iNLP Project: Final Report

Rudransh Agarwal (2021101033)
Akshit Sharma (2021101029)
Pranit Khanna (2021101057)

May 2024

Team: 7
Title: Hypernym Discovery

1 Introduction

Hypernym Discovery: Hypernym discovery refers to the process of identifying higher level categories or broader terms that encompass more specific concepts or entities. Hypernyms are words that represent categories or classes which other words (hyponyms) belong to. For example, "animal" is a hypernym of "cat" and "dog".
We use the term hyponym similarly to the sense used in (Miller et el. 1990): a concept represented by a lexical item L_0 is said to be a hyponym of the concept represented by a lexical item L_1 if native speakers of English accept sentences constructed from the frame: a L_0 is a (kind of) L_1. Here L_1 is the hypernym of L_0 and the relationship is reflexive and transitive, but not symmetric.
Traditionally, the task of identifying hypernymic relations from text corpora has been evaluated within the broader task of Taxonomy Evaluation. Alternatively, many approaches have been specializing on Hypernym Detection, i.e. the binary task consisting of, given a pair of words, deciding whether a hypernymic relation holds between them or not. This experimental setting has already led to criticisms regarding its alleged oversimplification.

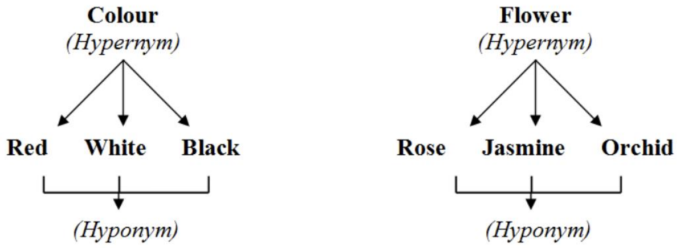


Figure 1: Few examples of hyponym-hypernym relationship

2 Data Collection and Pre-Processing

We downloaded the 18GB UMBC corpus and tokenised it (removed punctuation as well) using the nltk library in python. It is a 3 billion word corpus containing 100 million web pages from more than 50,000 websites. It contains 408 files with paragraphs extracted from web pages, one to a line with blank lines between them. A second set of 408 files have the same paragraphs, but with the words tagged with their part of speech.

Since our task involves predicting hypernyms upto trigrams, we replace all the trigrams and bigrams provided to us in the vocabulary with underscore(_) separated words in the entire corpus as well as in the train, test and val data. For example, we have "anticoagulant medication" in the vocabulary, we find all the instances of this in the corpus and replace it with "anticoagulant_medication" After this, we generated word embeddings using Word2Vec (skip gram with negative sampling).

3 Hearst rule based discovery

This method is from a 1992 published paper by Hearst wherein many rules are mentioned for extracting hypernym-hyponym pairs by finding out words (referred to as x and y below) that follow certain patterns such as the following:

- such x as y, eg. such sports as football
- x like y, eg. nations like France
- x including y, eg. colors including red
- x especially y, eg. sectors especially manufacturing
- x and other y, eg. BMW and other manufacturers

