WIKINET: WIKIPEDIA AS A NETWORK

CHEN ZHANG (CZ1389) & GUANG YANG (GY552)

ABSTRACT. We present the prototype of WikiNet, a relation-based search engine for Wikipedia. A heuristic crawler is built to colloect a small propotion of wikipedia pages. Using the the structural infomation and content of the pages, we model the dataset as a directed weighted graph. Based on this graph, WikiNet carries our similarity based search. So far, two functions are implemented, namely, to search a page's most related pages, and to search the most related path between a pair of pages. Analysis show that WikiNet provides much higher-quality relations between wikipedia pages than naïve BFS strategy.

1. INTRODUCTION

Wikipedia is so far the largest public online database of knowledge. When searching over wikipedia, it is almost certain that there will be a page excatly match the query. Aside from the incredibly high precision, wikipedia also enjoys a high degree of diversity. The wikipedia articles are related to other pages through the numerous hyperlinks. When reading wikipedia articles, people always follow the links to some other related topics(usually in a depth-first pattern). At the end of the day, we can be very far from what we intended in the first place.

The quality of the relation between connected pages varies drasticly. If a link with poor relation is followed, people may be diverted into a very different field.

That is where WikiNet starts. We seek to search for high quality relations in wikipedia. In Section 2, we introduce a heuristic crawler that help us collect reasonable amount of kind-of-related wikipedia pages. In Section 3, we modeled our dataset as directed weighted graph based on the structural information and the content of the pages. In Section 4, the function of WikiNet is introduced. We conclude in Section 5, where we raised an evaluation measurement and discussive the performance of WikiNet against the naïve approach.

2. THE WIKI PART: HEURISTIC CRAWLER

Because of our limited storage and computation resource, it is not feasible for us to traversd and index the whole wikipedia. As a result, heuristic methods are heavily used.

Wikipedia pages have in- and out-links to other pages. Some links are pointing at a highly related page, while others are not. (i.e. The page of apple the fruit, has a hyper link to Adam and Eve. However, the word of "apple" doesn't even appear on the latter. In fact, it is apple the symbol that is closely related to Adams and Eves.) So when crawling wikipedia pages, we conduct for- and backwards evaluation and only index the highly related pages.

The algorithm of the heuristic crawller is shown as Fig. 1. In order to guarantee that only the related pages are indexed, we perform forward and backward check.

Forward: In order to judge if a $hyperlink$ is important in page v, we devloped a scoring mechanism. Each occurrence of the anchor text in the introduction part counts for 3 point, while others count for 1 point. $Hyperlinks$ that score **more than** 1 **point** is considered realted. The related pages are added into the Queue

```
HEURISTICCRAWLER(seed, maxDepth):
  QUEUE U
  SET V
  U.PUSH(⟨seed, 0⟩)
  While U.NOTEMPTY():
    ⟨p, depth⟩ ← U.Pop()
    If p is closely related to its parent:
      V.INSERT(p)
    If depth < maxDepth:
      For all hyperlink in the content of p:
        If hyperlink is important in p:
          U.PUSH(hyperlink)
  Return V
```

FIGURE 1. Heuristic crawler alogrithm

Backward: The forward check guarantees that the page being visited is some how related to its parent. We perform a backward check to guarantee the mutual relation. Again, a scoring mechanism is introduced. Each link back to its parent page counts for 3 points, and each reference to the title of its parent counts for 1 point. Only the pages (except the seed) that scores **at least** 3 **points** are indexed. If a page is found irrelevant, it is droped immediately along with the hyperlinks in it.

This stragety is not perfect. For example, it is common sense that deep learning has a lot to do with Yann LeCun. However, the former wasn't mentioned in the content (literally, not a word) of the latter. In this case, when visiting Yann LeCun and perform backward check, the page will be marked as irrelevant. Actually, the wikipedia page of Yann LeCun is of such a low quality that it is almost isolated.

