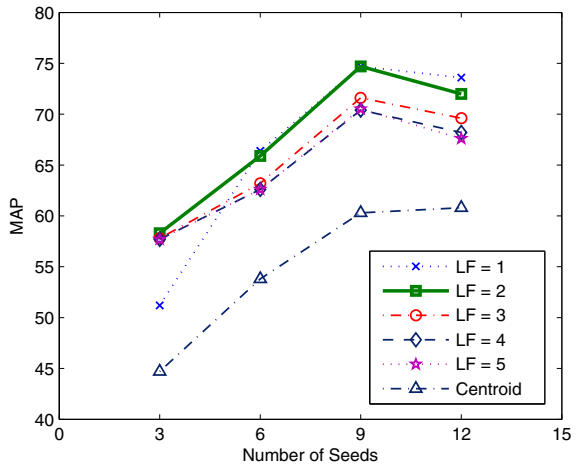


Figure 1: This figure shows the effect of list factor ($\mathcal{F}$) on the performance of set-expansion. When averaged across different number of seeds, $\mathcal{F} = 2$ gave the best results.

$$reward(r, n) = \begin{cases} (-1)^n \times \mathcal{L} \times a & \text{if } r = 1 \\ (-1)^n \times (\mathcal{L} - r) \times b & \text{if } 1 < r \leq \mathcal{L} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

In the above equation, r refers to the rank of the entity in the list. n is set to 0 for lists corresponding to positive examples and to 1 for lists corresponding to negative examples. Thus, for lists corresponding to negative examples, the reward is negative and hence, acts like a penalty. The values of all the parameters were determined empirically. Equation (8) gives higher reward or penalty to the entities with lower rank.

Figure 1 shows the effect of $\mathcal{F}$ on the Mean Average Precision (MAP) (please see Section VI for a discussion on MAP) for the concept of female tennis players as the number of seeds is varied. Only positive examples were used for generating Figure 1. To find the best value of $\mathcal{F}$, we take the average of MAP across different number of seeds. We find that $\mathcal{F} = 2$ has the highest average of 67.7. Although $\mathcal{F} = 1$ gives good performance for higher number of seeds, its performance is quite low when the number of seeds is small. As we increase the value of $\mathcal{F}$ beyond 2, the performance goes on decreasing. We used $\mathcal{F} = 2$ for all our experiments.

Steps 3, 4 and 5 in Algorithm 1 compute the *score* for each entity in the entity-set E. Step 3 initializes the *score* for each entity to 0. Steps 4 and 5 compute the contributions from the lists corresponding to the positive and negative examples, respectively, towards the score of entities. If $rank(i, j)$ denotes the rank of entity e_i in the list corresponding to example e_j, then the final score of entity e_i can be written as:

Attribute	Value
Number of Files	44
Total Size	1,216 MB
Number of Docs	656,269
Total Tokens	170,969,000
Vocabulary Size	548,862
Distinct Entities (PER)	386,209

Table III: Characteristics of AFE section of GCOR

$$score_i = \sum_{w \in W} [reward(rank(i, w), 0) + sim_{iw}] + \sum_{b \in B} [reward(rank(i, b), 1)] \quad (9)$$

In the above equation, W and B refer to the list of positive and negative examples, respectively. Finally, step 6 of Algorithm 1 sorts the entities in descending order based on the final score as computed in Equation (9). The first M members of the resulting list are output by the system. Here, M is the required list length.

VI. Experiments

We used the AFE section of English Gigaword Corpus (henceforth referred to as *GCOR*) for our experiments. This is a comprehensive archive of newswire text data in English that is available from LDC. The characteristics of the corpus are shown in Table III.

The parameters of linear program (Eqns (2)-(7)) and reward function (Eqn (8)) were determined empirically. ζ, ϑ, λ, a and b were set to 0.2, 0.0001, 10, 100 and 10 respectively.

