Class Label Enhancement via Related Instances

Zornitsa Kozareva
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
kozareva@isi.edu

Konstantin Voevodski
Boston University
111 Cummington St.
Boston, MA, 02215
kvodski@bu.edu

Shang-Hua Teng
University of Southern California
941 Bloom Walk, SAL 300
Los Angeles, CA 90089
shanghua@usc.edu

Abstract

Class-instance label propagation algorithms have been successfully used to fuse information from multiple sources in order to enrich a set of unlabeled instances with class labels. Yet, nobody has explored the relationships between the instances themselves to enhance an initial set of class-instance pairs. We propose two graph-theoretic methods (centrality and regularization), which start with a small set of labeled class-instance pairs and use the instance-instance network to extend the class labels to all instances in the network. We carry out a comparative study with state-of-the-art knowledge harvesting algorithm and show that our approach can learn additional class labels while maintaining high accuracy. We conduct a comparative study between class-instance and instance-instance graphs used to propagate the class labels and show that the latter one achieves higher accuracy.

1 Introduction

Many natural language processing applications use and rely on semantic knowledge resources. Since manually built lexical repositories such as Word-Net (Fellbaum, 1998) cover a limited amount of knowledge and are tedious to maintain over time, researchers have developed algorithms for automatic knowledge extraction from structured and unstructured texts. There is a substantial body of work on extracting is-a relations (Etzioni et al., 2005; Kozareva et al., 2008), part-of relations (Girju et al., 2003; Pantel and Pennacchiotti, 2006) and general facts (Lin and Pantel, 2001; Davidov and Rappoport,

2009; Jain and Pantel, 2010). The usefulness of the generated resources has been shown to be valuable to information extraction (Riloff and Jones, 1999), question answering (Katz et al., 2003) and textual entailment (Zanzotto et al., 2006) systems.

Among the most common knowledge acquisition approaches are those based on lexical patterns (Hearst, 1992; Etzioni et al., 2005; Kozareva et al., 2008) and clustering (Lin and Pantel, 2002; Davidov and Rappoport, 2008). While clustering can find instances and classes that are not explicitly expressed in text, they often may not generate the granularity needed by the users. In contrast, pattern-based approaches generate highly accurate lists, but they are constraint to the information matched by the pattern and often suffer from recall. (Paşca, 2004; Snow et al., 2006; Kozareva and Hovy, 2010) have shown that complete lists of semantic classes and instances are valuable for the enrichment of existing resources like WordNet and for taxonomy induction. Therefore, researchers have focused on the development of methods that can automatically augment the initially extracted class-instance pairs.

(Pennacchiotti and Pantel, 2009) fused information from pattern-based and distributional systems using an ensemble method and a rich set of features derived from query logs, web-crawl and Wikipedia. (Talukdar et al., 2008) improved class-instance extractions exploring the relationships between the classes and the instances to propagate the initial class-labels to the remaining unlabeled instances. Later on (Talukdar and Pereira, 2010) showed that class-instance extraction with label propagation can be further improved by adding semantic information

118

in the form of instance-attribute edges derived from independently developed knowledge base. Similarly to (Talukdar et al., 2008) and (Talukdar and Pereira, 2010), we are interested in enriching class-instance extractions with label propagation. However, unlike the previous work, we model the relationships between the instances themselves to propagate the initial set of class labels to the remaining unlabeled instances. To our knowledge, this is the first work to explore the connections between instances for the task of class-label propagation.

Our work addresses the following question: *Is it possible to effectively explore the structure of the text-mined instance-instance networks to enhance an incomplete set of class labels?* Our intuition is that if an instance like *bear* belongs to a semantic class *carnivore*, and the instance *bear* is connected to the instance *fox*, then it is more likely that the unlabeled instance *fox* is also of class *carnivore*. To solve this problem, we propose two graph-based approaches that use the structure of the *instance-instance* graph to propagate the class labels. Our methods are agnostic to the sources of semantic instances and classes. In this work, we carried out experiments with a state-of-the-art instance extraction system and conducted a comparative study between the original and the enhanced class-instance pairs. The results show that this labeling procedure can begin to bridge the gap between the extraction power of the pattern-based approaches and the desired recall by finding class-instance pairs that are not explicitly mentioned in text. The contributions of the paper are as follows:

- We use only the relationships between the instances themselves to propagate class labels.

- We observe how often labels are propagated along the edges of our semantic network, and propose two ways to extend an initial set of class labels to all the instance nodes in the network. The first approach uses a linear system to compute the network centrality relative to the initially labeled instances. The second approach uses a regularization framework with respect to a random walk on the network.

- We evaluate the proposed approaches and show that they discover many new class-instance pairs compared to state-of-the-art knowledge

harvesting algorithm, while still maintaining high accuracy.

- We conduct a comparative study between class-instance and instance-instance graphs used to propagate class labels. The experiments show that considering relationships between instances achieves higher accuracy.

