Tag Ontology Construction for CQA sites

Introduction

- CQA sites are question answering platforms where users can post a question or answer any question that was posted.
- Questions are usually posted along with some tags
- A tag to a question is something that specifies the topic of the question.
- Tags are helpful in organizing the questions and finding the other related questions.
- Tag helps the platform to reach its target audience who are related to that specific tags.
- Our goal is to organize the tags so as to understand and perceive their relationships by building a Tag Ontology.
- A Tag Ontology is the hierarchical representation of tags and it’s relationships.
- The relations can be of different types like
  - Parent-child
  - Predecessor-successor
  - Sibling

Some statistics of CQA sites

Questions: 20m
Answers: 30m
Tags: 59k

Questions: 1.3m
Answers: 1.8m
Tags: 1.8k

Questions: 357k
Answers: 449k
Tags: 3.1k

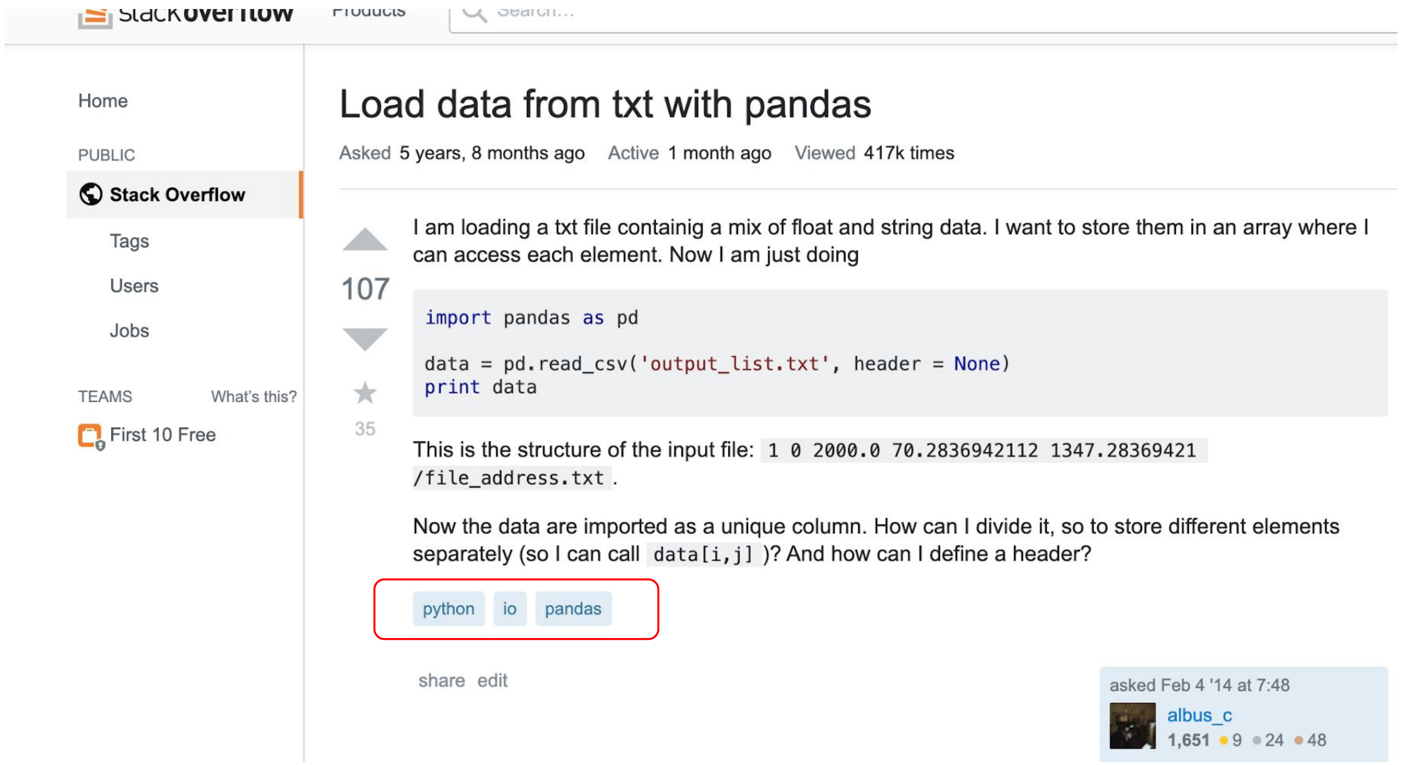
Questions: 202k
Answers: 261k
Tags: 1.8k

