Using Graph Based Method to Improve Bootstrapping Relation Extraction

Haibo Li, Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka

Graduate School of Information Science and Technology
University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
lihaibo@mi.ci.i.u-tokyo.ac.jp, danushka@iba.t.u-tokyo.ac.jp,
matsuo@biz-model.t.u-tokyo.ac.jp, ishizuka@i.u-tokyo.ac.jp

Abstract. Many bootstrapping relation extraction systems processing large corpus or working on the Web have been proposed in the literature. These systems usually return a large amount of extracted relationship instances as an out-of-ordered set. However, the returned result set often contains many irrelevant or weakly related instances. Ordering the extracted examples by their relevance to the given seeds is helpful to filter out irrelevant instances. Furthermore, ranking the extracted examples makes the selection of most similar instance easier. In this paper, we use a graph based method to rank the returned relation instances of a bootstrapping relation extraction system. We compare the used algorithm to the existing methods, relevant score based methods and frequency based methods, the results indicate that the proposed algorithm can improve the performance of the bootstrapping relation extraction systems.

1 Introduction

For many real world applications, background knowledge is intensively required. The acquisition of relational domain knowledge is still an important problem. Relation extraction systems extract structured relations from unstructured sources such as documents or web pages. These structured relations are as useful as knowledge. Acquiring relational facts *Acquirer–Acquiree* relation or *Person–Birthplace* relation with a small number of annotated data could have an important impact on applications such as business analysis research or automatic ontology construction.

Currently, research in relation extraction focuses mainly on pattern learning and matching techniques for extracting relational entity pairs from large corpora or the Web. The Web forms a fertile source of data for relation extraction, but users of relation extraction system are typically required to provide a large amount of annotated text to identify the interesting relation. This requirement is not feasible in real world applications. Therefore, many systems have been proposed to address the task of Web-based relation extraction, which usually only need a small number of seed entity pairs of relations. These systems typically build on the paradigm of bootstrapping of entity pairs and patterns as proposed by Brin[1].

However, an entity pair often has more than one type of semantic relations in real world. Consequently, a bootstrapping-based extraction system might introduce some

A. Gelbukh (Ed.): CICLing 2011, Part II, LNCS 6609, pp. 127–138, 2011.

irrelevant noises into further iteration. For example, given the entity pair (*Bill Gates, Microsoft*) for *CEO-of-Company* relation, two context patterns: "*was the CEO of*" and "*has retired from*" can be easily extracted from the Web. These context patterns express two different relationships and only the former is relevant to the target relation. If the irrelevant context pattern is used for further iteration, more irrelevant context patterns will be introduced into the extracted results. Although many filtering functions have been proposed in the literature, the extraction precision still is not satisfactory[2]. A significant number of noise or weakly relevant relationship instances are returned. Therefore, we use a graph based multi-view learning algorithm to rank all the extracted entity pairs by their relevance to the given seeds.

A semantic relation between two entities can be represented from two different aspects or views: the entity pair itself and the surrounding context. For example, the *Person–Birthplace* relation can be expressed as a set of entity pairs, such as (*Albert Einstein, Ulm*) and (*Jesus, Bethlehem*). From a lexical pattern view, the *Person–Birthplace* relation can also be represented with some context patterns, such as "*A was born in B*", "*B, the birth place of A*". Meanwhile, for a bootstrapping relation extraction system, an entity pair by context pattern co-occurrence matrix can be constructed easily. Then we construct two weighted complete graphs for all entity pairs and context patterns respectively. Concretely, we use each entity pair as vertex to construct a complete graph G_e, the edges are weighted with certain similarity between the entity pairs. We construct a context pattern graph G_c similarly. Since G_e and G_c is composed of entity pairs and context patterns in each view respectively, the two graphs are termed intra-view graph. We also construct a bipartite graph G_b composed with entity pairs and context patterns. Each edge on G_b links an entity pair and a context pattern. These edges are weighted with the relevance of the linked entity pair and context pattern. Vertices of graph G_b are composed of entity pair view and context pattern view, so G_b is an inter-view graph. We combine these three graphs together to accurately compute the ranking score of each entity pair.

