Supplementary Material for "Constrained Information-Theoretic Tripartite Graph Clustering to Identify Semantically Similar Relations"

Chenguang Wang[a], Yangqiu Song[b], Dan Roth[b], Chi Wang[c], Jiawei Han[b], Heng Ji[d], Ming Zhang[a]
[a]School of EECS, Peking University
[b]Department of Computer Science, University of Illinois at Urbana-Champaign
[c]Microsoft Research, [d]Department of Computer Science, Rensselaer Polytechnic Institute
wangchenguang@pku.edu.cn, {yqsong,danr}@illinois.edu, chiw@microsoft.com
hanj@illinois.edu, jih@rpi.edu, mzhang@net.pku.edu.cn

Additional Experiments

In this supplementary material, we first present detailed statistics of Rel-KB dataset, which particularly includes multi-hop relations. Then we show the case study of clustering results on Rel-KB. Finally, we analyze the errors in clustering results on Rel-OIE.

A. Rel-KB Statistics and Multi-Hop Relations

Freebase is a publicly available knowledge base consisting of entities and relations collaboratively collected by its community members. Now, it contains over 2 billions relation expressions between 40 millions entities. We call these direct relations in Freebase as one-hop relations. There can be multiple relations linking two entities. For example, "Google" and "Larry Page" can be linked by relations "'s founder is," "'s CEO is," etc.. In this dataset, we also find multi-hop relations. For example, "Google" is linked to "Sergey Brin" with the relation "'s founder is" and "Sergey Brin" is further linked to "Larry Page" with the relation "is influence peer of." Thus, (X, 's founder is, Sergey Brin) $\wedge$ (Sergey Brin, is influence peer of, Y) can also be regarded as a relation that links "Google" and "Larry Page," where X="Google" and Y="Larry Pages." In the following, we show the details about how to generate these multi-hop relations.

Given an entity pair (e^1, e^2) and the maximum number of hops L, we want to retrieve a set of relations for the entity pair. It is time consuming and impractical to enumerate all the relations, starting from entity e^1 and ending with e^2 in Freebase. Here we use an alternative algorithm to find the multi-hop relations. We first enumerate all the $l = L/2$-hop relations for each entity. Then we build an inverted index for all the relations in the l-hop. For each unique relation, we look up the relation's left and right entities to generate multi-hop relations. The complexity of the procedure is $O(V \cdot N^l)$, where V is the number of entities and N is the average number of neighbor entities to an entity (experimentally, $N \approx 53$ and $L = 4$).

The Rel-KB dataset is finally constructed. In Table 1, we show the six relations in Rel-KB with some statistics. We have $16,516$ relation expressions of six relation categories. Compared to the datasets used by the previous research [Sutskever *et al.*, 2009], this is really a large set in the sense of number of relations. Moreover, there are near one million (981,153) relation triplets in total. There are lots of relation expressions (about 39%) being with only one entity pair. Only 32% relation expressions are with more than five unique entity pairs. This means that the data are very sparse. In this case, to generate more reasonable clusters of relations, constraints should be very helpful, since the number of entity types is relatively small and relations should have more overlapped entity types.

Relation Category	#(Left $\mathcal{E}^1$)	#(Right $\mathcal{E}^2$)	#($\mathcal{R}$)
Organization-Founder	2,824	2,883	8,826
Book-Author	4,779	4,779	1,339
Actor-Film	1,182	5,000	481
Location-Contains	217	5,000	2,532
Music-Track	397	5,000	88
Person-Profession	3,264	388	3,250

Table 1: Rel-KB dataset statistics. #(Left $\mathcal{E}^1$) means the number of entities in the left entity set $\mathcal{E}^1$; #(Right $\mathcal{E}^2$) means the number of entities in the right entity set $\mathcal{E}^2$; #($\mathcal{R}$) means the number of relation expressions in the relation set $\mathcal{R}$.

B. Case Study of Clustering Results on Rel-KB

We show some examples of TGC in Table 2(a), and examples of CTGC in Table 2(b). We use "-1" to represent the inverse order of the relation. For example, (X, write^{-1}, Y) means (Y, write, X). We find that both TGC and CTGC have reasonable clustering results. In the *Organization-Founder* cluster, TGC and CTGC find (X, founded by, Y), (X, created by, Y) and (X, started by, Y). They are all semantically similar to each other. Qualitatively, we feel that CTGC is better than TGC. For example, TGC clusters (X, 's profession^{-1}, Y) in the *Book-Author* cluster, which is not correct.