Questions: 447k
Answers: 643k
Tags: 5.5k

Questions: 293k
Answers: 473k
Tags: 3.8k

CQA sites in Stack Exchange

Sample post from Stackoverflow



Load data from txt with pandas

Asked 5 years, 8 months ago Active 1 month ago Viewed 417k times

I am loading a txt file containig a mix of float and string data. I want to store them in an array where I can access each element. Now I am just doing

```
import pandas as pd

data = pd.read_csv('output_list.txt', header = None)
print data
```

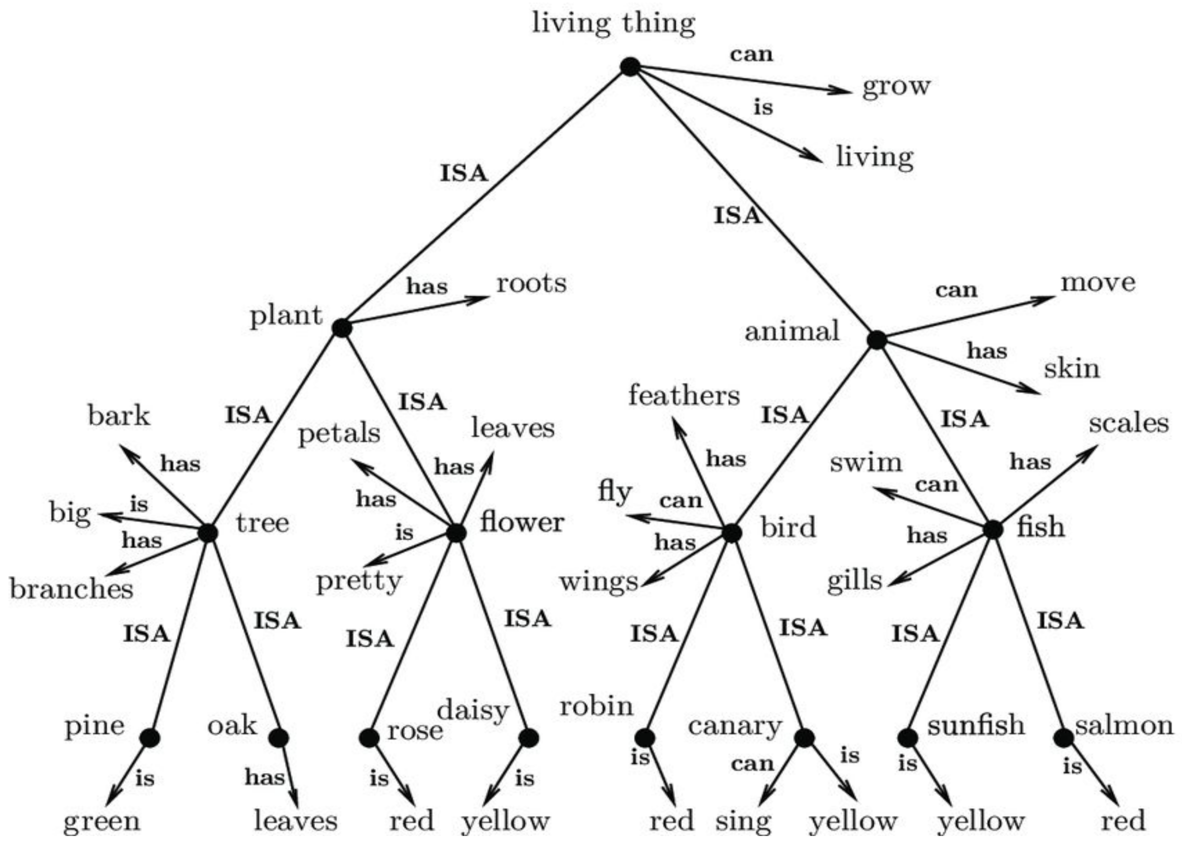
This is the structure of the input file: `1 0 2000.0 70.2836942112 1347.28369421 /file_address.txt`.