The multi-view learning algorithm is based on inter-view and intra-view consistency assumptions. Given an entity pair e, if e strongly links to other high ranking score entity pairs on G_e, then e is likely to achieve a high ranking score. Meanwhile, on the graph G_b, if e links frequently to the context patterns whose ranking scores are high, then e is likely to achieve a high ranking score. The intra-view consistency assumption means that nearby entity pairs on G_e are likely to have similar ranking score. In addition, the context pattern graph provides us with similarity between extracted context pattern and given context pattern seed. The inter-view consistency assumption makes the similarity between context patterns useful to compute the relational similarity between entity pairs. Because if an entity pair e frequently co-occur with a context pattern c which is very similar to the given context pattern seeds, then ranking score of entity pair e should be high.

The remainder of the paper is organized as follows. The following section presents a discussion of related works. Subsequently, a bootstrapping-based relation extraction system is introduced. Using this relation extraction system, we extract some entity pairs for ranking. Thereafter, the multi-view ranking algorithm and some empirical results are presented. Finally, we conclude this paper.

2 Related Work

Bootstrapping-based relation extraction [1,3,4,5,6] leverage large amounts of data on the Web efficiently. The method is initialized using a seed set; it extracts relative facts or relations. Sergey Brin propose DIPRE system [1] to extract *author–book* relation form the Web; The Snowball system[3] extracts entity pairs including a predefined relation from a corpus. KnowItAll[4] and Espresso[5] is also bootstrapping-based system, but a different type of pattern evaluation method is used. They compute co-occurrence of context patterns and entity pairs to filter context patterns. The SatSnowball [6] extends the Snowball using statistical methods and extracts entity pairs and keywords around the entities. Furthermore, both DIPRE [1] and SatSnowball use a general form to represent extracted patterns. Although these general form patterns improve the coverage of extracted patterns, they decrease the precision. Moreover, general form patterns cannot be used directly as a query for a Web search engine, which is an efficient tool to retrieve texts on the Web. A crucial issue of these system is to filter noise out of the instances for further iteration. Sebastian Blohm et. al. systematical evaluated the impact of different filtering functions [2].

Open Information Extraction (Open IE)[7] is a domain independent information extraction paradigm which uses some generalized patterns to extract all potential relations between name entities. The generalized patterns are extracted from a dependency parsed corpus. Although Open IE is different from the bootstrapping-based method, the ranking of extracted entity pairs is also useful to retrieve these entity pairs.

Many previous reports have described that the proper use of unlabeled data can complement a traditional labeled dataset to improve the performance of a supervised algorithm. For example, a named entity classification algorithm proposed in [8], which is based on co-training framework, can reduce the need for supervision to a handful of seed rules. Label propagation [9] is a graph-based semi-supervised learning model in which the entire dataset is presented as a weighted graph; then the label score is propagated on this graph. Zhou et al. proposed a label propagation algorithm working on spectral graph [10]. In this paper, our Multi-View Ranking algorithm combines the label propagation approach with the multiple views idea from co-training.

3 A Framework for Bootstrapping Based Relation Extraction

This section provides an overview of the bootstrapping relation extraction framework which is used to extract entity pairs for ranking. The main components are explained in upcoming sections. Figure 1 portrays the framework architecture. In the framework, each sentence containing target relation is represented as a tuple, (e, c), where $e = (e_a, e_b)$ is an entity pair and c is a context pattern. The *input* of framework is a sentence set $S^0 = \{(e_i, c_j)|i = 1, 2, \cdots, n; j = 1, 2, \cdots, m;\}$ which is composed of the entity pair e_i and the context pattern c_j. The *output* of the framework is a set of entity pairs and a set of context patterns. The system distends S^0 to construct a potential target entity pair set E and a context pattern set C.

