Collectively Representing Semi-Structured Data from the Web

Bhavana Dalvi
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
bbd@cs.cmu.edu

William W. Cohen
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
wcohen@cs.cmu.edu

Jamie Callan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
callan@cs.cmu.edu

Abstract

In this paper, we propose a single low-dimensional representation of a large collection of table and hyponym data, and show that with a small number of primitive operations, this representation can be used effectively for many purposes. Specifically we consider queries like set expansion, class prediction etc. We evaluate our methods on publicly available semi-structured datasets from the Web.

1 Introduction

Semi-structured data extracted from the web (in some cases extended with hyponym data derived from Hearst patterns like "X such as Y") have been used in several tasks, including set expansion (Wang and Cohen, 2009b; Dalvi et al., 2010) automatic set-instance acquisition (Wang and Cohen, 2009a), fact extraction (Dalvi et al., 2012; Talukdar et al., 2008)), and semi-supervised learning of concepts (Carlson et al., 2010). In past work, these tasks have been addressed using different methods and data structures. In this paper, we propose a single low-dimensional representation of a large collection of table and hyponym data, and show that with a small number of primitive operations, this representation can be used effectively for many purposes.

In particular, we propose a low-dimensional representation for entities based on the embedding used by the PIC algorithm (Lin and Cohen, 2010a). PIC assigns each node in a graph an initial random value, and then performs an iterative update which brings together the values assigned to near-by nodes, thus producing a one-dimensional embedding of a graph. In past work, PIC has been used for unsupervised clustering of graphs (Lin and Cohen, 2010a); it has also been extended to bipartite graphs (Lin and Cohen, 2010b), and it has been shown that performance can be improved by using multiple random starting points, thus producing a low-dimensional (but not one-dimensional) embedding of a graph (Balasubramanyan et al., 2010).

2 The PIC3 Representation

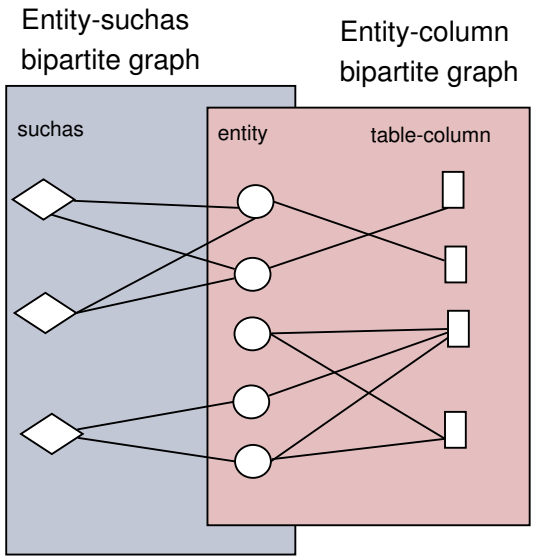


Figure 1: Entities on the Web

We use PIC to produce an embedding of a tripartite graph, in particular the data graph of Figure 1. We use the publicly available (Dalvi et al., 2012) entity-tableColumn co-occurrence dataset and Hyponym Concept dataset. Each edge derived from the entity-tableColumn dataset links an entity name with an identifier for a table column in which the entity name appeared. Each edge derived from the Hyponym Concept Dataset links an entity X and a concept Y with which it appeared in the context of

Proc. of the Joint Workshop on Automatic Knowledge Base Construction & Web-scale Knowledge Extraction (AKBC-WEKEX), pages 7–12,
NAACL-HLT, Montréal, Canada, June 7-8, 2012.

a Hearst pattern (weighted by frequency in a large web corpus). We combine these edges to form a tripartite graph, as shown in Figure 1. Occurrences of entities with hyponym (or "such as") concepts form a bipartite graph on the left, and occurrences of entities in various table-columns form the bipartite graph on the right. Our hypothesis is that entities co-occurring in multiple table columns or with similar suchas concepts probably belong to the same class label.

Since we have two bipartite graphs, entity-tableColumn and entity-suchasConcept, we create bipartite PIC embeddings for each of these in turn (retaining only the part of the embedding relevant to the entities). Specifically, we start with m random vectors to generate m-dimensional PIC embedding. Since we have two bipartite graphs, entity-tableColumn and entity-suchasConcept, we create PIC embeddings for each of them separately. The embedding for entities is then the concatenation of these separate embeddings (refer to Algorithm 1). Below we will call this the PIC3 embedding.