Now the data are imported as a unique column. How can I divide it, so to store different elements separately (so I can call `data[i,j]`)? And how can I define a header?

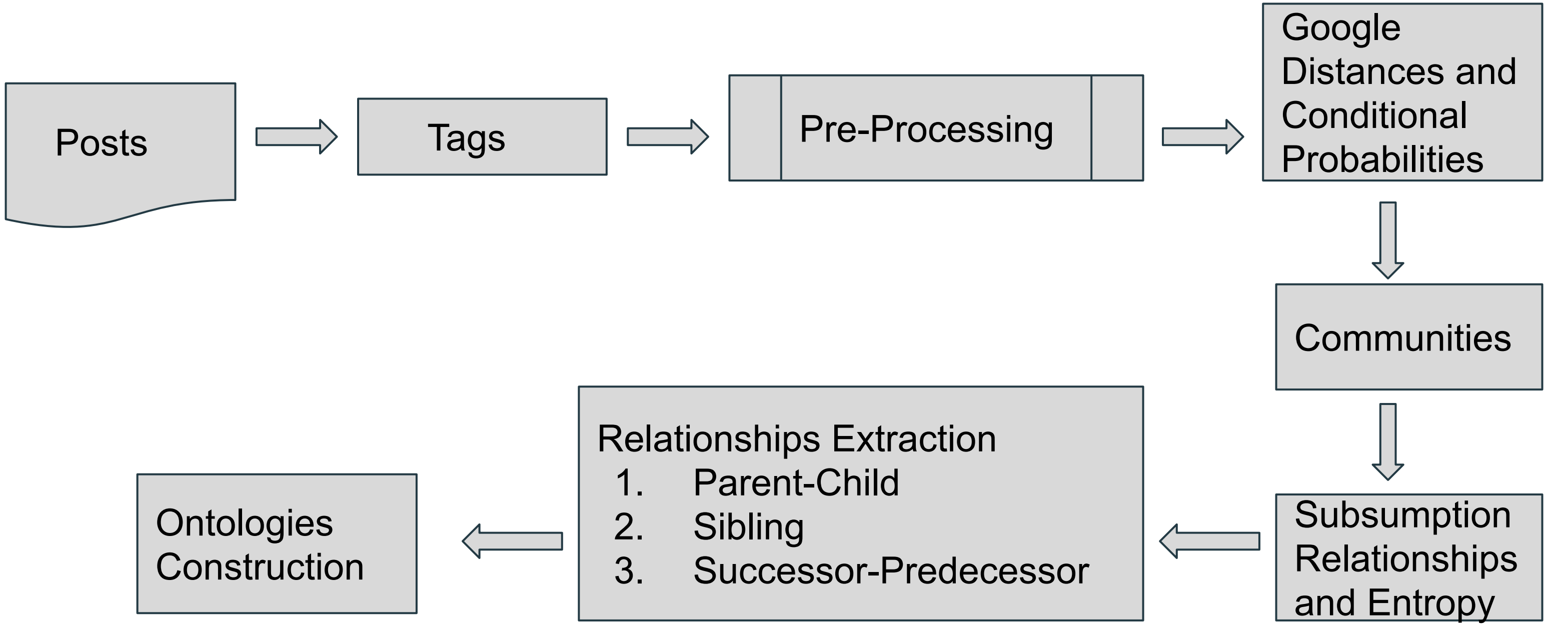
python io pandas

asked Feb 4 '14 at 7:48
albus_c
1,651 ●9 ●24 ●48

Example of Ontology



Process Flow



Relations examples

Parent-of

Office → P

Browsers →

Images → jpeg

Predecessor-of

W 2007 → W 2010

ASP.NET → ASPNET Core 2.0

→

Sibling-of

→

→

→ YAHOO!