The *Extraction* part uses a dual extraction model. $E^0 = \{e_i|(e_i, c.) \in S^0\}$ and $C^0 = \{c_j|(e., c_j) \in S^0\}$ represent respectively the entity pairs and context patterns

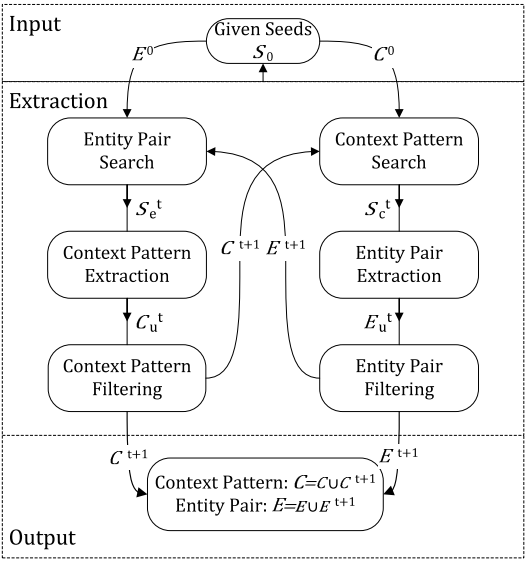

Fig. 1. Framework of a bootstrapping relation extraction system

in S^0. In *t*-th extracting iteration, we respectively submit some queries generated from entity pair set E^t and context pattern set C^t (at the beginning $t = 0$) to a search engine. The context pattern set C^t is used in the Context Search part and the entity pairs in E^t are used in the Entity Pair Search part. We collect some top ranking web pages returned by the search engine.

In the *t*-th entity pair search step, we use an entity pair e to generate some queries for a search engine. These queries are designed to retrieve the web pages in which the two entities occur. We collect some top ranking pages containing the entity pair. Then, these web pages are split into sentences. We extract the sentences in which the two entities appear simultaneously. The context patterns used in our extraction system are the contexts between an entity pair. We select and submit the lexical context patterns composed of less than five words to the search engine. We claw some top ranking pages which contain the context pattern. Then, these web pages are split into sentences. Then, the sentences captaining the context pattern are selected for further steps. In this way, the sentences set S_e^t and S_c^t is constructed respectively.

In the Entity Pair Extraction step, a Named Entity Recognization tool is used to label the named entities in each sentence. Then all entity pair candidates are extracted and added to the candidate set E_u^t. The system selects a subset of entity pair, $E^{t+1} \subseteq E_u^t$, for *t*+1 round of extraction. Simultaneously, these selected entity pairs in E^{t+1} are added to the output. Similarly, the set of context pattern C_u^t is extracted and some context patterns, $C^{t+1} \subseteq C_u^t$, are selected for *t*+1 round of entity pair expansion. The corresponding context patterns in C^{t+1} are also added to the output.

4 Entity Pair and Context Pattern Filtering Function

Because of the many-to-many relation between the entity pairs and the context patterns, extracted entity pairs and context patterns are not all applicable to the next round of extraction. For an entity pair, some context patterns that represent different types of relation may be extracted. For example, using the entity pair (*Albert Einstein, Ulm*), we can extract two types of context pattern:"*A was born in B*" and "*A's stay in B*". The two context patterns have totally different semantic relation. Therefore, the context pattern and the entity pair filtering is necessary.

In order to select most promising context patterns for further iteration, we measure each context pattern $c \in C_u^t$ using the entity pair set E^t as follow:

$$S(c) = \frac{1}{|E^t|} \sum_{e \in E^t} \frac{|e_a, c, e_b|}{|e_a, *, e_b|}$$

We write $|e_a, c, e_b|$ to denote the number of sentence $s \in S_c^t$ in which both entity pair e and context pattern c appear. $|e_a, *, e_b|$ denotes the number of sentence $s \in S_c^t$ containing entity pair (e_a, e_b). The top n patterns are selected for the next iteration. If the extracted pattern is less than n, we use all extracted patterns for further iteration.

Similarly, we measure the entity pair $e \in E_u^t$ using the context pattern set C^t as follow:

$$S(e) = \frac{1}{|C^t|} \sum_{c \in C^t} \frac{|e_a, c, e_b|}{|*, c, *|}$$

where $|*, c, *|$ is the number of sentence $s \in S_e^t$ containing context pattern c. We select the top m entity pairs for the next iteration. If the number of entity pair in E_u^t is less than m , we use all entity pairs in E_u^t for the next iteration.

5 Semi-supervised Multi-view Ranking

In this section, we illustrate the graph based multi-view algorithm which ranks all the extracted entity pairs by their relevance to the given entity pair seeds.