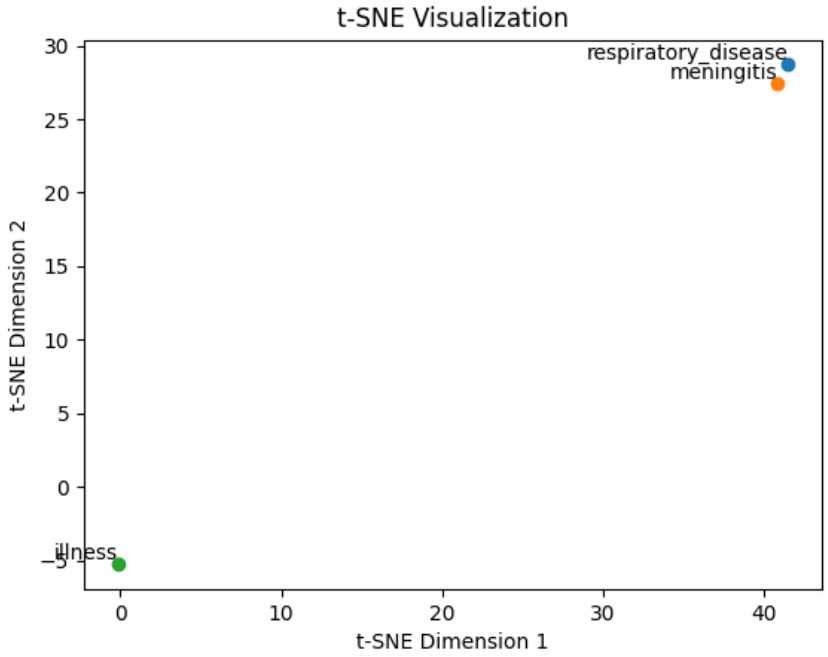
We have used a regex based approach to extract relevant substrings from the larger corpus. These rules are not stringent and do not give direct pairs so this can be followed by manual evaluation. The problem with rule based approach is that hypernym-hyponym pairs seldom occur in pairs, for example one would mention sports as football but it would seldom happen that one would mention sports as football, cricket and make an exhaustive list, so all hypernym-hyponym pairs can't be captured with the rule based approach. Thus we try to use distributional embeddings based approach which also capture the meanings of words.
There will be many cases where the hypernyms will not be discovered because their hypernym-hyponym relationships are not present in the training corpus, Example- spelunker. On the other hand, there might be cases with hundreds of hypernyms. Example- reflex. Thus, we come up with a distributional semantics approach in the next section.

Example results:

- Query- staphylococci
  Gold- true bacteria eubacteria bacteria micro-organism
  Predicted- foodborne microbes infants germs products isolated exotoxins infections flora bacteria agents sweethearts cocci evidence staphylococci pathogens

4 Nearest Neighbours: A Statistical Approach

Embeddings derived from word2vec capture the meanings of the words based on their context. Co-hyponyms can be defined as words with the same hypernym. For example, cat and dog are co-hyponyms with the same hypernym that is "animal". Since these co-hyponyms have a very high chance of appearing in the same kinds of contexts, their embeddings will also have a a considerably high similarity score. For example in our dataset, we have "respiratory disease" and "meningitis" as co-hypnyms with the same hypernym illness and their embeddings have a cosine similarity of 0.5. Also, co-hyponyms tend to have the same hypernyms for example, both the above will also have "disease" as their hypernym. Thus given a new word, we can find the co-hyponyms for it and then consider the hypernyms for those co-hyponyms as candidate hypernyms. As we can see in Figure 2, respiratory disease and meningitis are co-hyponyms, on projecting the embeddings to a 2D space, we notice that, the co-hyponymns are pretty close to each other. And illness which is their corresponding hypernym, is almost equally distant from both. Thus, the hypernyms for co-hyponyms server as good candidates for the query's hyponyms.



4.1 Procedure

1. First find top K co-hyponyms. We do this by taking the embeddings of train data and picking the top k words with highest cosine similarity.

2. Then we take the hypernyms for these co-hyponyms and calculate the cosine similarity with the candidate query.

3. Finally, we take the top 15 hyponyms with the highest similarity score.

4.2 Results

Here we report the metrics on the test set (in %):

- MRR: 14.34
- MAP: 7.16
- P@1: 9.66
- P@3: 7.84
- P@5: 7.19
- P@15: 6.62

P@k simply calculates the precision of the top k hypernymns, by checking how many of them are relevant. Looking at our P@1 metric, we can clearly see that our approach works well with the most relevant(top hypernym), that is even if we choose one hypernym from our outputs, in many cases it is relevant.