The rest of the paper is organized as follows. In Section 2, we review related work. Section 3 describes the Web-based knowledge harvesting algorithm used to extract the instance network and the class-instance pairs necessary for our experimental evaluation. Section 4 describes the two graph-theoretic methods for class label propagation using an instance-instance network. Section 5 shows a comparative study between the proposed graph algorithms and different baselines. We also show a comparison between class-instance and instance-instance graphs used in the label propagation. Finally, we conclude in Section 6.

2 Related Work

In the past decade, we have reached a good understanding on the knowledge harvesting technology from structured (Suchanek et al., 2007) and unstructured text. Researchers have harvested with varying success semantic lexicons (Riloff and Shepherd, 1997) and concept lists (Katz et al., 2003). Many efforts have also focused on the extraction of is-a relations (Hearst, 1992; Paşca, 2004; Etzioni et al., 2005; Paşca, 2007; Kozareva et al., 2008), part-of relations (Girju et al., 2003; Pantel and Pennacchiotti, 2006) and general facts (Etzioni et al., 2005; Davidov and Rappoport, 2009; Jain and Pantel, 2010). Various approaches have been proposed following the patterns of (Hearst, 1992) and clustering (Lin and Pantel, 2002; Davidov and Rappoport, 2008). A substantial body of work has explored issues such as reranking the harvested knowledge using mutual information (Etzioni et al., 2005) and graph algorithms (Hovy et al., 2009), estimating the goodness of text-mining seeds (Vyas et al., 2009), organizing the extracted information (Cafarella et al., 2007a; Cafarella et al., 2007b) and inducing term taxonomies with WordNet (Snow et al., 2006) or starting from scratch (Kozareva and Hovy, 2010).

Since pattern-based approaches tend to be high-precision and low-recall in nature, recently of great interest to the research community is the development of approaches that can increment the recall of the harvested class-instance pairs. (Pennacchiotti and Pantel, 2009) proposed an ensemble semantic framework that mixes distributional and pattern-based systems with a large set of features from a web-crawl, query logs, and Wikipedia. (Talukdar et al., 2008) combined extractions from free text and structured sources using graph-based label propagation algorithm. (Talukdar and Pereira, 2010) conducted a comparative study of graph algorithms and showed that class-instance extraction can be improved using additional information that can be modeled as instance-attribute edges.

Closest to our work is that of (Talukdar et al., 2008; Talukdar and Pereira, 2010) who model class-instance relations to propagate class-labels. Although these algorithms can be applied to other relations (Alfonseca et al., 2010), to our knowledge yet nobody has modeled the connections between the instances themselves for the task of class-label propagation. We propose regularization and centrality graph-theoretic methods, which exploit the instance-instance network and a small set of class-instance pairs to propagate the class-labels to the remaining unlabeled instances. While objectives similar to regularization have been used for class-label propagation, the application of node centrality for this task is also novel. The proposed solutions are intuitive and almost parameter-free (both methods have a single parameter, which is easy to interpret and does not require careful tuning).

3   Knowledge Harvesting from the Web

Our proposed class-label enhancement approaches are agnostic to the sources of semantic instances and classes. Several methods have been developed to harvest instances from the Web (Paşca, 2004; Etzioni et al., 2005; Paşca, 2007; Kozareva et al., 2008) and potentially we can use any of them. In our experiments, we use the doubly-anchored (DAP) method of (Kozareva et al., 2008), because it achieves higher precision than (Etzioni et al., 2005; Paşca, 2007), it is easy to implement and requires minimum supervision (only one seed instance and a lexico-syntactic pattern).

For a given semantic class of interest say *animals*, the algorithm starts with a *seed* example of the class, say *whales*. The *seed* instance is fed into a doubly-anchored pattern "<*semantic-class*> such as <*seed*> and *", which extracts on the position of the * new instances of the semantic class. Then, the newly acquired instances are individually placed on the position of the *seed* in the DAP pattern. The bootstrapping procedure is repeated until no new instances are found. We use the harvested instances to build the instance-instance graph in which the nodes are the learned instances and directed edges like (*whales*,*dolphins*) indicate that the instance *whales* extracted the instance *dolphins*. The edges between the instances are weighted based on the number of times the DAP pattern extracted the instances together.

Different strategies can be employed to acquire semantic classes for each instance. We follow the fully automated approach of (Hovy et al., 2009), which takes the learned instance pairs from DAP and feeds them into the pattern "* such as <*instance$_1$*> and <*instance$_2$*>". The algorithm extracts on the position of the * new semantic classes related to *instance$_1$*. According to (Hovy et al., 2009), the usage of two instances acts as a disambiguator and leads to much more accurate semantic class extraction compared to (Ritter et al., 2009).

