

Ontology based Algorithms for Indexing and Search of semantically close Natural Language phrases

Srikanth Kamath U

Department of Information Technology

National Institute of Technology

Surathkal, India

usrkamath@gmail.com

Abstract

Free text constitutes a overwhelming fraction of information available on the World Wide Web. Specifically, consider small chunks of natural language phrases frequently used by Web users to describe stuff relevant to them. For example, consider the following two posts on a classifieds site (which serves a small locality, say, a university campus) - "2 Tickets for the prom tonight" and "Trade 2 extra passes for tonight's Ball for \$25". For a human looking at these two posts, its trivial to conclude that he has found what he wanted. But when there are thousands of such posts and in the absence of any common keywords or any additional information from the user it is unlikely that naive keyword based matching will be of any help in reflecting the glaring similarity between these descriptions. This problem is very relevant and challenging because users tend to describe the same item in several different ways. Humans frequently use their common-sense and background knowledge to infer that these relate to the same item. However the enormous sizes of most datasets prohibit manual classification. To automate this, we present intuitive and scalable algorithms which use existing Ontologies like WordNet to correctly relate semantically close descriptions.

1 Introduction

Search is a ubiquitous tool that we come across on the WWW. It is found on almost every portal, discussion forum, classified sites etc. People might wish to search at different levels of granularity which can range from a web page or just short phrases or sentences. Google PageRank [Page *et al.*, 1998] and other related algorithms work vey well when we deal with web pages. But when we are dealing with the phrases, the currently used technology is very simple and naive. It merely looks up all the occurrences of the set of keywords provided by the end user(in some cases boolean operators are allowed too). This works well in certain situations where the entity being searched is written in some canonical form - say the title of a book - "Harry Potter and the half blood prince". There is little scope for ambiguity here. But there are situations where

people tend to use more freedom in their descriptions. Consider the example of people selling furniture on a classifieds site. Consider two posts "Antique table for sale" and "Old wooden furniture in good condition". The need for common-sense powered search is clear from this example. With this motivation, we present, novel algorithms and their effectiveness by presenting results on a real world dataset.

1.1 Previous work on this area

Primary motivation for this paper has been the ongoing work on commonsense reasoning at the MIT Media Lab [Lieberman, 2003]. Specifically ConceptNet[Liu and Singh, 2004] and its applications. Though none of these directly address the problem being addressed in this paper they have provided useful insights and approaches solving related problems in building commonsense powered systems.

In Section 2 we present the Framework and Notations used in our experiments. This is followed by Section 3 where we present the mathematical analysis of our approach. We present the results obtained by using our algorithms on a real-world dataset in Section 5 and conclude with a short note on proposed future work in Section 6

2 Framework and Notations

We use WordNet[Fellbaum, 1998] as the Ontology required to perform semantic analysis. We use archives from MIT Reuse Ask/Sell mailing list as our dataset. It is available to the public at <http://diswww.mit.edu/barb/reuse-sell> (Sell List),<http://diswww.mit.edu/barb/reuse-ask/> (Ask List). These mailing lists are used to promote reuse of commonly used stuff within the campus. Each email contains description and other details of an item. Specifically the Subject line contains short description about the item. We stripped off these Subject lines from the archives and amassed about 3000 descriptions. These items are not restricted to a specific category, instead, they range from anything like "Used hiking gear" to "Kitchen appliances" and from "Computer Accessories" to "Concert Tickets". With these items in the index, a query interface is setup to fish out all items that are semantically close to the query specified.

WordNet is made up of synsets, s , each of which is a particular sense in which a word/phrase is used. Let S be the set of all synsets present in WordNet. Let e denote an entry in the index which corresponds to a short sentence describing

an item. Let E denote the set of all entries in the index. Each entry e is made up of tokens t and let T denote a set of these tokens. At this point we assume that stop words have already been eliminated and t denotes only those tokens which are known to WordNet. Other tokens such as proper nouns are treated separately. For convenience we denote i^{th} entry in the index as e_i , and the set of tokens corresponding to this entry as T_i . Further the j^{th} token in T_i is denoted by t_{ij} . $P(S)$ denotes the power set of S . The query posed by the user is structurally similar to an index entry and is denoted by Q . Hence all operations valid on e_i are also valid on Q .

3 Mathematical analysis of our approach

We define a relation *Semantically Related*, SR as

$$SR : T \times S \longrightarrow \{0, 1\} \quad (1)$$

$$SR(t, s) = \begin{cases} 1 & \text{if } s \text{ is a synset of } t \text{ (directly or indirectly)} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Stop words are eliminated and raw tokens are stemmed, if necessary, by WordNet during Morphological Preprocessing which is the first step. Note that, for a token t , the only part of speech considered is NOUN. This is reasonable since the grammatical subject of the sentence will be a noun and the subject of the sentence is what we are interested in.