5.1 Intra-view and Inter-view Graphs Generation

Before applying the multi-view ranking algorithm, we need to construct a co-occurrence matrix M of the extracted entity pairs in E and the extracted context patterns in C. m_{ij} is the co-occurrence frequency of entity pair $e_i = (e_{ia}, e_{ib}) \in E$ and context pattern $c_j \in C$. For entity pair e_i and context pattern c_j, we submit a query, like "e_{ia} c_j e_{ib}", to the search engine. Then the number of hits returned by the search engine is set as m_{ij}. For two entity pairs e_h and e_i, the corresponding rows $M_{h\cdot}$ and $M_{i\cdot}$ are the vector form of entity pairs. The context patterns in C are treated as features to represent entity pairs. Consequently, we use function $Sim_e(M_{h\cdot}, M_{i\cdot})$ to calculate the similarity between e_h and e_i. Similarly, we can compute the similarity of context pattern c_j and c_k using the function $Sim_c(M_{\cdot j}, M_{\cdot k})$.

Algorithm 1. Multi-View Ranking

Given:
– Intra-view similarity matrices T^e, T^c
– Inter-view correlation matrices T^b, $[T^b]^\top$
– Ranking score vector of two views $Y = \left[\begin{smallmatrix} Y^e \\ Y^c \end{smallmatrix}\right]$
1. Generate Matrix T.

$$T = \begin{bmatrix} T^e & T^b \\ [T^b]^\top & T^c \end{bmatrix}$$

2. Normalize matrix T: $W = \frac{1}{\lambda}T$, where $\lambda = \max_i \sum_j T_{ij}$.
3. Propagate on Matrix W.
repeat
$Y_{t+1} \leftarrow WY_t + Y_0$
until converge
4. Sort entity pairs in E by corresponding score in Y^e.

Following [9], we use both labeled and unlabeled nodes to create fully connected graph. We construct two intra-view graphs $G_e =< V_e, L_e >, G_c =< V_c, L_c >$. Here, $V_e = E$ and $V_c = C$ respectively represent the data points in *entity pair* view and *context pattern* view; the weighted edges in L_e and L_c correspond to similarities between data points in different views. Taking G_e as example, a $|E| \times |E|$ matrix T^e is constructed to represent graph G_e. The edge between entity pair e_h and e_i is weighted as Eq.1. Parameter σ is the average similarity of all node pairs in G_e. Using the same method, we construct weighted graph G_c and get a $|C| \times |C|$ matrix T^c.

$$T^e_{hi} = \exp\left(\frac{-Sim_e(M_{h\cdot}, M_{i\cdot})}{2\sigma^2}\right) \tag{1}$$

We also construct a bipartite graph $G_b =< V_b, L_b >$. The vertex set V_b is composed of entity pairs and context patterns, the edges of G_b connect an entity pair and a context pattern. The function $Sim_b(e_i, c_j)$ is designed to measure the correlation of the entity pair e_i and the context pattern c_j. We construct a $|E| \times |C|$ matrix T^b to express this graph.

$$T^b_{ij} = \exp\left(\frac{-Sim_b(e_i, c_j)}{2\sigma^2}\right)$$

Above matrices T^e, T^c and T^b are used as the input of the multi-view ranking algorithm.

5.2 Multi-view Ranking Algorithm

Let Y^e be a $|E|$ dimensional ranking score column vector, where $[Y^e]_i$ denotes the similarity of e_i to the given seeds. The given seeds are evaluated as 1 in Y^e_0, other elements are initialized as zero. Similarly, Y^c is a $|C|$ ranking score column vector, whose i-*th* row represents the ranking score of context pattern c_i. In addition, Y^c_0 is initialized similarly as Y^e_0. Let Y be a $(|E| + |C|)$-dimensional column vector.

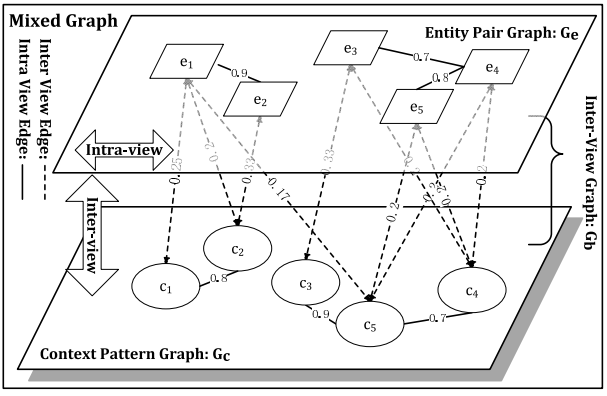

Fig. 2. Illustration to multi-view ranking algorithm. This mixed graph contains two intra-view graphs and an inter-view graph.