Figure 2 shows the schematic diagrams for final and intermediate matrices while creating the PIC3 embedding. We have experimented with a version of this algorithm in which we create PIC embeddings of the data by concatenating the dimensions first instead of computing separate embeddings and later concatenating them. We observed that the version showed in Algorithm 1 performs as good as or better than its variant.

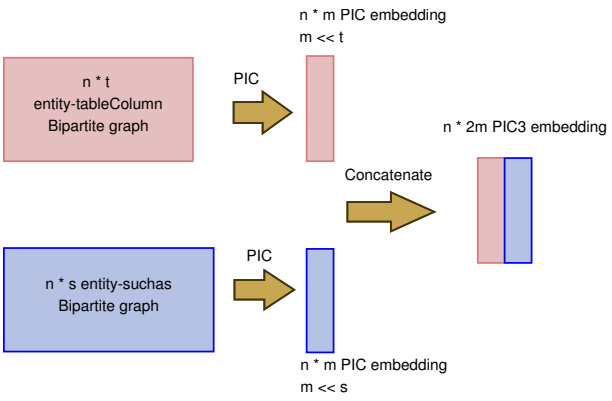


Figure 2: Schematic diagram of matrices in the process of creating PIC3 representation (n : number of entities, t : number of table-columns, s : number of suchas concepts and m : number of PIC dimensions per bipartite graph).

Algorithm 1 Create PIC3 embedding

function Create_PIC3_Embedding(E, X_T, X_S, m): X_{PIC3}
Input: E: Set of all entities,
X_T: Co-occurrence of E in table-columns,
X_S: Co-occurrence of E with suchasConcepts,
m: Number of PIC dimensions per bipartite graph
Output: X_{PIC3}: 2*m dim. PIC3 embedding of E.
$X_{PIC3} = \phi$
t = a small positive integer
for d = 1 : m **do**
  V_0 = randomly initialized vector of size $|E| * 1$
  V_t = *PIC_Embedding*(X_T, V_0, t)
  Add V_t as d^{th} column in X_{PIC3}
end for
for d = 1 : m **do**
  V_0 = randomly initialized vector of size $|E| * 1$
  V_t = *PIC_Embedding*(X_S, V_0, t)
  Add V_t as d^{th} column in X_{PIC3}
end for
end function

Our hypothesis is that these embeddings will cluster similar entities together. E.g. Figure 3 shows a one dimensional PIC embedding of entities belonging to the two classes "city" and "celebrity". The value of embedding is plotted against its entity-index, and color indicates the class of an entity. We can clearly see that most entities belonging to the same class are clustered together. In the next section, we will discuss how the PIC3 embedding can be used for various semi-supervised and unsupervised tasks.

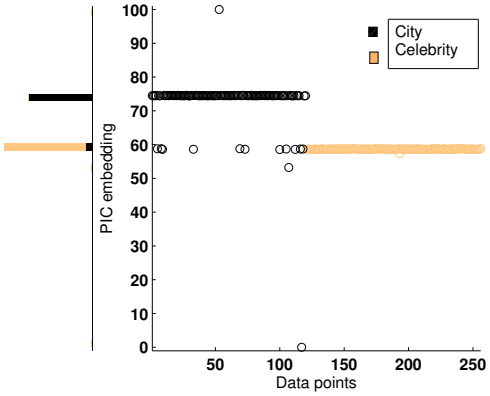


Figure 3: One dimensional PIC embedding for 'City' and 'Celebrity' classes

3 Using the PIC3 Representation

In this section we will see how this PIC3 representation for entities can be used for three different tasks.

3.1 Semi-supervised Learning

In semi-supervised transductive learning, a few entities of each class are labeled, and learning method extrapolates these labels to a larger number of unlabeled data points. To use the PIC3 representation for this task, we simply learn a linear classifier in the embedded space. In the experiments below, we experiment with using labeled entities Y_n from the NELL Knowledge Base (Carlson et al., 2010). We note that once the PIC3 representation has been built, this approach is much more efficient than applying graph-based iterative semi-supervised learning methods (Talukdar and Crammer, 2009; Carlson et al., 2010).