Similarity between Tags

- The tags tagged by the user to the questions are usually tagged carefully so we don’t have to do much of the pre-processing.
- We simply remove the least frequent tags to avoid noisy relationships.
- The basic Idea is that if two tags are co-occurring in a post, they are very likely to be related.
- Similarity between two tags is define by using Google Distance.
    - ***s(a,b)*** = ***(1+ exp(d(a,b)))*** $^{-1}$ where *d(a,b)* represent the google distance as shown below
    -
    - ***d(a,b)*** = $$\frac{max(logN(a), logN(b)) - logN(a,b)}{logN_{total} - min(logN(a), logN(b))}$$
    -

Google Distance captures the co-occurrence between the tags based on the number of posts they are appearing together.

Community Detection

- The number of tags in any QA site would be very big to represent them in one Ontology.
- So we wanted to divide the tags into communities based on their similarities.
- Infomap - https://www.mapequation.org/infomap/
- Input - Unweighted Graph G with tags as vertices and weight of the edges as similarity between them(and Conditional Probability defined in next slide)
- Output - Communities

Mining Parent-Child

- The subsumption relation between tag *a* and tag *b* is defined as - If *a* subsumes *b*, then whenever *b* is used, *a* can be used without ambiguity.
- Using Conditional Probability *p(a/b)*

$$p(a|b) = \frac{s(a,b))}{\sum_c s(b,c))}$$

- If a subsumes *b* then *a* will have wider coverage or in a sense it stands above *b* in the tag ontology.
- Entropy H - semantic broadness of each tag

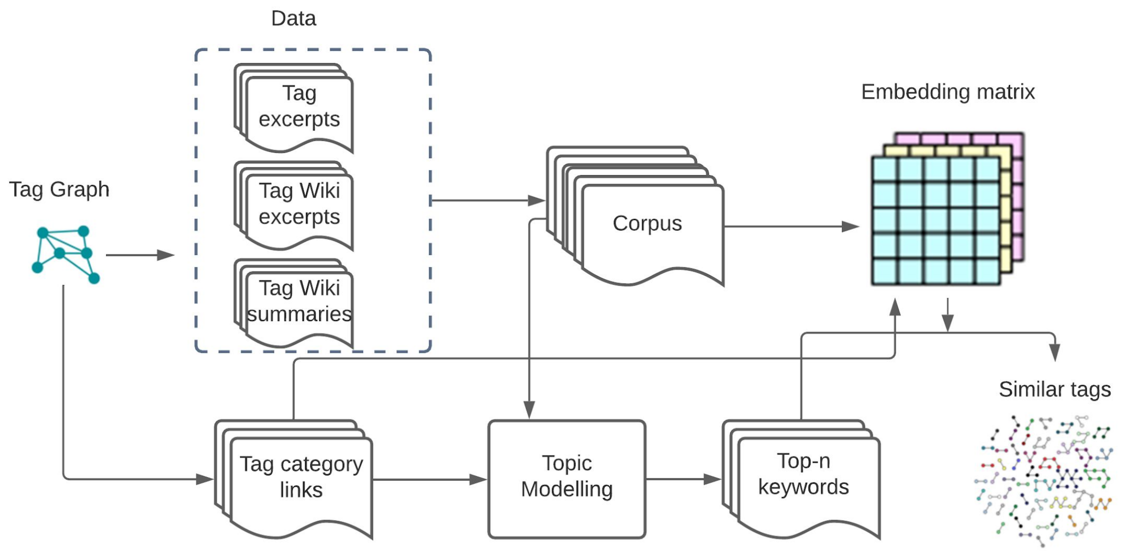
$$H(a) = -\sum_{c} p(a|c)log(p(a|c))$$

- High entropy tag is present above the low entropy tag.
- If *a* subsumes *b* then we can say *a* is the parent of *b*