Algorithm 1 presents the multi-view ranking algorithm. In the first step of the algorithm, we use three graphs mentioned above to generate a matrix T. Putting the three graphs together makes the algorithm more concision. Matrix T can represent a mixed graph, as Figure 2 shows, in which both entity pairs and context patterns are vertices. On this mixed graph $G =< V, L >$, we have $V = E \cup C$ and $L = L_e \cup L_c \cup L_b$. The ranking score of each entity pair is decided by both linked entity pairs and linked context patterns on graph G. On the context pattern graph G_c, the context patterns that are similar to give seeds get high ranking score. Each context pattern's ranking score is propagated to linked entity pairs through inter-view graph G_b. The weight of edges in graph G_b control the context patterns' influence to the linked entity pairs.

In the second step, the matrix M is normalized symmetrically, which is necessary for the convergence of the following iteration.

In the third step, the label score is propagated on the mixed graph generated in the first step. Finally, entity pairs $e_i = (e_{ia}, e_{ib})$ whose relevance scores in $[Y^e]_i$ are the highest l are returned.

6 Experiments

6.1 Relation Extraction System

In order to generate entity pairs for ranking experiment, we built a relation extraction system using the framework in Section 3. In this relation extraction system, we index 4556821wikipedia[1] pages and built a local search engine using Lucene[2] toolkit. In entity pair extraction step of the relation extraction system, we construct a dictionary based named entity recognizer. The used entity dictionary is composed with all extracted named entities of YAGO project[3].

[1] http://www.wikipedia.org/

[2] http://lucene.apache.org/

[3] http://www.mpi-inf.mpg.de/yago-naga/yago/

Table 1. Relations used for evaluation

hasChild	Person and their children, $n = 4454$
isLeaderOf	Person and the organization led by them, $n = 2887$
bornIn	Person and their birth place, $n = 36189$
hasCapital	Countries or Provinces and their capital, $n = 1368$
hasWonPrize	prize winners and prizes, $n = 23075$

In the entity pair filtering step, 100 entity pairs are selected for further iteration. These entity pairs are also outputted for ranking. In the context pattern filtering step, 50 context patterns with the highest score are used for the next round of context pattern searching.

We repeat the bootstrapping process 5 times for every relation. At the beginning, 50 entity pairs and 10 context patterns are inputted as seeds.

6.2 Datasets and Evaluation Measures

We construct a large relation set using the result of YAGO project. We select 5 types of relation extracted by YAGO project for our experiments. Some facts about these selected relations are given in Table 1.

In order to evaluate the ranking performance, we use the extracted entity pairs of YAGO project as golden standard. For each relation type, we use $rel(i)$ to measure the relevance of a given entity pair e_i. For a target relation type, $rel(i)$ is set to 1 when e_i appears in the corresponding golden standard, otherwise $rel(i)$ is set to 0. For two lists showing the same instances with different order, we suppose that the list in which highly relevant instances appear earlier (have higher ranks) is more useful. Recently, some experiments show that the Mean Average Precision (MAP) measure is sensitive to the query set size or may even provide misleading results. Comparing with MAP, Normalized Discounted Cumulative Gain ($nDCG$) appeared more robust to query set size and more relaible[11]. Therefore, we adopt $nDCG$ to measure ranking quality. $nDCG$ emphasizes highly relevant instances appearing early in the result list. The $nDCG$ measure is built on DCG metric which is defined as follow:

$$DCG@p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(1+i)}$$

For a extracted entity pair set, an ideal list can be produced by sorting entity pairs of the list by $rel(i)$. $IDCG@p$ is used to annotate the DCG at position p of this ideal list. Then, we have $nDCG@p$ as follow:

$$nDCG@p = \frac{DCG@p}{IDCG@p}$$

We compare the ranking results of 5 relations using the $nDCG@p$ measure.