4   Methods

We model the output of the instance harvesting algorithm as a directed weighted graph that is given by a set of vertices V and a set of edges E. We use n to denote the number of vertices. A node u corresponds to a learned instance, and an edge $(u, v) \in E$ indicates that the instance v was learned from the instance u using the *DAP* pattern. The weight of the edge $w(u, v)$ specifies the number of times the pair of instances were found by the *DAP* pattern. We define the adjacency matrix of the graph as:

$$A(u,v) = \begin{cases} w(u,v) & \text{if } (u,v) \in E \\ 0 & \text{otherwise.} \end{cases}$$

We use $d_{\text{out}}(u)$ to specify the out-degree of u: $d_{\text{out}}(u) = \sum_{(u,v) \in E} w(u,v)$, and $d_{\text{in}}(v)$ to specify the in-degree of v: $d_{\text{in}}(v) = \sum_{(u,v) \in E} w(u,v)$.

We represent the initial set of instances L that are believed to belong to class C (the set of *labeled* instances) by a row vector $l \in \{0,1\}^n$, where $l(u) = 1$ if $u \in L$. Our objective is to compute a vector $\hat{l}$ where $\hat{l}(u)$ is proportional to how likely it is that u belongs to C. We write all vectors as row vectors, and use $\vec{c}$ to denote a 1 by n *constant* vector such that $\vec{c}(u) = c$ for all $u \in V$.

4.1 Personalized Centrality

Our first approach is based on the intuition that if $u \in C$ and $(u,v) \in E$, then it is more likely that $v \in C$. Moreover, the larger the weight of the edge $w(u,v)$, the more likely it is that $v \in C$. When we extend this intuition to all the in-neighbors, we say that the score of each node is proportional to the sum of the scores of its in-neighbors scaled by the edge weights: $\hat{l}(v) = \alpha \sum_{(u,v) \in E} \hat{l}(u) w(u,v)$. We can verify that the vector $\hat{l}$ must then satisfy $\hat{l} = \alpha \hat{l} A$, so it is an eigenvector of the adjacency matrix of the graph with an eigenvalue of α.

However, this formulation is insufficient because even though it captures our intuition that the nodes get their scores from their in-neighbors, we are still ignoring the initial scores of the nodes. A way to take the initial scores into consideration is to compute the following steady-state equation:

$$\hat{l} = l + \alpha \cdot \hat{l} A. \tag{1}$$

Equation 1 specifies that the score $\hat{l}(u)$ of each node u is the sum of its initial score $l(u)$ and the weighted sum of the scores of its neighbors, which is scaled by α. This equation is known as α-centrality, which was first introduced by (Bonacich and Lloyd, 2001). The α parameter controls how much the score of each node depends on the scores of its neighbors. When $\alpha = 0$ the score of each node is equivalent to its initial score, and does not depend on the scores of its neighbors at all.

Alternately, we can think of the vector $\hat{l}$ as the fixed-point of the process in which in each iteration some node v updates its score $\tilde{l}(v)$ by setting $\tilde{l}(v) = l(v) + \alpha \sum_{(u,v) \in E} w(u,v) \tilde{l}(u)$.

Solving Equation 1 we can see that $\hat{l} = l(I - \alpha A)^{-1}$, where I is the identity matrix of size n. The solution is also closely related to the following expression, which is known as a Katz score (Katz, 1953):

$$s \sum_{t=1}^{\infty} \alpha^t A^t.$$

We can verify that $A^t(u,v)$ gives the number of paths of length t between u and v. Katz proposed using the above expression with the starting vector $s = \vec{1}$ to measure centrality in a network. Therefore, the score of node v is given by the number of paths from u to v for all $u \in V$, with longer paths given less weight based on the value of α. The method proposed here measures a similar quantity with a non-uniform starting vector. To show the relationship between the two measures we use the identity that $\sum_{t=1}^{\infty} \alpha^t A^t = (I - \alpha A)^{-1} - I$. It is easy to see that

$$\begin{aligned}
\hat{l} &= l(I - \alpha A)^{-1} \\
&= l(\textstyle\sum_{t=1}^{\infty} \alpha^t A^t + I) \\
&= l \textstyle\sum_{t=1}^{\infty} \alpha^t A^t + l \\
&= l \textstyle\sum_{t=0}^{\infty} \alpha^t A^t.
\end{aligned} \tag{2}$$

Equation 2 shows that $\hat{l}(v)$ is given by the number of paths from u to v for all $u \in L$ (the initial labeled set). Using a larger value of α corresponds to giving more weight to paths of longer length. The summation $\sum_{t=0}^{\infty} \alpha^t A^t$ converges as long as $|\alpha| < 1/\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of A. Therefore, we can only consider values of α in this range.