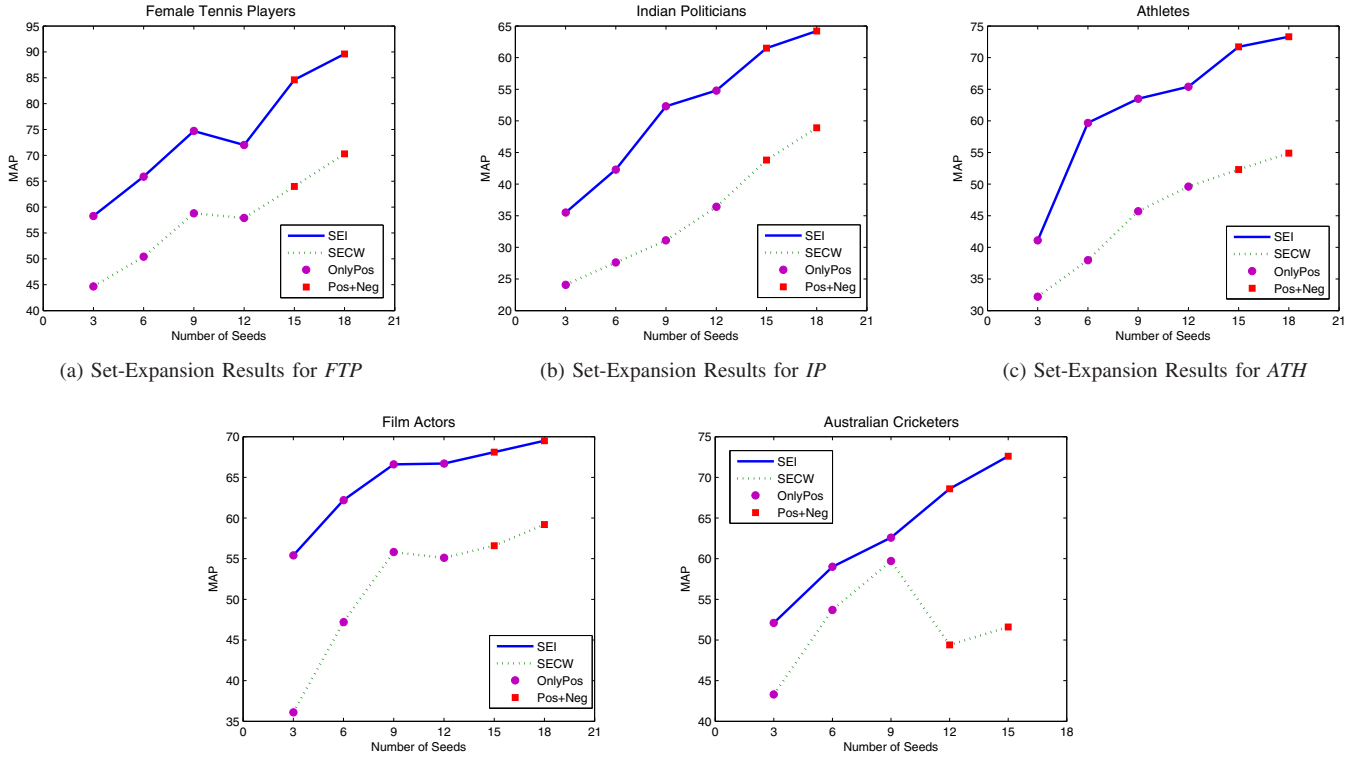
A. Inference vs Centroid Based Approaches

We use the following notation to refer to the three set-expansion systems that we present and compare:

1) *SEC* - Set Expansion system using Centroid. This is the current state-of-the-art [1], [2] and was presented in Section III. This system can't learn from the negative examples.
2) *SECW* - Set Expansion system using Centroid where Weights are associated with the vocabulary terms. This system was developed and explained in Section IV. This system can learn from negative examples.
3) *SEI* - Set Expansion system using Inference. This is the new approach to set-expansion and it was developed and explained in Section V.

SEC and SECW serve as baseline systems. Table IV compares the performance of SEI with the two baselines on 5 different concepts as mentioned below:

1) Female Tennis Players (FTP)
2) Indian Politicians (IP)
3) Athletes (ATH)



(a) Set-Expansion Results for *FTP*

(b) Set-Expansion Results for *IP*

(c) Set-Expansion Results for *ATH*

(d) Set-Expansion Results for *FA*

(e) Set-Expansion Results for *AC*

Figure 2: This figure shows the MAP values for 5 different concepts for both SEI and SECW (Baseline). Two things can immediately be noted from the graphs: (1) Negative examples significantly improve the MAP values for both SEI and SECW. (2) SEI performs much better than SECW for all the five concepts.

Concept	SEC	SECW	SEI
FTP	57.9	70.3	**89.6**
IP	36.4	48.9	**64.2**
ATH	49.6	54.9	**73.3**
FA	55.1	59.2	**69.5**
AC	59.7	51.6	**72.6**

Table IV: *A comparison of the MAP values of SEI with the 2 baselines on 5 different concepts*. Our algorithm, SEI, performs significantly better than both baselines on all the concepts. SECW is our improvement to the centroid method and is the second best. It performs better than SEC (current state-of-the-art) on all concepts except AC. For details, please refer to Section VI-A.

4) Film Actors (FA)
5) Australian Cricketers (AC)

The evaluation metric used in Table IV and in later experiments is Mean Average Precision (MAP). MAP is the mean value of average precision (AP) computed for each ranked list separately. For a list of length $\mathcal{M}$, the AP is given by the following equation:

$$AP = \frac{\sum_{r=1}^{\mathcal{M}}[P(r) \times rel(r)]}{\#\text{TrueEntities}} \tag{10}$$

In the above equation, *r* is the rank, *rel* is a binary function on the relevance of a given rank and *P(r)* is the precision at given cutoff rank.