4.3 Observations

Some examples of the given gold and our generation:

- Query: maliciousness
  Gold: malevolence,distaste,hatred,hate,malignity
  Predicted: falsehood,prevarication,hatred,distaste,untruth,hate,unskillfulness, unfaithfulness, wrongdoing, falsity, punishment, offence, disgust, fictitious, character, natural virtue
- Query: quo warranto
  Gold: proceedings,legal proceedings, proceeding, due process, legal proceeding
  Predicted: action at law, proceeding, legislative act, statute, legislative body, magistrate, judge, proceedings, judgment, pleading, legal proceeding, proscription, judicial opinion, judicial decision, legal proceedings
- Query: buckler
  Gold: body armor
  Predicted: sword,armour,suit of armour, suit of armor, horse, bond servant, truncheon, coat, workman, outer garment, potentate, bludgeon, horsie, king, weapon
- Query: bread
  Gold: foodstuff, food product
  Predicted: soup,dessert, salad, meal, pastry, stew, sweets, hot cereal, recipe, sweet, cereal, breakfast cereal, dish, breakfast food, pastry dough

As we can see in the first two examples, our algorithm correctly predicts almost all the hypernyms that needed to be present. The third example is interesting, as the original gold text is supposed to be body armor, but the predicted text from our algorithm consists of words like suit of armor and armour, although these are related enough to the real gold text,

they are not a part of it. This shows that the search space of our algorithm has been limited as body armor might not appear in any of the gold for train data. Hence we will come up with algorithms that have better candidate hypernyms. As we can see in the fourth example, for bread, we don't just get the hypernyms, but also words that are similar to bread, this happens due to the fact that we only consider cosine similarity and do not learn whether the relation we are learning is actually hypernym-hyponym or just a high correlation due to similar distributional semantics. This motivates a neural networks based approach which we define in the next section.

5 Deep Learning Approach

We plan to capture hypernym-hyponym relationship by training a neural network architecture to perform a binary classification task on whether the two words have a hyponym-hypernym relationship.
A negative sampling technique is employed to obtain negative samples in addition to the ground truth provided in the gold data. This model will then take word embeddings and these samples and optimize on a logistic loss function.
In order to predict the hypernyms for test data, we provide a set of candidate hypernyms (or the entire vocabulary itself), and choose the top 15 hypernyms predicted by the model.

5.1 Results

On using different number of negative samples per positive sample, we get different evaluation metrics. These are given below:

Negative Samples=100

- MRR: 13.1
- MAP: 6.3
- P@1: 8
- P@3: 6.7
- P@5: 6.2
- P@15: 5.9

Negative Samples=1000

- MRR: 25
- MAP: 11.6
- P@1: 20
- P@3: 12
- P@5: 11
- P@15: 10

Negative Samples=5000

- MRR: 25.05
- MAP: 11.9
- P@1: 20.4
- P@3: 12.9
- P@5: 11.3
- P@15: 10.6

As we can see, the results with n = 100 are not very great, whereas results with n = 1000 and n = 5000 basically beat the previous model by a huge margin. Let us look at an example to understand (n is number of negative samples for training the model):

Query- footway
Gold- walkway, path

Predicted with n=100 - roads, public building, pedestrian traffic, thoroughfare, roadway, highway system, vehicular, local road, vehicle traffic, freeway, traffic congestion, vehicular traffic, private road, toll road, traffic flow
Predicted with n=1000 - passageway, means of transport, railway track, air passage, superhighway, flight of stairs, obstruction, bicycle, motorway, transport infrastructure, footpath, bicycle parking, elevators, sidewalk, access road

Clearly, we can see that the output with n = 100 consists not of hypernyms but just words which would appear in similar contexts like traffic regulations etc. Whereas the one with 1000 are much more sensible, as some might not be in gold but are actually hypernyms like transport infrastructure, passageway, motorway etc. There is a simple explanation, with just 100 negative samples, model does not get enough context of what is a hypernym and what is not. Words which are similar might not be even going into the negative samples, but with 1000 negative samples, the model gets much more sensible negative samples. After this it reaches a saturation and no significant improvement happens when n = 5000, this is probably because all the context it could receive about what a hypernym could and could not be is already provided with 1000 samples.