6.3 Baseline Methods

We compare our algorithm against following methods:

VSM: This method is a vector based method which is proposed by Turney et al.[12]. Since the co-occurrence matrix M of entity pair and context pattern is built as mentioned in previous section, the entity pair can be presented with context patterns using the rows of M. The similarity between the entity pairs can be computed as the cosine of the two corresponding vectors. We compute the similarity of candidate entity pair and given seed entity pair as below:

$$VSM(e) = \frac{\sum_{i=1}^{|S|} cos(s_i, e)}{|S|}$$

where S is the set of entity pair seeds. Then the entity pairs which have the highest similarity score are selected.

CON: This is the measure proposed by Agichtein et.al[3]. The patterns are measured by the *confidence*, by which the context patterns that tend to generate wrong entity pairs are filtered. In this experiment, we use the instance confidence to measure the quantity of context pattern and entity pair.

$$CON(e) = \frac{e_{positive}}{e_{positive} + e_{negative}}$$

in which $e_{positive}$ is the number of positive matches of entity pair or context pattern. Taking entity pair as example, if entity pair e matches the pattern c which can be found in previous iteration, then this match is considered as a *positive* match. Otherwise, the match is *negative*.

LRA: The Latent Relational Analysis(LRA) is proposed by Turney [12]. For a matrix M, supposing the rows represent the entity pairs and the columns represent context patterns. Then Singular value decomposition(SVD) is performed on the matrix, in which the matrix toolkit [4] is used. The relation similarity of entity pair can be measured by the cosine of the angle between the two vector in matrix $U_k \Sigma_k$. Similarly, the relevance of context pattern can be measured using the vector in matrix $\Sigma_k V_k^T$. In our experiment, k is set as 10. LRA is the current state-of-the-art relation similarity measure.

PMI: The *Espresso* information extraction system[5] uses the pointwise mutual information (PMI) to measure the relation between context pattern and entity pair:

$$PMI(e_i, c_j) = \frac{|e_{ia}, c_j, e_{ib}|}{|e_{ia}, *, e_{ib}||*, c_j, *|}$$

The ranking score of entity pair e_i is set as $\sum_{j=1}^{|C|} PMI(e_i, c_j)$.

[4] http://code.google.com/p/matrix-toolkits-java/

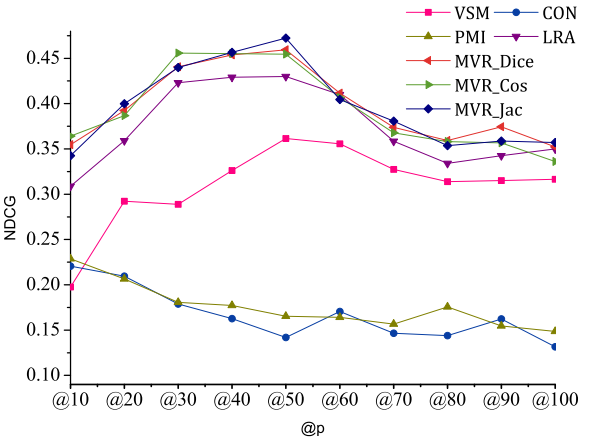

Fig. 3. Average $nDCG@p$ of five relations

In this experiment, we take Dice coefficient as inter-view measure to weight the graph G_b and test the sensitivity of multi-view ranking to the intra-view similarity measure. We test three frequently used similarity measures in the naturel language processing community. These measures are used to weight the edges of graph G_e and G_c. Following definitions only take graph G_e as example, context patterns' similarity are calculated using column vector $M_{\cdot j}$ and $M_{\cdot k}$.

Dice: Dice coefficient is a usually used measure in Natural Language Processing community. In this experiment we want to test the sensitivity of label propagation algorithm to the similarity measures.The Dice coefficient is used to weight the graph G_e which is defined as:

$$Sim_e(M_{h\cdot}, M_{i\cdot}) = 2\frac{|M_{h\cdot} \bigcap M_{i\cdot}|}{|M_{h\cdot}| + |M_{i\cdot}|}$$

Cos: In this method, the cosine similarity is used to weight the graph G_e and G_c. Given co-occurrence matrix of context pattern and entity pair M, we can construct the graph G_e with the following cosine similarity measure:

$$Sim_e(M_{h\cdot}, M_{i\cdot}) = Cosine(M_{h\cdot}, M_{i\cdot})$$

Jac: In this setting, we compute the *Jaccard* score between each entity pairs e_h and e_i, using following equation:

$$Sim_e(M_{h\cdot}, M_{i\cdot}) = \frac{|M_{h\cdot} \cap M_{i\cdot}|}{|M_{h\cdot} \cup M_{i\cdot}|}$$