4.2 Regularization Using Random Walks

Our second approach constrains $\hat{l}$ to be as consistent or *smooth* as possible with respect to the structure of the graph. The simplest way to express this is to require that for each edge $(u,v) \in E$, the scores of the endpoints $\hat{l}(u)$ and $\hat{l}(v)$ must be as similar as possible. Moreover, the greater the weight of the edge $w(u,v)$ the more important it is for the scores to match. Using this intuition we can define the following optimization problem:

$$\text{argmin}_{\hat{l} \in \{0,1\}^n} \sum_{(u,v) \in E} (\hat{l}(u) - \hat{l}(v))^2.$$

Setting $\hat{l} = \vec{0}$ or $\hat{l} = \vec{1}$ clearly optimizes this function, but does not give a meaningful solution. However, we can additionally constrain $\hat{l}$ by requiring that the initial labels cannot be modified, or more generally penalizing the discrepancy between $\hat{l}(u)$ and $l(u)$ for $u \in L$. The methods of (Talukdar and Pereira, 2010) optimize objective functions of this type.

Unlike the work of (Talukdar and Pereira, 2010), here we use an objective function that considers smoothness with respect to a *random walk* on the graph. Performing a random walk allows us to take more of the graph structure into account. For example, if nodes u and v are part of the same cluster then it is likely that the edge (u, v) is heavily traversed during the random walk, and should have a lot of probability in the stationary distribution of the walk. Simply considering the weight of the edge $w(u, v)$ gives us no such information. Therefore if our objective function requires the scores to be consistent with respect to the stationary probability of the edges in the random walk, we can compute scores that are consistent with the *clustering structure* of the graph.

Our semantic network is not strongly connected, so we must make some modifications to the random walk to ensure that it has a stationary distribution. Section 4.2.1 describes our random walk and how we compute the transition probability matrix P and its stationary probability distribution π. The definition of our objective function and the description of how it is optimized is given in Section 4.2.2.

4.2.1 Teleporting Random Walk

Formally, a random walk is a process where at each step we move from some node to one of its neighbors. The transition probabilities are given by edge weights, therefore the transition probability matrix W is the normalized adjacency matrix where each row sums to one:

$$W = D^{-1}A.$$

Here the D matrix is the degree matrix, which is a diagonal matrix given by

$$D(u, v) = \begin{cases} d_{\text{out}}(u) & \text{if } u = v \\ 0 & \text{otherwise.} \end{cases}$$

In our semantic network some nodes have no out-neighbors, so in order to compute W we first add a self-loop to any such node. In addition, we modify the random walk to *reset* at each step with nonzero probability β to ensure that it has a steady-state probability distribution. When the walk resets it jumps or *teleports* to any node in the graph with equal probability. The transition probability matrix of this process is given by

$$P = \beta K + (1 - \beta)W,$$

where K is an n by n matrix given by $K(u, v) = \frac{1}{n}$ for all $u, v \in V$. The stationary distribution π must satisfy $\pi = \pi P$. Equivalently π can be viewed as a solution to the following PageRank equation:

$$\pi = \beta s + (1 - \beta)\pi W.$$

Here the *starting* vector $s = \frac{1}{n}\vec{1}$ gives the probability distribution for where the walk transitions when it resets. In our computations we use a jump probability $\beta = 0.15$, which is standard for computations of PageRank. The stationary distribution π can be computed by either solving the PageRank equation or computing the eigenvector of P corresponding to the eigenvalue of 1.

4.2.2 Regularization

(Zhou et al., 2005) propose the following function to measure the smoothness of $\hat{l}$ with respect to the stationary distribution of the random walk:

$$\Omega(\hat{l}) = \frac{1}{2} \sum_{(u,v) \in E} \pi(u)P(u, v)\left(\frac{\hat{l}(u)}{\sqrt{\pi(u)}} - \frac{\hat{l}(v)}{\sqrt{\pi(v)}}\right)^2.$$

Here $\pi(u)P(u, v)$ gives the steady-state probability of traversing the edge (u, v), and $\pi(u)$ and $\pi(v)$ specify how much probability u and v have in the stationary distribution π. Zhou et al. point out that using this function gives better results than smoothness with respect to the edge weights, which can be formulated by replacing $\pi(u)p(u, v)$ with $w(u, v)$, and replacing $\pi(u)$ and $\pi(v)$ with $d_{\text{out}}(u)$ and $d_{\text{in}}(v)$, respectively. This observation is consistent with our intuition that considering a random walk takes more of the graph structure into account.