3.2 Set Expansion

Set expansion refers to the problem of expanding a set of seed entities into a larger set of entities of the same type. To perform set expansion with PIC3 representation, we find the K nearest neighbors of the centroid of the set of seed entities using a KD-tree (refer to Algorithm 2). Again, this approach is more efficient at query time than prior approaches such as SEAL (Wang and Cohen, 2009b), which ranks nodes within a graph it builds on-the-fly at set expansion time using queries to the web.

Algorithm 2 Set Expansion with K-NN on PIC3

1: **Input**: Q: seed entities for set expansion ,
X_{PIC}: low dimensional PIC3 embedding of E
2: **Output:** Q' : Expanded entity set
3: k = a large positive number
4: Q_c = centroid of entities in Q
5: Q' = Find-K-NearestNbr(Q_c, X_{PIC}, k)

3.3 Automatic Set Instance Acquisition (ASIA)

This task takes as input the name of a semantic class (e.g.,"countries") and automatically outputs its instances (e.g., "USA" , "India" , "China" etc.). To perform this task, we look up instances of the given class in the hyponym dataset, and then perform set expansion on these - a process analogous to that used in prior work (Wang and Cohen, 2009a). Here, however, we again use Algorithm 2 for set expansion, so the entity-suchasConcept data are used only to find seeds for a particular class Y. Again this method requires minimal resources at query time.

Dataset	Toy_Apple	Delicious_ Sports
$\|\boldsymbol{X}\|$: # entities	14,996	438
$\|\boldsymbol{C}\|$: # table-columns	156	925
$\|(\boldsymbol{x}, \boldsymbol{c})\|$: # $(\boldsymbol{x}, \boldsymbol{c})$ edges	176,598	9192
$\|\boldsymbol{Y_s}\|$: # suchasConcepts	2348	1649
$\|(\boldsymbol{x}, \boldsymbol{Y_s})\|$: # $(\boldsymbol{x}, \boldsymbol{Y_s})$ edges	7683	4799
$\|\boldsymbol{Y_n}\|$: # NELL Classes	11	3
$\|(\boldsymbol{x}, \boldsymbol{Y_n})\|$: # $(\boldsymbol{x}, \boldsymbol{Y_n})$ pairs	419	39
$\|\boldsymbol{Y_c}\|$: # manual column labels	31	30
$(\boldsymbol{c}, \boldsymbol{Y_c})$: # $(\boldsymbol{c}, \boldsymbol{Y_c})$ pairs	156	925

Table 1: Table datasets Statistics

4 Experiments

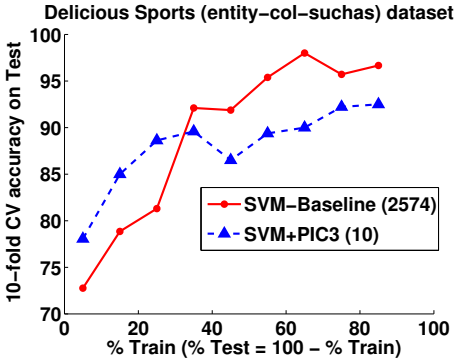
Although PIC algorithm is very scalable, in this paper we evaluate performance using smaller datasets which are extensively labeled. In particular, we use the Delicious_Sports, Toy_Apple and Hyponym Concept datasets made publicly available by (Dalvi et al., 2012) to evaluate our techniques. Table 1 shows some statistics about these datasets. Numbers for $|Y_s|$ and $|(x, Y_s)|$ are derived using the Hyponym Concept Dataset.

4.1 Semi-supervised Learning (SSL)

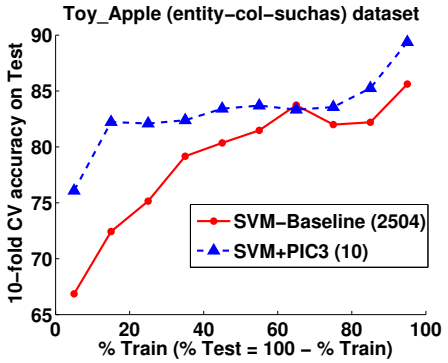
To evaluate the PIC embeddings in terms of predicting NELL concept-names, we compare the performance of an SVM classifier on PIC embeddings (named SVM+PIC3) vs. the original high-dimensional dataset (named SVM-baseline). In SVM-baseline method the hyponyms and table-columns associated with an entity are simply used as features. The number of iterations t for PIC and number of dimensions per view m were set to $t = m = 5$ in these experiments. (Experiments with $m > 5$ showed no significant improvements in accuracy on Toy_Apple dataset.)

