Discovering new drug treatments from UMLS knowledge base

Thanvir Mohamed
mpeermohamed@cs.sunysb.edu

Vijay Teja
vgottipati@cs.sunysb.edu

Abstract

UMLS knowledge base is a large repository of different medical entities and relations between them. As with all large KBs, the relations between the entities are not complete i.e. there could be latent relations between entities within the knowledge base that are not explicitly expressed. Our goal is to leverage the relation instances present in the knowledge base to extract drug-disease relations in order to discover new beneficial drugs for diseases. We employ Subgraph Feature Extraction algorithm for this knowledge base completion task.

1 Introduction

The UMLS is a large database that contains information about biomedical and health-related concepts, and relationships among them. One common feature of such manually curated databases is that they tend to be incomplete in capturing facts about the relevant entities. Further, there could be instances of relations between entities which are actually present in the database in the form of multiple relationships but not explicitly expressed as a single relation instance.

We are particularly interested in discovering new instances of beneficial drugs for diseases. The knowledge base consists of instances of drugs producing certain effects and such effects preventing or leading to some disease. From such instances, we intend to infer instances of drugs treating or leading to diseases.

We extract relevant instances from the UMLS database and perform the knowledge base completion algorithm Subgraph Feature Extraction (Gardner et al., 2015) on the extracted instances. We specify the algorithm to infer instances of beneficial drugs for diseases.

In the experiments, we considered three different subsets of the instances of UMLS and performed knowledge base completion. The results are poor when the graph of the UMLS relation instances is either too restricted or too broad. But, when the graph allows for the discovery of all and only one-hop relations, the system performs exceptionally well in discovering latent relation instances.

2 Problem Definition

The problem is, given a set of relation instances involving drugs or diseases, infer new instances of relations between drugs and diseases.

In addition to the relation instances, the algorithm requires as input training instances of beneficial and harmful drugs.

3 Approach

From the UMLS database, we extract relations of interest. Due to the large nature of the database, we try to discover those drug-disease relation pairs which are separated by only one hop i.e. there is one intermediary entity which links the drug and the disease. For example, expressing relation instance as (entity 1, relation, entity 2), from the relation instances (insulin, lowers, blood glucose) and (blood glucose, causative_agent_of, diabetes) it could be inferred that (insulin, treats, diabetes). The intermediary entity is referred to as theme.

In the UMLS knowledge base, there are no semantic types directly corresponding to drug, disease or theme. Rather, there are more fine-grained semantic types corresponding to each of them. Those fine-grained semantic types are aggregated into treatment, theme and disease.

The semantic types antibiotic, clinical drug, hazardous or poisonous substance, organic chemical, pharmacological substance, steroid, vitamin are all aggregated into the type Treatment.

The semantic types acquired abnormality, anatomical abnormality, congenital abnormality, disease or syndrome, cell or molecular dysfunction, neoplastic process, pathologic function, sign or symptom are aggregated into the type Disease.

And the semantic types anatomical structure, body location or region, body part, organ or organ component, body space or junction, cell component, cell, laboratory or test result, biologic function, cell function, genetic function, molecular function, organism function, organ or tissue function, physiologic function, amino acid, peptide or protein, enzyme, hormone are aggregated into Theme.

Similarly, there are no relations in the knowledge base directly corresponding to the beneficial and harmful relations between drugs and diseases.

The relations may_treat, may_prevent, treats, prevents are aggregated into the super-relation Beneficial.

The relations cause_of, causative_agent_of and contraindicated_drug are aggregated into the super-relation Harmful.

The above aggregations of semantic types and relations are taken from the LatEnt paper.

Now, relation instances are extracted from the UMLS database such that at least one of the entities is of the type treatment, theme or disease.

The relation whose instances which we seek to discover using knowledge base completion is the beneficial relation between drugs and diseases. For the purpose of training by the Subgraph Feature Extraction (SFE) algorithm, we use the instances of harmful relation as negative instances of beneficial.

Considering all the instances of the beneficial and harmful relations between treatments and diseases, we pass as training data to the SFE algorithm a subset of the beneficial and harmful relation instances. The harmful relation instances are passed as negative examples. The remaining beneficial and harmful relation instances are held out as the testing data. Similar to the training case, the harmful relation instances are marked as negative instances of the beneficial relation in the test data.

The remainder of the filtered-out instances are passed as the input graph file to the algorithm. In addition, the input graph file contains links of the semantic network. The semantic network encodes links between semantic types. Further, the input graph file contains instances which specify the categorization of concepts within semantic types. These are instances of the type (concept, is-a, semantic type). The algorithm builds the knowledge graph from these instances.

Using the given training data and the built knowledge graph, the algorithm learns how to infer new positive or negative instances of the beneficial relation between treatments and diseases.

The algorithm is run on the testing data and thus its performance is tested on both the positive and negative instances of the beneficial relation.

4 Evaluation

As our initial experiment, we considered a small subset of the UMLS database and ran the algorithm on it. The subset consisted of those relation instances where both the entities were of the types treatment, theme or disease. And it contained only instances of the relation types beneficial and harmful. Thus the graph file consisted of those instances of beneficial or harmful relations that are between treatment and theme or between theme and disease. The training and testing data consisted of beneficial and harmful relation instances between treatment and disease. As mentioned in the previous section, the harmful relation instances are marked as the negative instances of the beneficial relation in the training and testing data.

We performed a second experiment where we considered a larger subset of the UMLS database. This subset consisted of those relation instances where both the entities were of the types treatment, theme or disease but, unlike the previous experiment, we did not restrict to any particular types of relations. The training and testing data were the same as in the previous experiment. The input graph file now consisted of all the relation instances where both the entities were of the types treatment, theme or disease except the instances of the relations beneficial and harmful between treatment and disease.