Table 2. Average $nDCG@p$ of Multi-View Ranking and Baselines, ($p = \{10, 20, ..., 100\}$)

Relation \ Method	VSM	CON	PMI	LRA	MVR-Dice	MVR-Cos	MVR-Jac
hasChild	0.2691	0.2541	0.2492	0.3995	**0.4247**	0.4120	0.4083
isLeaderOf	0.1977	0.1431	0.1562	0.3703	**0.4004**	0.3791	0.3927
bornIn	0.4052	0.1450	0.1588	0.3964	0.3878	**0.4122**	0.4049
hasCapital	**0.4424**	0.1380	0.1538	0.3758	0.3960	0.3896	0.4018
hasWonPrize	0.2292	0.1495	0.1617	0.3132	0.3655	**0.3656**	0.3587
Average	0.3087	0.1659	0.1759	0.3710	**0.3949**	0.3917	0.3932

6.4 Experimental Results

These seven methods described above presented for comparison in table 2 and Figure 3.

Figure 3 shows the average $nDCG@p$ score of five relationships. We observe from Figure 3 that multi-view ranking algorithm(MVR) outperforms other methods. Comparing vector space mode (VSM) with measure based CON and PMI, we can observe that VSM works better than CON and PMI. Comparing multi-view ranking based method (MVR) with LRA, the multi-view ranking algorithms outperform LRA in most cases except $p = 60$ and $p = 100$.

Table 2 shows the performance of these algorithms on different relations. The results show that MVR algorithm outperforms other methods except the *hasCapitial* relation. Furthermore, the multi-view ranking based algorithms get the highest $nDCG$ score in average of five relations. A close look into the contexts extracted from the relation extraction system reveal that context patterns of *hasChild* relation contain less noise. Then when we rank the extracted entity pairs of *hasChild* relation, most algorithms achieve better performance than other relations.

7 Conclusions

We propose a graph based multi-view learning algorithm to rank the returned relation instances of a bootstrapping-based relation extraction system. We compare the MVR algorithm to the existing methods, relevant score based methods and frequency based methods, the results indicate that the MVR can improve the performance of the relation extraction systems.

References

1. Brin, S.: Extracting patterns and relations from the world wide web. In: WebDB Workshop at EDBT 1998, pp. 172–183 (1998)
2. Blohm, S., Cimiano, P., Stemle, E.: Harvesting relations from the web: quantifiying the impact of filtering functions. In: Proceedings of the 22nd National Conference on Artificial Intelligence, pp. 1316–1321 (2007)

3. Agichtein, E., Gravano, L.: Snowball: Extracting relations from large plain-text collections. In: Proceedings of the Fifth ACM International Conference on Digital Libraries (2000)
4. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligence 165, 91–134 (2005)
5. Pantel, P., Pennacchiotti, M.: Espresso: Leveraging generic patterns for automatically harvesting semantic relations (2006)
6. Zhu, J., Nie, Z., Liu, X., Zhang, B., Wen, J.R.: Statsnowball: a statistical approach to extracting entity relationships. In: Proceedings of the 18th International World Wide Web Conference, pp. 101–110 (2009)
7. Banko, M., Cafarella, M.J., Soderl, S., Broadhead, M., Etzioni, O.: Open information extraction from the web. In: IJCAI, pp. 2670–2676 (2007)
8. Collins, M., Singer, Y.: Unsupervised models for named entity classification. In: Proc. Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, pp. 100–110 (1999)
9. Zhu, X., Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107 (2002)
10. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: Advances in Neural Information Processing Systems, vol. 16, pp. 321–328. MIT Press, Cambridge (2004)
11. Radlinski, F., Craswell, N.: Comparing the sensitivity of information retrieval metrics. In: Proceeding of the 33rd international ACM SIGIR Conference on Research and Development in Information Retrieval 2010, pp. 667–674 (2010)
12. Turney, P.D.: Similarity of semantic relations. Computational Linguistics 32, 379–416 (2006)