Mining Sibling relation

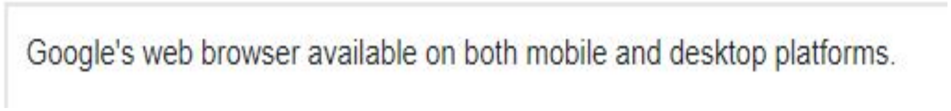
- A tag has some category links in wikipedia which define it
- Similar category links -> Most likely to be siblings
- Get the category links from Wikipedia and find word embeddings of these category links.
- How to get embeddings:
    - Using Labelled LDA(for getting topics of each category link) and Glove(to find embeddings of the topics)
    - Corpus: tag excerpt + wiki summary + excerpt post + wiki post
- After getting embeddings we use Word Movers Distance(WMD)
- ***If(wmd(a,b) < some_threshold) then siblings***
- Why WMD? - Takes semantic similarity into account and two sentences can be of different length

Sibling Architecture



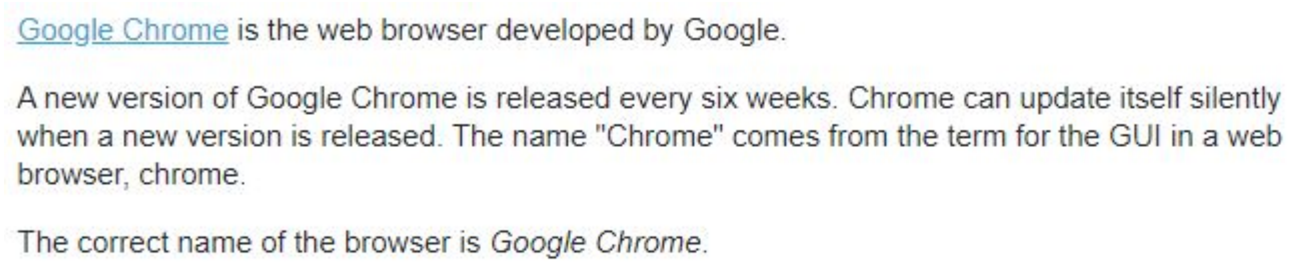
Sample Data of Google Chrome

Tag Excerpt :



Google's web browser available on both mobile and desktop platforms.

Tag wiki summary :

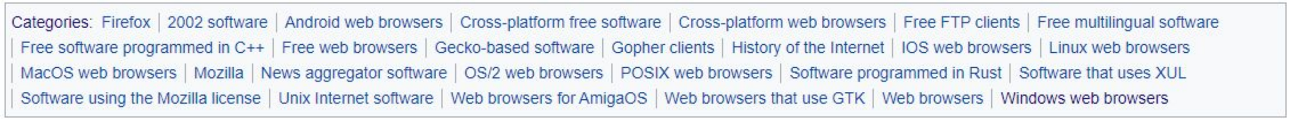


Google Chrome is the web browser developed by Google.

A new version of Google Chrome is released every six weeks. Chrome can update itself silently when a new version is released. The name "Chrome" comes from the term for the GUI in a web browser, chrome.

The correct name of the browser is *Google Chrome*.