In addition to minimizing $\Omega(\hat{l})$, we also want $\hat{l}$ to be as close as possible to l, which gives the following optimization problem:

$$\text{argmin}_{\hat{l} \in \mathbb{R}^n}\{\Omega(\hat{y}) + \mu||\hat{l} - l||^2\}. \tag{3}$$

Here the $\mu > 0$ parameter specifies the tradeoff between the two terms: using a larger μ corresponds to placing more emphasis on agreement with the initial labels. (Zhou et al., 2005) show that this objective is optimized by computing

$$\hat{l} = (I - \gamma\Theta)^{-1}l, \tag{4}$$

where $\Theta = (\Pi^{1/2}P\Pi^{-1/2} + \Pi^{-1/2}P\Pi^{1/2})/2$, and $\gamma = 1/(1 + \mu)$. Π is a diagonal matrix given by

$$\Pi(u,v) = \begin{cases} \pi(u) & \text{if } u = v \\ 0 & \text{otherwise.} \end{cases}$$

Zhou et al. propose this approach for semi-supervised learning of labels on the graph, given an initial vector l such that $l(u) = 1$ if vertex u has the label, $l(u) = -1$ if u does not have the label, and $l(u) = 0$ if the vertex is unlabeled. They propose taking the sign of $\hat{l}(u)$ to classify u as positive or negative. Using our labeling procedure we do not have any negative examples, so our initial vector l is non-negative, resulting in a non-negative vector $\hat{l}$. This is not a problem because we can still interpret $\hat{l}(u)$ to be proportional to how likely it is that u has the label. Rather than trying different settings of μ, we directly vary γ, with a smaller γ placing more emphasis on agreement with initial labels.

5 Experimental Evaluation

5.1 Data Collection

For our experimental study, we select three widely used domains in the harvesting community (Etzioni et al., 2005; Paşca, 2007; Hovy et al., 2009; Kozareva and Hovy, 2010): *animals* and *vehicles*. For each domain we randomly selected different semantic classes, which resulted in 20 classes altogether. To generate the instance-instance semantic network, we use the harvesting procedure described in Section 3. For example, to learn instances associated with animals, we instantiate the bootstrapping algorithm with the semantic class *animals*, the seed instance *bears* and the pattern "*animals such as bears and **". We submitted the pattern as queries to Yahoo!Boss and collected new instances. We ranked the instances following (Kozareva et al., 2008) which resulted in 397 animal, 4471 plant and 1425 vehicle instances. Table 1 shows the number of nodes (instances) and directed edges for the constructed semantic networks.

class	#instances	#directed-edges
animals	397	2812
vehicles	1425	3191

Table 1: *Nodes & Edges in the Instance Network.*

Next, we use the harvested instances to automatically learn the semantic classes associated with them. For example, *bears* and *wolves* are *animals* but also *mammals*, *predators*, *vertebrates* among

others. The obtained class harvesting results are shown in Table 2. We indicate with *Inst(Hovy et al., 2009)* the number of instances in the semantic network that discovered the class during the pattern-based harvesting, and with *InstInWordNet* the number of instances in the semantic network belonging to the class according to WordNet.

ClassName	Inst(Hovy et al., 2009)	InstInWordNet
arthropods	12	50
carnivores	24	57
chordates	2	313
eutherians	3	193
insects	5	29
invertebrates	53	84
mammals	114	205
reptile	5	22
ruminants	14	34
ungulates	16	66
crafts	24	68
motor vehicles	27	127
self-propelled vehicles	36	145
vessels	11	36
wheeled vehicles	54	190

Table 2: *Learned & Gold Standard Class-Instances.*

We can see that the pattern-based approach of (Hovy et al., 2009) does not recover a lot of the class-instance relations present in WordNet. Because of this gap between the actual and the harvested class-instance pairs arises the objective of our work, which is to explore the relationships between the instances to propagate the initially learned class labels to the remaining unlabeled instances. To evaluate the performance of our approach, we use as a gold standard the WordNet class-instance mappings.

5.2 Testing Our Approach

Our approach is based on the intuition that given a labeled instance u of class C, and an instance v in our network, if there is an edge (u,v) then it is more likely that v has the label C as well. For example, if the instance *bears* is of class *vertebrates* and there is an edge between the instances *bears* and *wolves*, then it is likely that *wolves* are also *vertebrates*. Before proceeding with the instance-instance class-label propagation algorithms, first we study whether this intuition is correct.

Individually for each class label C, we construct a set T_C that contains all instances in the network belonging to C according to WordNet. Then we compute the probability that v belongs to C in WordNet

given that (u, v) is an edge in the instance network and u belongs to C in WordNet: $Pr_h = \Pr[v \in T_C \mid (u, v) \in E$ and $u \in T_C]$. We compare this to the background probability $Pr_b = \Pr[v \in T_C \mid u, v \in V$ and $u \in T_C]$, which gives the probability that v belongs to C in WordNet if it is chosen at random. In other words, if $Pr_h = 1$, this means that whenever u has the label C and (u, v) is an edge, then v is always labeled with C. If indeed this is the case, then a good classifier can simply take the initial set L and extend the labels to all nodes reachable from L in the semantic network. The larger the difference between Pr_h and Pr_b, the more information the links of the instance network carry for the task of label propagation. Table 3 shows the Pr_h and Pr_b values for each class.