Figure 4 shows the plot of accuracy vs. training size for both datasets. We can see that SVM+PIC3 method is better than SVM-Baseline with less training data, hence is better in SSL scenarios. Also note that PIC3 embedding reduces the number of dimensions from 2574 (Delicious_Sports) and 2504

(Toy_Apple) to merely 10 dimensions. We checked the rank of the matrix which we use as PIC3 representation to make sure that all the PIC embeddings are distinct. In our experiments we found that an m dimensional embedding always has $rank = m$. This is achieved by generating a new random vector V_0 using distinct randomization seeds each time we call the PIC embedding procedure (see Line 7 and 12 in Algorithm 1).



(a) Delicious_Sports dataset



(b) Toy_Apple dataset

Figure 4: SSL Task : Comparison of SVM+PIC3 vs. SVM-Baseline

4.2 Set Expansion

We manually labeled every table column from Delicious_Sports and Toy_Apple datasets. These labels are referred to as Y_c in Table 1. This also gives us labels for the entities residing in these table-columns. We use the set of entities in each of these table columns as "a set expansion query" and evaluate "the expanded set of entities" based on manual labels. The baseline runs K-Nearest Neighbor on the original high-dimensional dataset (referred to as K-NN-Baseline).

As another baseline, we adapt the MAD algorithm (Talukdar and Crammer, 2009), a state-of-the art semi-supervised learning method. Similar to a prior work by (Talukdar et al., 2010), we adapt MAD for unsupervised learning by associating each table-column node with its own id as a label, and propagating these labels to other table-columns. MAD also includes a "dummy label", so after propagation every table-column T_q will be labeled with a weighted set of table-column ids $T_{s_1}, ...T_{s_n}$ (including its own id), and also have a weight for the "dummy label". We denote MAD's weight for associating table-column id T_s with table column T_q as $P(T_s|T_q)$, and consider the ids $T_{s_1}, ...T_{s_k}$ with a weight higher than the dummy label's weight. We consider e_1, e_2, ... e_n, the union of entities present in columns $T_{s_1}...T_{s_k}$, and rank them in descending order score, where $score(e_i, T_q) = \sum_{j:e_i \in T_{s_j}} P(T_{s_j}|T_q)$. Figure 5 shows Precision-

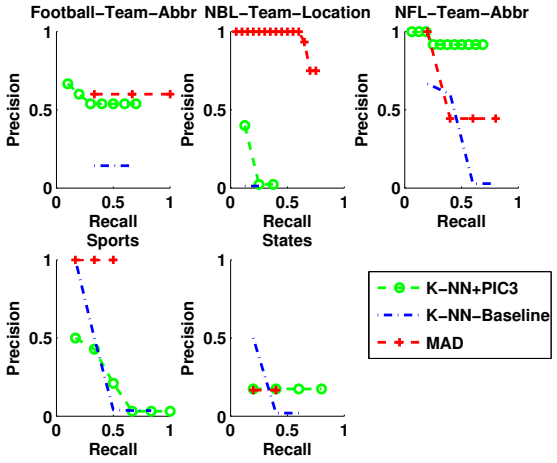


Figure 5: Set Expansion Task : Precision recall curves

Recall curves for all 3 methods, on sample set expansion queries. These plots are annotated with manual column-labels (Y_c). For most of the queries, K-NN+PIC3 performs as well as K-NN-Baseline and is comparable to MAD algorithm. Table 2 shows the running time for all three methods. K-NN+PIC3 method incurs a small amount of pre-processing time (0.02 seconds) to create embeddings and compared to other two methods it is very fast at the query time. The numbers show total query time for 881 Set Expansion queries and 25 ASIA queries (described below).

Method	Total Query Time (s)	
	Set Expansion	ASIA
K-NN+PIC3	12.7	0.5
K-NN-Baseline	80.1	1.4
MAD	38.2	150.0

Table 2: Comparison of Run Time on Delicious_Sports

4.3 Automatic Set Instance Acquisition

For the automatic set instance acquisition (ASIA) task, we use concept-names from Hyponym Concept Dataset (Y_s) as queries. Similar to the Set Expansion task, we compare K-NN+PIC3 to the K-NN-Baseline and MAD methods.