Category Links:



Categories: Firefox | 2002 software | Android web browsers | Cross-platform free software | Cross-platform web browsers | Free FTP clients | Free multilingual software | Free software programmed in C++ | Free web browsers | Gecko-based software | Gopher clients | History of the Internet | IOS web browsers | Linux web browsers | MacOS web browsers | Mozilla | News aggregator software | OS/2 web browsers | POSIX web browsers | Software programmed in Rust | Software that uses XUL | Software using the Mozilla license | Unix Internet software | Web browsers for AmigaOS | Web browsers that use GTK | Web browsers | Windows web browsers

Abstract from wikipedia:

Google Chrome is a cross-platform web browser developed by Google. It was first released in 2008 for Microsoft Windows, and was later ported to Linux, macOS, iOS, and Android where it is the default browser built into the OS. The browser is also the main component of Chrome OS, where it serves as the platform for web applications.\nMost of Chrome\'s source code comes from Google\'s free and open-source software project Chromium, but Chrome is licensed as proprietary freeware. WebKit was the original rendering engine, but Google eventually forked it to create the Blink engine; all Chrome variants except iOS now use Blink.As of May 2020, StatCounter and NetMarketShare estimates that Chrome has a 68% worldwide browser market share (after peaking at 72.38% in November 2018) on personal computers (PC), 63.58% and 65.01% respectively across all platforms. Because of this success, Google has expanded the "Chrome" brand name to other products: Chrome OS, Chromecast, Chromebook, Chromebit, Chromebox, and Chromebase.

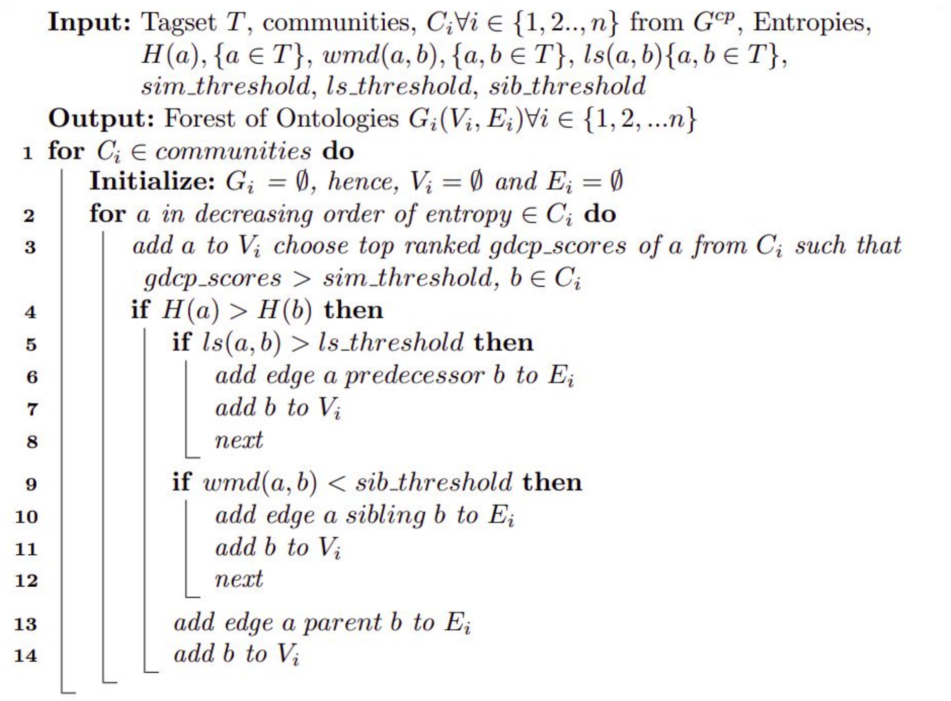
Mining Succ-Pred relationship

- Based on simple lexical criteria

$$\frac{len(Initial\ matching\ string)}{max(len(a), len(b))} > some\_threshold$$

- If the above condition satisfies :
  - *if(len(a) > len(b))* then *a* is successor and *b* is predecessor
  - *Else if(len(a) < len(b))* then *b* is successor and *a* is predecessor
  - *Else if(len(a) == len(b))* then we look for the version number at the end(like *windows-7* and *windows-8*)
  - *Else* tag with highest entropy is successor