CLASS	Pr_h	Pr_b
arthropods	.46	.12
carnivores	.49	.14
chordates	.95	.80
eutherians	.80	.49
insects	.31	.07
invertebrates	.74	.21
mammals	.82	.52
reptile	.27	.05
ruminants	.39	.08
ungulates	.60	.16
crafts	.07	.05
motor vehicles	.10	.09
self-propelled vehicles	.11	.10
vessels	.08	.02
wheeled vehicles	.13	.13

Table 3: *Learned & Gold Standard Class-Instances.*

This study verifies our intuition that using the relationships between the instances to extend a class label to the remaining unlabeled nodes is an effective approach to enhancing an incomplete set of initial labels.

5.3 Comparative Study

The objective of our work is given a set of initially labeled nodes L, to assign to each node a score that indicates how likely it is to belong to L. The simplest way to do this using the edges of the instance network is to say that a node that has more in-neighbors that have a certain label is more likely to have this label. We define the *in-neighbor* score $i(v)$ of a node v as $i(v) = |\{u \in V | (u, v) \in E$ and $u \in L\}|$. We expect that the higher the *in-neighbor* score of v, the more likely it is that v has

the label L. The *personalized centrality* method that we proposed generalizes this intuition to indirect neighbors (see Methods). Our *regularization using random walks* technique further explores the link structure of the instance network by considering a random walk on it (see Methods). We compare our approaches with a method that labels nodes at *random*. The expected accuracy for class C is given by $\frac{|T_C|}{n}$, where n is the number of nodes in the network, and T_C is the set containing all nodes that belong to C according to WordNet. In other words, given that there are 84 nodes in the network that are classified as *invertebrate* according to WordNet, and there are 397 nodes in total, if we choose any number of nodes at random our expected accuracy is 21%.

We evaluate the performance of our approaches against the WordNet gold standard and show the obtained results in Tables 4 and 5.

Invertebrates				
rank	centrality	regularization	in-neighbor	random
5	1.0	1.0	.80	.21
10	1.0	1.0	.70	.21
20	.95	1.0	.75	.21
50	.96	.98	.76	.21
100	.69	.73	.67	.21
Mammals				
rank	centrality	regularization	in-neighbor	random
5	.80	1.0	.80	.52
10	.90	1.0	.90	.52
20	.95	.95	.85	.52
50	.86	.96	.80	.52
100	.92	.92	.76	.52
Carnivores				
rank	centrality	regularization	in-neighbor	random
5	1.0	1.0	.80	.14
10	.80	.80	.60	.14
20	.80	.85	.55	.14
50	.50	.68	.48	.14
100	.41	.44	.41	.14

Table 4: *Accuracy @ Different Ranks.*

Table 4 shows the accuracy at rank R calculated as the number of correctly labeled instances with class C at rank R divided by the total number of instances with class C at rank R. Due to space limitation, we show detailed ranking only for three of the classes. We can see that using the semantic network significantly enhances our ability to learn class labels. Even the simple *in-neighbor* method produces results that are very significant compared to chance. Our *centrality* and *regularization* techniques further explore the structure of the semantic network to give

better predictions.

Table 5 shows the accuracy of the class label propagation algorithms for each class. For each class we consider the top k ranked nodes, where k is the number of instances that belong to this class according to WordNet. For example, the accuracy of centrality for *carnivores* is 80% showing that from the top 57 ranked animal instances, 80% belong to *carnivores*. In the final column we also report the performance of a label propagation algorithm that uses class-instance graph instead of an instance-instance graph. To build the graph we remove the edges between the instances and keep the class-instance mappings discovered by the harvesting algorithm of (Hovy et al., 2009). We use the modified adsorption algorithm (MAD) of (Talukdar et al., 2008), which is freely available from the Junto toolkit[1]. To rank the instances for each class label produced by Junto, we use the computed label scores as a ranking criteria and measure accuracy similarly to centrality and regularization.

class	Centrality	Regular.	Rand	MAD
arthropods	.50	.60	.12	.56
carnivores	.80	.85	.14	.44
chordates	.81	.83	.80	.79
eutherians	.54	.60	.49	.60
insects	.38	.52	.07	.17
invertebrates	.94	.96	.21	.64
mammals	.82	.90	.52	.63
reptile	.45	.55	.05	.14
ruminants	.41	.44	.08	.41
ungulates	.44	.61	.16	.32
crafts	.47	.56	.05	.35
motor vehicle	.45	.48	.09	.24
self-propelled vehicle	.49	.47	.10	.27
vessel	.33	.39	.02	.31
wheeled vehicle	.51	.52	.13	.33

Table 5: *Comparative Study.*

The obtained results show that for almost all cases the methods that use the structure of the instance network significantly outperform predictions that use the class-instance graph. This indicates that we can indeed learn a lot form the instance-instance relationships by exploring the structure of the instance network. Among all approaches *regularization* achieves the best results. We believe that regularization works well because it considers a random walk on the semantic graph, and within-cluster