To use MAD for this task, the concept name Y_s is injected as label for the ten entities that co-occur most with Y_s, and the label propagation algorithm is run. Each entity e_i that scores higher than the dummy label is then ranked based on the probability of the label Y_s for that entity.

Figure 6 shows the comparison of all three methods. K-NN+PIC3 generally outperforms K-NN-Baseline, and outperforms MAD on two of the four queries. MAD's improvements over K-NN+PIC3 for the two queries comes at the expense of longer query times (refer to Table 2).

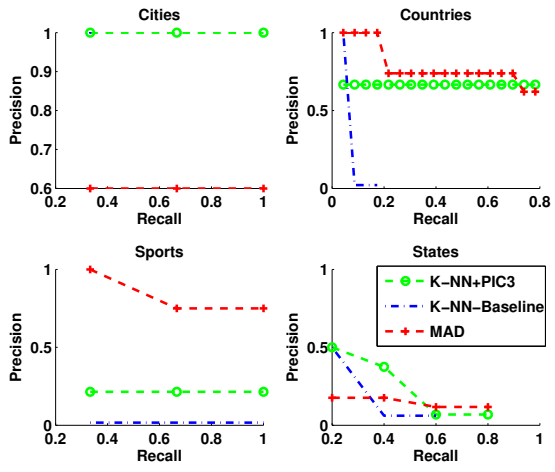


Figure 6: ASIA task : Precision recall curves

5 Conclusion

We presented a novel low-dimensional representation for entities on the Web using Power Iteration Clustering. Our experiments show encouraging results on using this representation for three different tasks : (a) Semi-Supervised Learning, (b) Set Expansion and (c) Automatic Set Instance Acquisition. The experiments show that "this simple representation (PIC3) can go a long way, and can solve different problems in a simpler and faster, if not better way". In future, we would like to use this representation for named-entity disambiguation and unsupervised class-instance pair acquisition, and to explore performance on larger datasets.

Acknowledgments

This work is supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Air Force Research Laboratory (AFRL) contract number FA8650-10-C-7058. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. This work is also partially supported by the Google Research Grant.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Google, IARPA, AFRL, or the U.S. Government.

References

Balasubramanyan, R., Lin, F., and Cohen, W. W. (2010). Node clustering in graphs: An empirical study. Workshop on Networks Across Disciplines in Theory and Applications, NIPS.

Carlson, A., Betteridge, J., Wang, R. C., Hruschka, Jr., E. R., and Mitchell, T. M. (2010). Coupled semi-supervised learning for information extraction. In *WSDM*. `http://rtw.ml.cmu.edu/rtw/`.

Dalvi, B., Callan, J., and Cohen, W. (2010). Entity list completion using set expansion techniques. In *Proceedings of the Nineteenth Text REtrieval Conference*.

Dalvi, B., Cohen, W., and Callan, J. (2012). Websets: Extracting sets of entities from the web using unsupervised information extraction. In *WSDM*. datasets : `http://rtw.ml.cmu.edu/wk/WebSets/wsdm_2012_online/index.html`.

Lin, F. and Cohen, W. W. (2010a). Power iteration clustering. In *Proceedings of International Conference on Machine Learning*, ICML'10.

Lin, F. and Cohen, W. W. (2010b). A very fast method for clustering big text datasets. ECAI.

Talukdar, P. and Crammer, K. (2009). New regularized algorithms for transductive learning. In *European Conference on Machine Learning (ECML-PKDD)*.

Talukdar, P. P., Ives, Z. G., and Pereira, F. (2010). Automatically incorporating new sources in keyword search-based data integration. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10.

Talukdar, P. P., Reisinger, J., Paşca, M., Ravichandran, D., Bhagat, R., and Pereira, F. (2008). Weakly-supervised acquisition of labeled class instances using graph random walks. In *EMNLP*.

Wang, R. C. and Cohen, W. W. (2009a). Automatic set instance extraction using the web. In *ACL*.

Wang, R. C. and Cohen, W. W. (2009b). Character-level analysis of semi-structured documents for set expansion. In *EMNLP*.