Algorithm



```
Input: Tagset $T$, communities, $C_i \forall i \in \{1, 2.., n\}$ from $G^{cp}$, Entropies,
       $H(a), \{a \in T\}, wmd(a,b), \{a, b \in T\}, ls(a,b)\{a, b \in T\}$,
       $sim\_threshold, ls\_threshold, sib\_threshold$
Output: Forest of Ontologies $G_i(V_i, E_i) \forall i \in \{1, 2, ...n\}$
for $C_i \in communities$ do
    Initialize: $G_i = \emptyset$, hence, $V_i = \emptyset$ and $E_i = \emptyset$
    for $a$ in decreasing order of entropy $\in C_i$ do
        add $a$ to $V_i$ choose top ranked $gdcp\_scores$ of $a$ from $C_i$ such that
         $gdcp\_scores > sim\_threshold, b \in C_i$
        if $H(a) > H(b)$ then
            if $ls(a,b) > ls\_threshold$ then
                add edge $a$ predecessor $b$ to $E_i$
                add $b$ to $V_i$
                next
            if $wmd(a,b) < sib\_threshold$ then
                add edge $a$ sibling $b$ to $E_i$
                add $b$ to $V_i$
                next
            add edge $a$ parent $b$ to $E_i$
            add $b$ to $V_i$
```

Experiments and Evaluation

- Dataset
  - Superuser: 5500 tags and over 4 lakh posts
  - Multiple domains like databases, networking, web, system hardware, programming languages,etc.
  - After pre-processing 1350 tags
- Clustering results(Infomap performance)

Data	Modularity	Clustering Coefficient
All tags	0.1	0.01
Popular tags with GD	0.35	0.1
Popular tags with CP	0.8	0.6

Comparison between our features and Baseline Features

- 4 different classifiers: Linear Regression, Naive Bayes, Support Vector Machine and Random Forrest.
- Parent-Child relationships

Classifier	Feature set	Performance			
		Accuracy	Precision	Recall	F1_score
LR	ofs	0.97	0.97	0.98	0.975
	bfs	0.73	0.77	0.76	0.765
SVM	ofs	0.97	0.97	0.98	0.975
	bfs	0.8	0.81	0.81	0.81
NB	ofs	0.96	0.95	0.96	0.955
	bfs	0.60	0.71	0.65	0.679
RF	ofs	0.98	0.97	0.98	0.975
	bfs	0.83	0.82	0.83	0.825

Sibling Comparison

Classifier	Feature set	Performance		
		Precision	Recall	F1_score
LR	ofs	78	68	70
	bfs	67	62	64
SVM	ofs	83	59	61
	bfs	55	43	46
NB	ofs	79	72	74
	bfs	55	56	56
RF	ofs	76	60	62
	bfs	58	53	55

Successor Predecessor

Threshold value	Precision	Recall	F1_score
0.1	9.6	100	17.5
0.3	17.2	97.9	29.25
0.4	30.1	92.7	45.44
0.5	65.5	86.2	74.43
0.6	82.9	83.3	83.09
0.7	89.7	68.7	77.80
0.8	92.6	45.8	61.28
0.9	100	26.2	41.5

Knowledge Bases

- We used 3 standard knowledge bases to compare our results.
  - Concept Net(above 34 million edges)
  - Dbpedia(above 65 million edges)
  - WebIsA(above 3 million edges)

Relation	**Data**	**Performance**		
		Precision	**Recall**	**F1_score**
Parent-child	Our method	91.7	91.11	91.3
	ConceptNet	100	2.4	4.68
	DBpedia	92.3	14.6	25.2
	WebIsAGraph	98.7	8.1	14.9
Sibling	Our method	80.6	76	78.23
	ConceptNet	100	8.1	14.98

Q and A