___
[1] http://code.google.com/p/junto/

edges are traversed more often in a random walk. The regularization technique computes scores that are consistent with the clustering structure of the graph by requiring that the endpoints of highly traversed edges, which are likely in the same cluster, have similar scores (see Methods). Overall, *regularization* enhanced the original output generated by the pattern-based knowledge harvesting approach of (Hovy et al., 2009) with 1219 new class-instance pairs (75% additional information) while maintaining 61.87% accuracy.
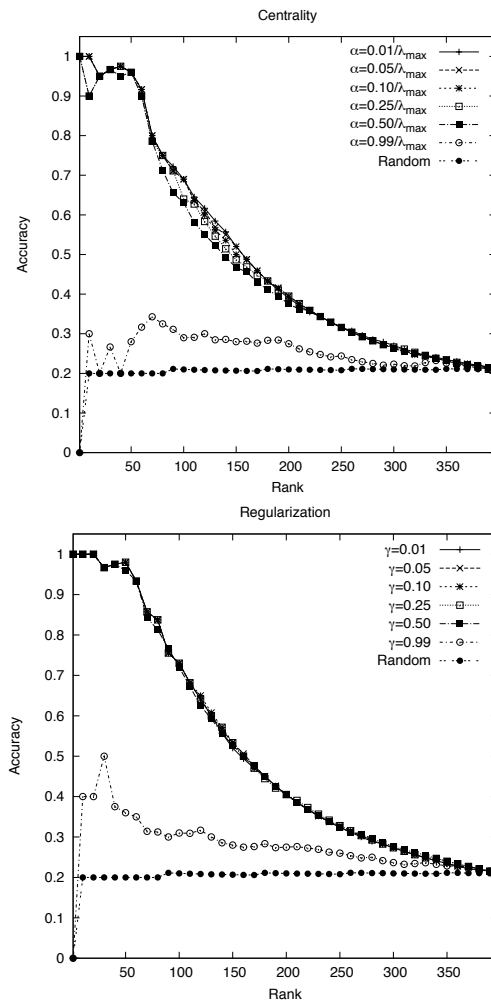

Figure 1: Parameter Tuning For **Invertebrates**.

5.4 Parameter Tuning

Both of our *centrality* and *regularization* methods have a single tunable parameter. For *centrality* the parameter α controls how much the label of each node depends on the labels of its neighbors in the

graph. The values range from 0 to $1/\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of the adjacency matrix of the semantic network. When $\alpha = 0$ the label of each node is equivalent to its initial label, while higher values of α give more weight to the labels of nodes that are further away.

For *regularization* the parameter γ controls how much emphasis is placed on the agreement between the initial and learned labels. The values of γ are between 0 and 1. Smaller values require that the learned labels be more consistent with the original labels. When $\gamma = 0$ the learned labels will exactly match the original labels.

For each method we try several parameter settings and show the results in Figure 1 for the propagation of the class label *invertebrate*. We can see that both methods are quite insensitive to the parameter settings, unless we choose very extreme values that ignore the original labels.

5.5 Effect of number of labeled class-instances

We also study how the quality of the results is affected by the number of initial class-instance pairs used by our propagation methods. We conduct experiments using only 25%, 50%, 75% and 100% of the initial class-instance pairs learned by (Hovy et al., 2009). Figure 2 shows the results for the label propagation of the class *invertebrate*.

The performance of our methods significantly improves when we incorporate more labels. Still, if we are less concerned with recall and want to find small sets of nodes with very high accuracy, the number of initial labels is less important. For example, starting with only 13 labeled nodes we can still achieve 100% accuracy for the top 30 nodes using *regularization*, and 96% accuracy for the top 25 nodes using *centrality*.

6 Conclusions

In this paper we proposed a centrality and regularization graph-theoretic methods that explore the relationships between the instances themselves to effectively extend a small set of class-instance labels to all instances in a semantic network. The proposed approaches are intuitive and almost parameter-free. We conducted a series of experiments in which we compared the effectiveness of the centrality and reg-
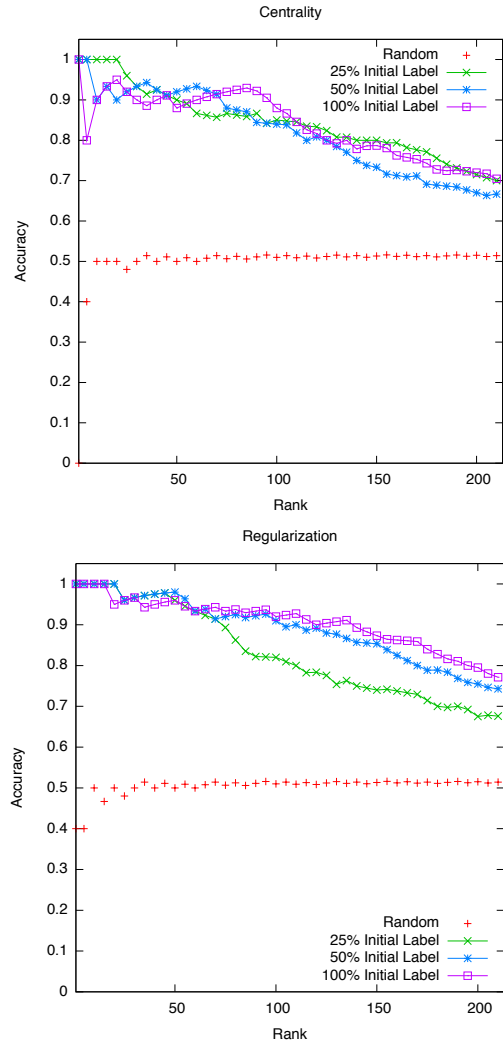