C. Error Analysis of Rel-OIE Clustering Results

In Rel-OIE dataset, we derive constraints based on a more realistic scenario. Such noisy constraints do not perform as good as the ones derived from Rel-KB with gold standard. Notice that there are cases that may be a little misleading from the clustering results of Rel-OIE. As shown in the examples of clustering results on Rel-OIE, the cluster *Organization-Founder*, (X, who left^{-1}, Y) has been clustered in. In this work, we aim to find semantically similar relations, and in this sense (X, who left^{-1}, Y) is also relevant to the cluster

(a) Examples generated by TGC.

Organization-Founder	(X, is founder of^{-1}, Y); (X, created by, Y); (X, member, Person) $\wedge$ (Person, 's gender, Gender) $\wedge$ (Gender, 's gender^{-1}, Y); (X, started by, Y); (X, leadership, Person) $\wedge$ (Person, mailing address, Location) $\wedge$ (Location, mailing address^{-1}, Y).
Book-Author	(X, write^{-1}, Y); (X, 's written work^{-1}, Y); (X, 's genre, Genre) $\wedge$ (Genre, written genre^{-1}, Y); (X, 's profession^{-1}, Y).
Actor-Film	(X, act in, Y); (X, act in, Film) $\wedge$ (Film, performance type, Type) $\wedge$ (Type, special, Y); (X, character, Y); (X, engineer, Y).
Location-Contains	(X, contained by^{-1}, Y); (X, partially contain, Y); (X, in state, State) $\wedge$ (State, 's capital, Y); (X, 's symbol, Y).
Music-Track	(X, recoded in^{-1}, Y); (X, release, Album) $\wedge$ (Album, release, Y); (X, release, Y); (X, part of profession^{-1}, Y).
Person-Profession	(X, 's profession, Y); (X, 's nationality, Y); (X, 's ethnicity, Y); (X, translated by, Y); (X, portray by^{-1}, Y); (X, lost, Y).

(b) Examples generated by CTGC.

Organization-Founder	(X, founded by, Y); (X, is creator of^{-1}, Y); (X, founded by, Person) $\wedge$ (Person, influence peer, Y); (X, business in, Industry) $\wedge$ (Industry, win award^{-1}, Y); (X, founded by, Person) $\wedge$ (Person, is a, Politician) $\wedge$ (Politician, is a^{-1}, Y).
Book-Author	(X, written by, Y); (X, part of, Series) $\wedge$ (Series, write^{-1}, Y); (X, win award, Award) $\wedge$ (Award, award winner, Y).
Actor-Film	(X, 's specialization, Actor) $\wedge$ (Actor, perform in, Film) $\wedge$ (Film, is a^{-1}, Y); (X, act in, Y); (X, perform in, Y).
Location-Contains	(X, contained by^{-1}, Y); (X, 's capital, Y); (X, 's government, Government) $\wedge$ (Government, 's jurisdiction, Y); (X, 's education, Education) $\wedge$ (Education, 's institution, Y); (Y, 's nationality^{-1}, Organization) $\wedge$ (Organization, contained by, Y).
Music-Track	(X, made^{-1}, Person) $\wedge$ (Person, same height, Person) $\wedge$ (Person, perform in, Video) $\wedge$ (Video, play in TV^{-1}, Y); (X, release, Y); (X, release, Track_list) $\wedge$ (Track_list, record, Y); (X, recording releases^{-1}, Y); (X, single version, Y).
Person-Profession	(X, 's profession, Y); (X, influence, Person) $\wedge$ (Person, 's profession, Y); (X, write, Book) $\wedge$ (Book, 's profession, Y).

Table 2: Examples of relation clusters from Rel-KB. We manually translate the relation expressions from Freebase into natural language for better understanding. We use "-1" to represent the inverse order of the relation. Notice that, we have all the cases generated by the other five clustering algorithms. Due to the space limitation, we only show the results of TGC and CTGC.

label. We will leave to find more subtle semantically similar relations in the future work.

References

[Sutskever *et al.*, 2009] Ilya Sutskever, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Modelling relational data using bayesian clustered tensor factorization. In *NIPS*, pages 1821–1828, 2009.