Building Mental Health Knowledge Graph

Group Members: Xiaoying Zhang, Yunrui Shao

1. Project Domain and Goal

With the rapid development of the contemporary world, young people's mental health problems have gradually become one of the focuses of our society. What should we know about mental health diseases, and what should we do if we have a mental health disease? Also, instead of the actual disease, all the small annoyances in our daily life should be taken seriously to prevent any potential risks of developing into a serious problem. Considering all the concerns, we built a knowledge graph to treat mental health disorders and all the associated areas.

2. Data Crawling and Information Extraction

The data used for our knowledge graph was crawled from 3 sources. Structured mental health disease data was extracted from Wikidata by SPARQL (2467 mental diseases). Therapist/ psychiatrist data was crawled from Psychology Today (9,956 therapists in California), and drug data was crawled from Drugs.com by Scrapy (153 drugs).

After cleaning and extracting from raw data, 4,015 specialties, 4 modalities, 6 age brackets, 23 communities, 67 drug classes, and 382 drug brands were obtained. Relevant attributes were merged together to deduplicate, and all the address values were converted to latitudes and longitudes by GeoPy to compare the distances in the recommendation section.

Unlike the amount of data crawled from Psychology Today, the number of disease drugs crawled from Wikidata was small, which means instead of record linkage, we could link them with data from Drugs.com by manually searching directly. (also, non-uniform URLs from Drugs.com might cause a problem with crawling)

3. Ontology Overview

In this project, we developed our own ontology, including information about detailed descriptions of different illnesses, their symptoms, risk factors, healing drugs (including descriptions, drug class, brand-name, and warnings), and recommended therapists/ psychiatrists (including name, title, about, specialties, website, tel, address, ethnicity, therapy type, age bracket, communities, and modality).

For generally-used characteristics, we created additional relationship types to link all nodes with the same features together (e.g., drug_class and brand_class for drug nodes). Also, we created some specific linkage rules to match different values with the same meanings from different classes, and we will discuss more details about this in the technical challenge part.

4. Neo4j and Web-Creation

By importing 24 preprocessing CSV files, we used Cypher in Neo4j to construct our knowledge graph. There were 17,427 nodes, 139,662 properties, and 560,055 relationships created, and it took us around one hour to generate the whole graph. In order to connect it to our website, we used the Neo4j python driver. In the web-developing process, Flask was used as a web server, and React.js was used for the front-end development.

5. Technical Challenges

5.1 Record Linkage

To link therapists and diseases, we need to link specialties from Psychology Today and diseases from Wikidata. Specialties in Psychology Today contain typos, self-defined issues, and unrelated words (eg. www.mychildadhd.com), while diseases in Wikidata are over jargon (eg. 11p15.4 microduplication syndrome). With a total of 4,016 specialties and 2,468 diseases, it's a great challenge to match all those mess strings.

In order to find the best threshold and the proper string-matching algorithm, we built a ground truth dataset. Which string pairs should be selected as ground truth was crucial. First, we used Needleman-Wunsch similarity as calibration to choose our pairs. Nearly all algorithms could recognize pairs with similarity under 0.4 as "totally different" and pairs with similarity over 0.95 as "highly similar"; so, mostly looking for pairs with similarity scores in the middle (between 0.6 and 0.8) could make sense to compare the performances of each algorithm. For example, the similarity of "suicidal ideation" and "Homicidal ideation" is 0.763889. This similarity is high, but they are not the same. 0.619048, the similarity between 'somatic disorders' and 'somatization disorder', is lower than the previous pairs, but the two words are the same. By using this strategy of picking more pairs in the middle, we generated a 294 entries ground truth dataset. By evaluating all string matching algorithms with the ground truth, we chose Jaro Winkler as a further algorithm based on its best performance for our data (All performance results are shown in Table 1 in the Appendix).

After applying the string matching method between specialties and diseases, only 10 percent of the diseases could be linked to specialties. The sparsity linkage promoted an alternative solution: creating linkage SIMILAR_AS from specialties in Psychology Today to symptoms in Wikidata. This solution would largely enrich the connected relationships between therapists and diseases.

In this project, we did not use clustering or community detection because these methods may destroy the accuracy. However, we also lost some candidate pairs because of this. A more suitable algorithm should be used in our project for better performance in the next step.

5.2 Customized Recommendation

The user-customized recommendation is an important section of our website; it allows users to input their recent troubles and symptoms, and it will return all the recommended matched therapist lists for them to talk with. In this case, how to let computers understand exactly what problems the users have, and how to recommend suitable specialists became the greatest challenge for us.

To achieve our goal, first, we picked the 31 most-frequent issues which covered 71% specialties from the specialties' list of our therapists, and created vocabulary lists for each of them. We collected symptoms and related descriptive sentences of each issue (from trustworthy sources like Wikipedia), removed all punctuations and meaningless words (e.g. pronouns like 'you', 'they', prepositions like 'in', 'at', 'on', and other frequent neutral terms like 'feel', 'feeling'), used the repetition of each word in the list to record the normalized weight (e.g. if a word 'sad' appeared 3 times in a 120 length word list, we would record its weight as 3/120), and saved them into a python dictionary (with keys of words and values of weights). Then, we dealt with the user input data in similar steps and generated another user python dictionary.

By using Jaro similarity with a threshold of 0.9, we got the similarity score for each matched word from the user dictionary, and by timing this with the weighted score, we were able to get a ranking of all the possible related issues. On our website, we only recommended therapists who are specialized in the top 3 predicted issues.

To evaluate this challenge, we collected 19 users' input data, then labeled these data by psychology major students as the ground truth, and recorded our top 10 outputs as predictions. We used two ranking-based metrics: Recall@R and truncated normalized discounted cumulative gain (NDCG@R). We compared the predicted rank with the manual labels for each user. By taking the R values of 3, 5, and 10 respectively, Table 2 in the Appendix summarized the average results of the 19 data, with the general criterion of the R higher the performance better. Recall@10 is higher than 0.5, which means we got a good result in the top 10 predictions. But the NDCG metric was not good enough, which means the top predictions are not the most accurate. This might be because of the potential biases inside the vocabulary lists. If the data we collected from different sources tend to repeat the same words, it would significantly increase the weight of those words, and ultimately cause a gap in the final result. Later, we could further improve the algorithm by limiting vocabulary lists from the same sources.

Appendix

String Matching Approach	Precision	Recall	F1 score	tp	fp	tn	fn	Best Threshold
Levenshtein	0.8275862069	0.5853658537	0.6857142857	24	5	248	17	0.79
Needleman wunsch	0.78125	0.6097560976	0.6849315068	25	7	246	16	0.7
Smith waterman	0.5185185185	0.6829268293	0.5894736842	28	26	227	13	0.81
Jaro winkler	0.8181818182	0.6585365854	0.7297297297	27	6	247	14	0.91
Jaccard	0.6896551724	0.487804878	0.5714285714	20	9	244	21	0.67
Hybrid jaccard	0.8181818182	0.4390243902	0.5714285714	18	4	249	23	0.89

Table 1 Results of String Matching Experiment

Recall@3	0.1842105263
NDCG@3	0.1954952513
Recall@5	0.3192982456
NDCG@5	0.2673461029
Recall@10	0.5114035088
NDCG@10	0.3518035094

Table 2 Results of Recommendation Experiment