Figure 2: Effect of Number of Initial Class-Instance Pairs for **Invertebrates**.

ularization methods to learn new labels for the unlabeled instances. We showed that the enhanced class labels improve the original output generated by the pattern-based knowledge harvesting approach of (Hovy et al., 2009). Finally, we have studied the impact of the class-instance and instance-instance graphs for the class-label propagation task. The latter approach has shown to produce much more accurate results. In the future, we want to apply our approach to Web-based taxonomy induction, which according to (Kozareva and Hovy, 2010) is stifled due to the lacking relations between the instances and the classes, and the classes themselves. The proposed methods can be also applied to enhance fact

farms (Jain and Pantel, 2010).

Acknowledgments

References

Enrique Alfonseca, Marius Pasca, and Enrique Robledo-Arnuncio. 2010. Acquisition of instance attributes via labeled and related instances. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 58–65.

Phillip Bonacich and Paulette Lloyd. 2001. *Social Networks*, 23(3):191–201.

Michael J. Cafarella, Christopher R, Dan Suciu, Oren Etzioni, and Michele Banko. 2007a. Structured querying of web text: A technical challenge. In *in CIDR*.

Michael J. Cafarella, Dan Suciu, and Oren Etzioni. 2007b. Navigating extracted data with schema discovery. In *Tenth International Workshop on the Web and Databases, WebDB 2007WebDB*.

Dmitry Davidov and Ari Rappoport. 2008. Classification of semantic relationships between nominals using pattern clusters. In *Proceedings of ACL-08: HLT*, pages 227–235, June.

Dmitry Davidov and Ari Rappoport. 2009. Geo-mining: Discovery of road and transport networks using directional patterns. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP-09*, pages 267–275.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, 165(1):91–134, June.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 1–8.

Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545.

Eduard Hovy, Zornitsa Kozareva, and Ellen Riloff. 2009. Toward completeness in concept extraction and classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 948–957.

Alpa Jain and Patrick Pantel. 2010. Factrank: Random walks on a web of facts. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 501–509.

Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. 2003. Integrating web-based and corpus-based techniques for question answering. In *Proceedings of the twelfth text retrieval conference (TREC)*, pages 426–435.

Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika*, 18:39–43.

Zornitsa Kozareva and Eduard Hovy. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1110–1118.

Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics ACL-08: HLT*, pages 1048–1056.

Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 323–328.

Dekang Lin and Patrick Pantel. 2002. Concept discovery from text. In *Proc. of the 19th international conference on Computational linguistics*, pages 1–7.

Marius Paşca. 2004. Acquisition of categorized named entities for web search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 137–145.

Marius Paşca. 2007. Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In *Proceedings of the 16th international conference on World Wide Web*, pages 101–110.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 113–120.

Marco Pennacchiotti and Patrick Pantel. 2009. Entity extraction via ensemble semantics. In *Proceedings of the 2009 Conference on Empirical Methods in Natural*

Language Processing: Volume 1 - Volume 1, EMNLP '09, pages 238–247.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI '99/IAAI '99: Proceedings of the Sixteenth National Conference on Artificial intelligence.*

Ellen Riloff and Jessica Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Empirical Methods for Natural Language Processing*, pages 117–124.

Alan Ritter, Stephen Soderland, and Oren Etzioni. 2009. What is this, anyway: Automatic hypernym discovery. In *Proceedings of the AAAI Spring Symposium on Learning by Reading and Learning to Read.*

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 801–808.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Partha Pratim Talukdar and Fernando Pereira. 2010. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1473–1481.

Partha Pratim Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 582–590.

Vishnu Vyas, Patrick Pantel, and Eric Crestan. 2009. Helping editors choose better seed sets for entity set expansion. In *Proceeding of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 225–234.

Fabio Massimo Zanzotto, Marco Pennacchiotti, and Maria Teresa Pazienza. 2006. Discovering asymmetric entailment relations between verbs using selectional preferences. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 849–856.

Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2005. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 1036–1043.