5.2 Observations

Some examples of the given gold and our generation:

- **Query**- respiratory disease
  Gold- illness, sickness
  Predicted - sickness, contagious disease, disease, illness, health, medicine, communicable disease, disorder, medical literature, communicable diseases, ailment, chronic, diagnosis, animal, infection
- **Query**- luffa
  Gold- seed plant
  Predicted - plant part, plant, citrus fruit, vegetable, foodstuff, plant material, animal, edible, seed, bananas ,cannabis sativa, food product, flavouring, herbaceous plant, animal tissue

6 LLM based approach

This section will talk about an LLM based approach. We prompted Google's Gemini using its API by chunking 10 examples at a time for guaranteeing a proper response without exceeding the limits of the API. We prompted it with the following - "Find exactly 15 numbered hypernyms for each of the words in the list: " + y + " and only the hypernyms with no extra text for each word in the same order and newline separated and also labelled with the actual word before it" where y is a chunk of 10 examples. The motivation behind this was to use the already massive pretrained LLM models and their vocabulary to see whether an aimed approach performs better or is general language understanding more useful.

6.1 Results

With Zero-Shot Learning

Query: maliciousness

Response 1. Vice 2. Evil 3. Malevolence 4. Depravity 5. Wickedness 6. Sinfulness 7. Iniquity 8. Corruption 9. Immorality 10. Heartlessness 11. Cruelty 12. Barbarity 13. Savagery 14. Brutal_force 15. Ruthlessness

The above was an example of a good output from the LLM. Below is presented a bad example wherein several of the answers given by the LLM are actually hyponyms and not hypernyms, for eg-

Query: Wine

Response 1. Alcoholic_beverage 2. Drink 3. Fermented_grape_juice 4. Grape_wine 5. Red_wine 6. White_wine 7. Dessert_wine 8. Fortified_wine 9. Sparkling_wine 10. Chardonnay 11. Cabernet_Sauvignon 12. Merlot 13. Pinot_Noir 14. Sauvignon_Blanc 15. Zinfandel

Here white wine, red wine, etc are types of wines making them hyponym instead of hypernym.

With One-Shot Learning

Query: woofer

Response 1. speaker 2. audio_device 3. loudspeaker 4. subwoofer 5. woofer 6. transducer 7. sound_system 8. audio_component 9. electroacoustic_transducer 10. audio_equipment 11. sound_reproduction_device 12. sound_transducer 13. bass_speaker 14. sound_driver 15. audio_driver

The same query for **zero-shot learning gives**:

Response 1. speaker 2. loudspeaker 3. transducer 4. driver 5. sound_device 6. audio_device 7. woofer

r We see that with zero-shot learning, the LLM did not give 15 hypernyms, even when asked too. But, in one-shot case, we get 15 hypernyms. Also, we observe that the quality of response itself is much better in one-shot case with hypernyms like sound_reproduction_device, electroacoustic_transducer, etc. being predicted. This is kind of expected from one-shot learning as the LLM has seen an example of what is expected from it in the prompt and is thus able to apply its knowledge in a better way, compared to zero-shot.

6.2 Evaluation Metrics

With Zero-Shot Learning

- MRR: 38.49
- MAP: 19.5
- P@1: 31.06
- P@3: 21.84
- P@5: 18,97
- P@15: 17.35

With One-Shot Learning

- MRR: 42
- MAP: 21.47
- P@1: 35.3
- P@3: 23.9
- P@5: 20.8
- P@15: 18.96

6.3 Drawbacks Observed

There were a few drawbacks of using LLMs as well apart from the one presented above-

- In some cases, it was not giving 15 hypernyms even when asked to.
- In some cases, it was giving the query word itself as its hypernym.
- In some cases, it even repeated hypernyms for a query.

7 References

- Hearst Paper: Automatic Acquisition of Hyponyms on Large Text Corpora
- UMBC dataset
- SemEval-2018 Task 9: Hypernym Discovery
- Gemini Api Documentation