In a third experiment, we considered the subset of UMLS database as mentioned in the previous section. This consisted of all the relation instances where at least one of the two entities is of the types treatment, theme or disease. The training and testing data were the same as in the previous experiments. The input graph file consisted of all the relation instances in the subset except those contained in the training data.

File Name	Number of relation instances
Experiment 1 Input Graph File	1,141,488
Experiment 2 Input Graph File	6,948,448
Experiment 3 Input Graph File	13,744,985

Table 1: Input graph file sizes in different experiments

Relation Name	Number of relation instances
Beneficial	38,613
Harmful	4,014

Table 2: Beneficial and Harmful relations between treatments and diseases in UMLS

In all the above experiments, 90% of the positive instances of the beneficial relation between treatment and disease and 90% of the negative instances of the relation were given as input training data to the algorithm. The remaining positive and negative instances of the beneficial relation are held out as test data.

The SFE algorithm is run with the BFS Path Finder parameter set to 2.

The experiment 1 provided very poor results in terms of recall. The precision is 0.5 (see Table 3).

The results of experiment 2 are impressive (see Table 4). The mean average precision of the resulting instances is 100%. The percentage of test instances which appeared in the results is 91%.

Interestingly, the results of experiment 3 are significantly worse than those of experiment 2 (see Table 5). The mean average precision is 0.5033 and the recall is 0.00118. The possible explanations for the worse performance in comparison to experiment 2 are discussed in the next section.

5 Discussion

The poor results of experiment 1 could be attributed to the limited size of the knowledge graph. Excluding the relation instances in the semantic network and the semantic_isa_links of the UMLS database, the input graph file has only 1,684 relation instances. Thus, the knowledge graph is build out of insufficient amount of information.

Statistic	Score
Mean Average Precision	0.5
Mean Reciprocal Rank	1.0
Recall	0.00024

Table 3: Results of experiment 1

Statistic	Score
Mean Average Precision	1.0
Mean Reciprocal Rank	1.0
Recall	0.91

Table 4: Results of experiment 2

Statistic	Score
Mean Average Precision	0.5033
Mean Reciprocal Rank	1.0
Recall	0.00118

Table 5: Results of experiment 3

In the case of experiment 2, it clearly benefits from more relation instances and thus a knowledge graph with more significant information is built.

But the trend doesn't extend to experiment 3, where despite the graph being built from a large number of relation instances, it performs worse in comparison to experiment 2.

An explanation for this could be that, in experiment 3, there are more entities in the paths linking the relevant concepts. Since, in the algorithm, we are limiting to only 2 hops, the links between the relevant concepts are not captured. But, in the case of experiment 2, where the relation instances are such that both the entities belong to treatment, theme or disease, the links to be captured would not be more than 2 hops apart. This is because the links on the path would typically be of the forms (treatment, theme) and (theme, disease). In the above kind of links, the required relation instance could be discovered in one hop. The most weighted paths in experiment 2 are 1-hop paths (see Table 7).

In experiment 2, the highest weighted path is the single relation "-_may_be_treated_by-". Thus, using the inverse of the instances of the relation "-may_be_treated_by-", the algorithm infers a considerable amount of positive instances of the beneficial relation. Interestingly, in experiment 3, this path is not considered for inferring relation instances. The other top weighted paths in experiment 2 show that the theme primarily occurs as a component of (ingredient_of, product_component_of) of the drug/treatment.

Path	Weight
-isa-prevents-occurs_in-_isa-	2.6016806945294886
-isa-treats-occurs_in-_isa-	2.6016806945294886
-isa-diagnoses-occurs_in-_isa-	1.901124167068688
-isa-prevents-_isa-harmful-	1.67024684947869
-isa-diagnoses-_isa-harmful-	1.67024684947869

Table 6: Top weighted paths in experiment 1

Path	Weight
-_may_be_treated_by-	5.048508431316066
-has_ingredient-_may_be_treated_by-	2.358357286622961
-_ingredient_of-_may_be_treated_by-	2.358357286622961
-_product_component_of-_may_be_treated_by-	2.2310561730606215
-has_product_component-_may_be_treated_by-	2.2310561730606215

Table 7: Top weighted paths in experiment 2

Path	Weight
-_has_contraindicated_drug-isa-_isa-_classifies-	0.08753923040850799
-contraindicated_with_disease-isa-_isa-_classifies-	0.08753923040850799
-contraindicated_with_disease-isa-_isa-classified_as-	0.08753923040850799
-_has_contraindicated_drug-isa-_isa-classified_as-	0.08753923040850799
-_may_be_treated_by-isa-_isa-_classifies-	0.07546827568726627

Table 8: Top weighted paths in experiment 3

6 Conclusion

Given the UMLS knowledge base, which is a repository of medical entities and relations between them, we explored the task of discovering latent instances of beneficial drugs for diseases. We employed a recent and algorithmically effective knowledge base completion algorithm Subgraph Feature Extraction to perform the task of discovering the latent relation instances. Using a particular set of filtered instances from the knowledge base, we were able to effectively discover instances of beneficial relation.

Due to memory issues, we were not able to get results of the experiments until the penultimate day and hence couldn't analyze the results comprehensively enough. As future work, we plan to perform further analysis of the results obtained.

Further, as future work, we plan to explore for baseline results to compare the above results with. We could also run more experiments by varying the BFS path length parameter and see the performance. We could also manually evaluate those relation instances output by the SFE algorithm which are not present in the test data. We could also run SFE on more different subsets of the UMLS database and the entire UMLS database and observe the results.

References

Matt Gardner and Tom Mitchell. 2015. Efficient and Expressive Knowledge Base Completion Using Subgraph Feature Extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1488−1498