Further, s can occur, either as a direct synset of t OR linked to one of its direct synsets via Hypernym/Hyponym relation. An Hypernym is a general term used to refer to a set of items. For example, furniture is a Hypernym, for "table", "chair", "couch", etc. Hyponym, on the other hand is a specific instance of a Hypernym. Continuing with the same example, "table", "chair", "couch" are among the Hyponyms for "furniture".

Another important aspect that needs to be clarified here is the type of links explored in the Wordnet Ontology. This is entirely dependent on the goal that we wish to achieve. For our purposes, the Hypernym/Hyponym pair does the job effectively and efficiently too because we are exploring only these two types of links among all the types available. This is based on the following simple observation - If a person specifies a hypernym in his query, there is a strong possibility that he will also be interested in entries which contain the hyponym instances for that hypernym. Using the earlier example, people interested in "buying/selling furniture" are likely to be interested in "buying/selling chair/tables/couch".

This is the sole part of the analysis that is ontology dependent. Each Ontology will have its own sets of links and larger the set of links used for analysis, larger will the resulting search space be. One may start with a basic set of links, and then more types of links, if the results are not satisfactory enough.

3.1 Indexing

It is necessary; to analyze and index the existing entries in the corpus once, before querying can begin. If a new item is added to the corpus, then this new item alone needs to be indexed and will not affect the existing entries in the index.

Similarly deleting an entry from the corpus will require corresponding entries to be deleted without affecting the rest of the index. The crux of indexing is a function *Activation Set*, A , which is defined as,

$$A : T \longrightarrow P(S) \quad (3)$$

$$A(t) = \{s | s \in S \wedge SR(t, s)\} \quad (4)$$

We extend this function to sets of tokens, T as

$$A^*(T) = \bigcup_j A(t_j) \forall t_j \text{ in } T \quad (5)$$

Hence the *Activation Set* for a given index entry e_i is given by $A^*(T_i)$. Activation sets of related tokens within the same item often contain common synsets, $A(t_i)$ and $A(t_j)$ are not necessarily disjoint. This leads us to the concept of frequency of a synset in the activation set of an item e_i . We define the frequency, f , as follows,

$$f(s, T_i) = \sum_j h(s, t_{ij}) \forall t_{ij} \text{ in } T_i. \quad (6)$$

where $h(s, t_{ij})$ is given by

$$h(s, t_{ij}) = \begin{cases} 1 & \text{if } s \in A(t_j) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Similarly, the frequency of synsets in the query Q is given by $f(s, T_Q)$, where T_Q is the set of tokens present in the query.

During the experimentation, we noted that in addition to *Frequency* we need to compute another related entity which we term as *Relative Frequency*. We denote it as f_r . This is defined as follows,

$$f_r(s, T_i) = \frac{f(s, T_i)}{\max\{f(s_j, T_i)\}} \quad (8)$$

where s_j is a synset in $A^*(T_i)$. The relevance of Relative Frequency is explained as follows, for a given entry, each time a token triggers a specific synset, its frequency w.r.t to that synset increases. The most relevant frequency can be expected to have the highest frequency. Suppose that, we wish to compare the relevance of a given synset in two entries. If frequency is used, we may get misleading conclusions since the range over which frequencies span can be different for each entry. Hence the frequency has to be normalized, which is achieved by computation of Relative Frequency.

Indexing phase involves computing and storing these three entities, namely *Activation Set*, *Frequency* and *Relative Frequency*. These will be used during the Search and Ranking phases.

3.2 Search

To Search the items that match a given query we define a relation *Reverse Activation*, RA , as follows

$$RA : S \longrightarrow E \quad (9)$$

$$RA(s) = \{e | s \in A^*(e)\} \quad (10)$$

Reverse Activation of a given synset all the entries which contain that synset in their *Activation Set*. Again we may extend *RA* to get *RA** which can be defined as

$$RA^*(S') = \bigcup_j RA(s_j) \quad \forall s_j \in S' \quad (11)$$

where S' is a subset of S .

3.3 Result set and Ranking

Using the definitions of *Activation Set* and *Reverse Activation* we can obtain the set of all entries forming the Result Set, R , it is given by

$$R = RA^*(A^*(Q)) \quad (12)$$

where Q is the query posed. Applying $A^*(Q)$ on the query gives the set of synsets in the activation set for that query Q . Applying RA^* on this gives all the entries which contain one or more of these synsets in their *Activation Set*.