For the results presented in Table IV, we used 12 positive and 6 negative examples for each concept other than AC. For AC, we used 9 positive and 6 negative examples.

Table IV clearly shows that SEI does much better than both the baselines on all concepts. We observed that the lists produced by SEI hardly contained any errors in the first half of the lists as we also saw in Table I. This is because the entities which come in the beginning of the list in SEI get high scores from several positive examples and are *not* penalized by any of the negative examples. On the other hand, SEC and SECW are unable to make such inferences.

We also see from Table IV that with the exception of AC, SECW performs better than SEC on all concepts. The better performance of SECW is due to the use of negative examples. Thus, the strategy developed in Section IV for incorporating negative examples is quite effective.

For further analysis, in Figure 2, we compare the performance of SEI with SECW on all the concepts as the seed-set size is increased. First we supplied only positive examples as indicated by the *circle* markers in Figure 2.

Desired Concept	Negative Examples
Female Tennis Players	Male Tennis Players
Film Actors	Musicians, Film Directors
Athletes	Football Platers, Skiers
Indian Politicians	Other Politicians
Australian Cricketers	Cricketers from other countries

Table V: This table shows the negative examples that were used for different concepts. We see that the negative examples are closely related to the instances of the desired concept.

After supplying sufficient number of positive examples, we provided negative feedback on 6 examples as indicated by *square* markers in Figure 2. For clarity, we do not show the performance of SEC in Figure 2. SEC performs similarly to SECW on positive examples but is unable to learn from negative examples.

Two conclusions can readily be drawn from Figure 2:

1) Negative examples significantly improve the performance of set-expansion for both SEI and SECW.
2) SEI performs much better than SECW on all the concepts irrespective of the seed-set size.

Table V categorizes the negative examples that were used for different concepts. We see that the good negative examples are closely related to the true instances of the desired concept. For example, the negative examples for the concept *Australian Cricketers* consist of the cricket players from other countries. We can see from Figure 2 that for SEI, the negative examples improve the MAP for FTP, IP, ATH, FA and AC by 17.6%, 9.4%, 7.9%, 2.8% and 10.0% respectively.

VII. Conclusions

This paper showed that negative examples can significantly improve the performance of set-expansion by helping to better define the underlying concept. We incorporated weights in the commonly used centroid-based approach so that it can benefit from negative examples. We developed a new, inference-based approach to set-expansion which naturally allows for negative examples and showed that it performs significantly better than the centroid-based approach.

Acknowledgments

This research was supported by Grant HHS 90TR0003/01 and by the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the HHS, DARPA, AFRL, or the US government.

References

[1] L. Sarmento, V. Jijkuon, M. de Rijke, and E. Oliveira, "More like these: growing entity classes from seeds," in *Proceedings of the sixteenth ACM conference on CIKM*. ACM, 2007, pp. 959–962.

[2] P. Pantel, E. Crestan, A. Borkovsky, A. Popescu, and V. Vyas, "Web-scale distributional similarity and entity set expansion," in *Proceedings of the 2009 Conference on EMNLP*. ACL, 2009, pp. 938–947.

[3] R. Wang and W. Cohen, "Language-independent set expansion of named entities using the web," in *ICDM*. IEEE Computer Society, 2007, pp. 342–350.

[4] E. Riloff and R. Jones, "Learning dictionaries for information extraction by multi-level bootstrapping," in *Proceedings of the National Conference on Artificial Intelligence*. JOHN WILEY & SONS LTD, 1999, pp. 474–479.

[5] P. Talukdar, T. Brants, M. Liberman, and F. Pereira, "A context pattern induction method for named entity extraction," in *Proceedings of the Tenth Conference on CoNLL*. ACL, 2006, pp. 141–148.

[6] P. Talukdar, J. Reisinger, M. Paşca, D. Ravichandran, R. Bhagat, and F. Pereira, "Weakly-supervised acquisition of labeled class instances using graph random walks," in *Proceedings of the Conference on EMNLP*. ACL, 2008, pp. 582–590.

[7] M. Pennacchiotti and P. Pantel, "Entity extraction via ensemble semantics," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 2009, pp. 238–247.

[8] M. Pasca and B. Van Durme, "Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs," in *Proceedings of the 46th Annual Meeting of the ACL (ACL-08)*. Citeseer, 2008, pp. 19–27.

[9] M. Thelen and E. Riloff, "A bootstrapping method for learning semantic lexicons using extraction pattern contexts," in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 2002, pp. 214–221.

[10] W. Lin, R. Yangarber, and R. Grishman, "Bootstrapped learning of semantic classes from positive and negative examples," in *Proceedings of ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, vol. 1, no. 4, 2003, p. 21.

[11] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2009, pp. 147–155.

[12] P. Pantel and D. Lin, "Discovering word senses from text," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002, pp. 613–619.