We limit the depth of the BFS crawling to 4 and crawled pages from 4 different seeds. A brief discription of the

seed	# visted pages	# indexed pages
https://en.wikipedia.org/wiki/Apple	13956	6619
https://en.wikipedia.org/wiki/Apple_Inc.	14929	6050
https://en.wikipedia.org/wiki/Artificial_intelligence	27123	12682
https://en.wikipedia.org/wiki/New_York_University	17688	6769

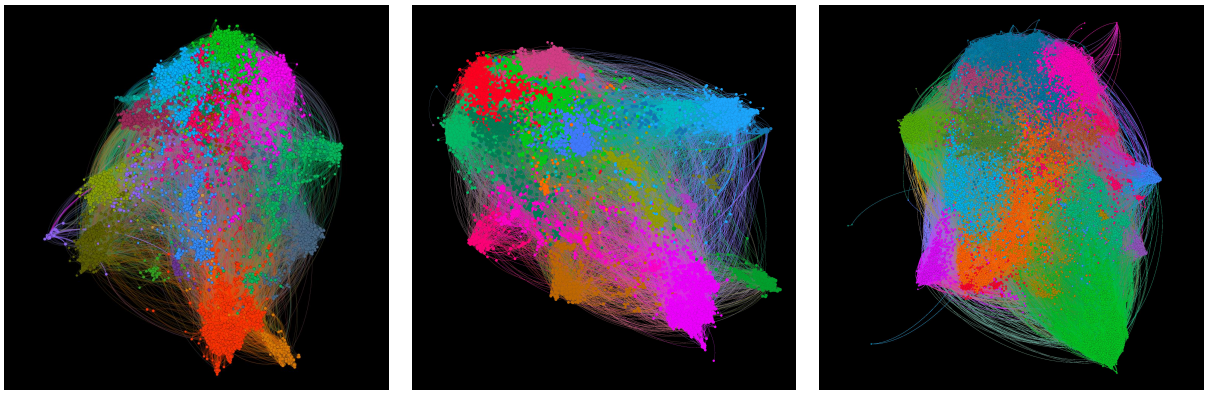
TABLE 1. A discription of the dataset we collected.

We only conduct experiment on the two apples. You may search using the keyword "apple" to find out which one is currently running.

3. THE NET PART: DIRECTED WEIGHTED GRAPH

Now that Wikipedia is an online database, the system is by nature a complex network, where artibles are the nodes, and hyperlinks the edges. We build a graph from the dataset. We used TF-IDF to index the main content of the pages and marked the weight of the edges using the cosine

coefficent between the two pages. We end up with a directed weighted graph. We visualized the graph resulted from 3 of the 4 datasets.



(A) apple (B) Apple Inc. (C) Artificial_intelligence

FIGURE 2. This is an illustration of the graph corresponding to the 3 of the dataset. Nodes of the same color fall into the same cluster based on a modularity-based community detection algorithm (the Louvain method).

Finding the most related page or path is similar to the Longest Simple Path problem, which is NP-hard. Luckily, in our case, the weight of the edges are between 0 and 1. If we define the weight of a path as the multiplication of the cosine coefficent of consecutive pair of pages on the path,

$$W(\mathcal{S}) = \prod_{i=0}^{length(\mathcal{S})} w_{i,i+1}, \text{ where } \mathcal{S} = \langle v_0, v_1, \ldots, v_{length(\mathcal{S})} \rangle,$$

which is a reasonable definition, the weight of a path decreases monotonously as we traverse the graph. This definition of the weight excludes loop in the path by definition. In this case, a smart modification to the widely-used all pair shortest path algorithm is sufficient to solve our problem. We therefore define the closeness between a pair of pages as the largest weight of all paths between them.

$$D(v_i, v_j) = \max_{\mathcal{S} \in \{\langle v_i, \ldots, v_j \rangle\}} W(\mathcal{S})$$

The algorithm to solve the problem is given in Fig 3. It is based on the Floyd-Warshall algorithm.

4. THE WIKINET

WikiNet is a relation-based search engine for wikipedia, where the records are too finely ground to tolerate any ambigurity, where diversity is implemented through the links between related articles. WikiNet makes uses of both the structure of wikipedia, and the content of it to carry out searching for relation among wikipedia pages.

So far, we provide two search functions.

1. Match the keywords with the most related wikipedia page in our dataset, then list the top 20 most related children pages and the top 20 most related parent pages.
2. Take a pair of keywords. Match them with the most related wikipedia page in our dataset respectively. Then return the most related path bewteen the two pages.