Once the Result Set, R , is obtained, it is necessary to rank the entries in R to reflect the varying degrees of similarity between the query and an entry in R . This is achieved by defining a score metric, $SC(e_i)$ for every entry as follows,

$$|C_i| \times \left\{ \sum_t f(s_{it}, T_Q) - \sum_t (|f_r(s_t, T_i) - f_r(s_t, T_Q)|) \right\} \quad (13)$$

where C_i is the set of synsets which are common to Activation sets of Q and e_i ,

$$C_i = A^*(Q) \cap A^*(T_i) \quad (14)$$

3.4 Intuitive Justification of the Score metric

It is worthwhile to note that the term,

$$\sum_t (|f_r(s_t, T_i) - f_r(s_t, T_Q)|)$$

has its minimum value, which is zero, for an entry that matches the query verbatim. Also if an entry shares with the query, those synsets with high relative frequency, then its score will increase. Similarly large values of C_i indicate that the entry under consideration and the query have a large number of synsets in common and hence it has higher relevance in the results.

4 Algorithms

The Indexing Algorithm

```
INDEX(e)
begin
  T = TOKENIZE(e)
  for each t in T do
    begin
      X = A(t)
      for each s in X do
        begin
          update frequency f
          track fmax among f
        end
      end
    end
  end
```

```
for each s in X do
begin
  update relative frequency
  using fmax
end
end
end
```

It may be noted that, if there are N entries to be indexed the time and space complexities of this algorithm are both $O(N)$.

The Search and Ranking Algorithm

```
SEARCH_AND_RANK(Q)
begin
  R = RA(A(Q))
  for each e in R do
    begin
      compute score SC
    end
  Sort R using SC
end
```

A and RA in the above algorithm refer to Extended *Activation* and *ReverseActivation* functions defined in equations 5 and 11 respectively.

It would be worthwhile to note that the entire universe of synsets S is not examined during search. Instead a subset $S' = A(Q)$ alone needs to be considered. This speeds up the search algorithm considerably. Hence the time complexity of this search algorithm will be $O(|R|)$.

5 Results

Java based API for Wordnet was used in the implementation of the algorithms presented in the previous section. This section presents the effectiveness of our algorithms by presenting sample results obtained using the Reuse Ask/Sell corpus. Each result entry is followed by the score, as computed by Equation 13

Query: "Computer Accessories"

- New Sony Memory Stick Media 256MB (Score=2.6)
- Konica 2400W color Laser Printer(Score=1.4)
- Looking for a working 15" LCD monitor(Score=0.7)
- Intel 845GLV motherboard (Score=0.7)
- two 512MB DDR2 SODIMMs IBM Laptop memory(Score=0.7)
- request for USB webcams (Score=0.7)
- Photoshop software (Score=0.4)
- Any (musical) keyboard (Score=0.2)

Query: "Used furniture"

- Mattress, boxspring, and bed frame (Score=2.0)
- Wooden Draft table (score=1.5)
- Filing cabinet (Score=1.0)
- Red Couch and freestanding coatrack (Score=1.0)
- Large brown bookcase (Score=1.0)

- need to rent/buy a cot (Score=1.0)
- guitar, keyboard stand (Score=1.0)

Query: "Camping Gear"

- Camping gear for sale (Score=30.0)
- backpack and hiking equipment (Score=10.0)
- Tent available sleeping bag and mat (Score=2.75)

Query: "Fish Bowl"

- 20 & 30 gallon aquariums, 24" brand new aquariums (Score=2.0)
- brand new fishbowl for sale (Score=1.5)
- need to borrow fishing gear for weekend (Score=1.5)

Query: "Unused LAN cable for sale"

- spare ethernet cable needed (Score=2.5)
- brimfield elementary needs network switches (Score=2.0)
- looking to buy a used modem (Score=2.0)

6 Future work

We plan to focus future work on the following key issues, first, to demonstrate that these algorithms are generic by validating them using other datasets. Secondly, to automate ontology learning, we have used existing ontologies here, but for a given dataset we may have to fine tune the ontology to reflect the slang terms used in that domain.

Acknowledgments

The work presented here, was done as part of senior year project work in college. Special thanks to Mr Himanshu Nautiyal, Bixee Inc, for his guidance and insightful comments.

References

- [Brin and Page, 1998] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107-117, 1998.
- [Cilibrasi and Vitanyi, 2004] R. Cilibrasi and P. Vitanyi. Automatic meaning discovery using google, 2004.
- [Fellbaum, 1998] Christiane D. Fellbaum. *WordNet - An electronic lexical database*. MIT Press, 1998.
- [Lieberman *et al.*, 2004] Henry Lieberman, Hugo Liu, Push Singh, and Barbara Barry. Beating commonsense into interactive applications. *AI Magazine*, 25(4):63-76, Winter 2004.
- [Lieberman, 2003] Henry Lieberman. Common sense reasoning for interactive applications. 2003.
- [Liu and Singh, 2004] H. Liu and P. Singh. Conceptnet - a practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4):211-226, October 2004.
- [Page *et al.*, 1998] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.