MOSTRELATEDPATH($cosineCoefficient$):
  $D = cosineCoefficient$
  $N = D$.SIZE
  $M = \big[M_{i,i} = i\big], i \in \{1, \ldots, N\}]$
  For $k \leftarrow 1, \ldots, N$:
    For $i \leftarrow 1, \ldots, N$:
      For $j \leftarrow 1, \ldots, N$:
        If $D_{i,j} < D_{i,k} \times D_{k,j}$:
          $D_{i,j} \leftarrow D_{i,k} \times D_{k,j}$
          $M_{i,j} \leftarrow k$
  Return D, M

FIGURE 3. Algorithm for all-pair most related path based on Floyd-Warshall

The matching is done by firstly match the keywords with the titiles in the dataset. If no match is found, then the keywords are matched with the content. If nothing is archived, then a default page (very likely the seed) is matched.

For example, in the apple_inc dataset, the keyword of "stanford" will be matched with Stanford, California (the word of "california" may have a smaller score than "university", so Stanford University ranks lower). However, the keyword of "yann lecun" will be matched with deep learning, because Yann LeCun is not indexed in the data base.

5. EVALUATION

It is hard to evaluate a search engine. However, since WikiNet is working on the relation between wikipedia pages, we managed to figure out a reasonable measurement.

For a source page, each page in the returned list several hops away (in the means of all-pair shortest path in a graph).

Consider the naïve way of doing relation search. The most simple yet somehow reasonable way is to do naïve breadth-first search. In this case, in response to a query matched with page q, the naïve strategy will return a list of pages visited in a BFS fashion starting from the source page. In comparison, WikiNet returns the destinations of the 20 most related paths starting from the source.

Denote the documents returned by the naïve strategy $\mathcal{N}(q) = \{d_{q,i}\}$. Use $\mathcal{D}(d_{q,i})$ to mark the depth of document d_i in the BFS tree rooted at q. Denote the list returned by WikiNet as $\mathcal{R}(q) = \langle r_{q,1}, r_{q,2}, \ldots, r_{q,20} \rangle$

If we take the WikiNet result as a reference, the recall of the naïve stratergy can be represented as

$$Recall(q,k) = \frac{1}{\sigma(q,k)} \Big| \Big[\bigcup_{i=1}^{\sigma(q,k)} r_{q,i} \Big] \bigcap \{ d_{q,j} \in \mathcal{N}(q) | \mathcal{D}(d_{q,j}) <= k \} \Big|,$$

where $\sigma(q,k) = \min\{20, \big|\{d_{q,j} \in \mathcal{N}(q) | \mathcal{D}(d_{q,j}) <= k\}\big|\}$, which means the number of pages within k hops from q. The recall equation measures how many of the top $\sigma(q,k)$ pages are at most k hops from q.

We measured every of the 6050 pages in the apple_inc dataset. Since the dataset has only 4 layers, and we only return the top 20 pages, we consider $k \in \{1,2,3\}$ to be reasonalbe. We plot the CDF of $Recall(q,k) : \forall q, \forall k \in \{1,2,3\}$ as Fig 4.

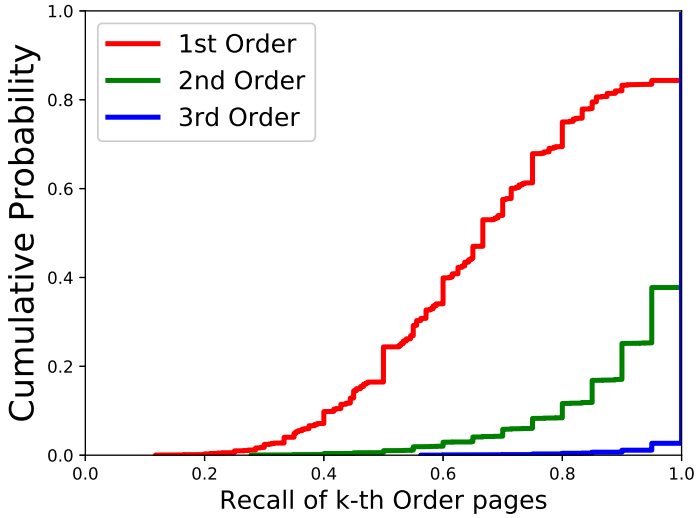


FIGURE 4. CFD for $Recall(q, k)$

As we can see, only in less than 20% of the cases are the top $\sigma(q, 1)$ results from WikiNet is fully overlap with the first order pages. In more than 40% of the cases, the top $\sigma(q, 1)$ results from WikiNet only contains 60% of the first order pages. Even when it comes to second order pages, there are still around 40% of the cases where some of the top $\sigma(q, 2)$ (usually 20) results are at least 3 hops away.

We are confident that WikiNet provide content of high quality and diversity.