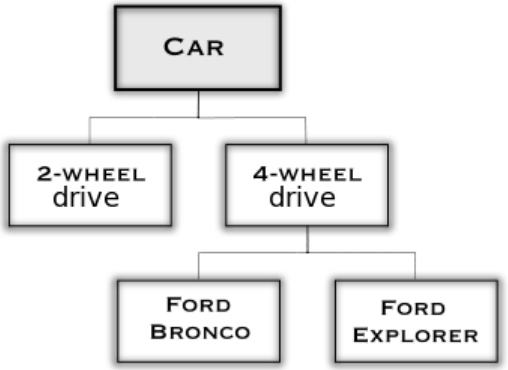
Data Driven Approaches for Large-scale Taxonomy Construction

Jiaqing Liang, Fudan University

Yanghua Xiao, Fudan University

Preliminaries

- is-a
  - *is-a*, (*is-a-subtype-of* or *is-a-subclass-of*).
  - This defines which objects are classified by which class.
    - For example, Ford Explorer *is-a-subclass-of* 4-Wheel Drive Car, which in turn *is-a-subclass-of* Car
- Hypernym & hyponym, concept and entity
  - *apple* isA *fruit,* or hyponym(*apple, fruit)*
  - *fruit* is *apple's* hypernym/concept (superclass)
  - *apple* is *fruit's* hyponym/entity (subclass)
    - Here the `entity` may be a `sub-concept`
- Taxonomy
  - The addition of isa relationships creates a taxonomy: a tree-like structure
  - *We simply call a node in the taxonomy (entity or concept) a term, it is a word or a phrase.*



From：https://en.wikipedia.org/wiki/Ontology_components

Why taxonomy is so important

- Understand an instance
  - *iphone isA smart phone* enables machine to understand the search intent of iphone (i.e. smart phone).
- Entity recommendation
  - *galaxy s4 isA smart phone* further allows to recommend the related keyword galaxy s4
- Many applications
  - machine translation
  - query expansion
  - document classification
  - data cleaning
  - entity resolution
  - information integration

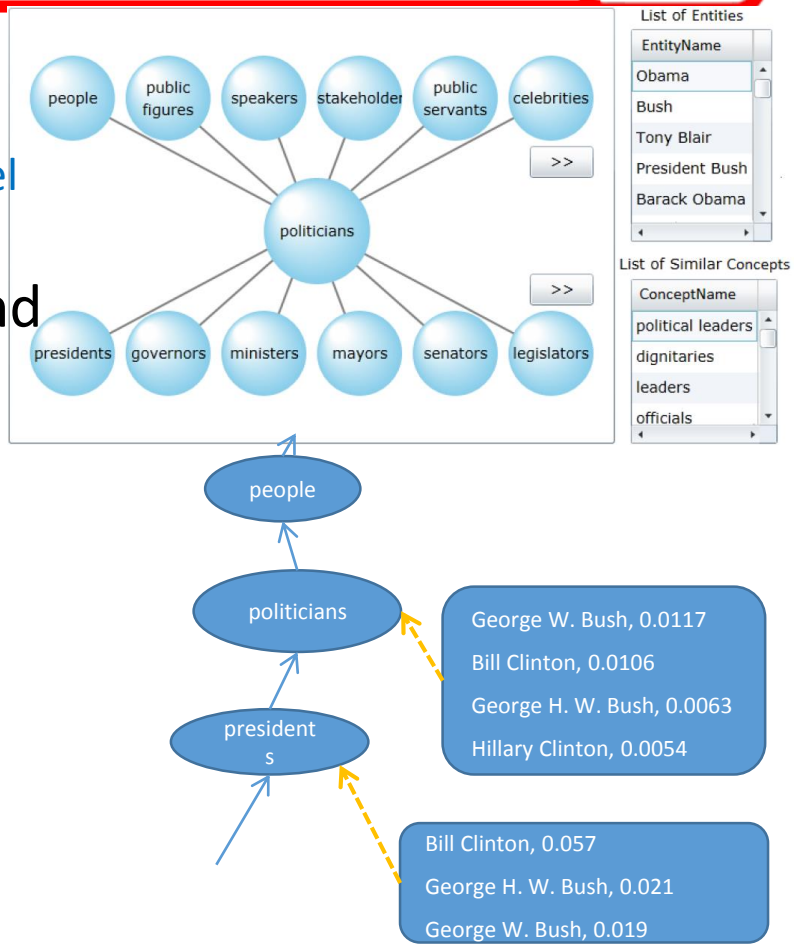
Data Driven vs Hand Crafted

- Manually constructed knowledge graph
  - Examples: WordNet, Cyc
  - Size: **Small** **(Huge human cost)**
  - Quality: Almost perfect (Each relation is checked by expects)

- Auto-constructed knowledge graph
  - Automatically extracted from huge web corpus
  - Examples: Probase, WikiTaxonomy, etc
  - Size: Huge (From huge corpus)
  - Quality: Good (The accuracy can't reach 100%)
    - Because of the huge size, there are many wrong facts

Probase

- A web-scale taxonomy derived from web pages by *Hearst linguistic patterns*
  - "…famous basketball players such as Michael Jordan …"
    - domestic animals such as cats and dogs …
    - China is a developing country.
    - Life is a box of chocolate.
- 10M terms, and 16M isA relations
- Probabilistic knowledge base



https://www.microsoft.com/en-us/research/project/probase/

Pipeline and our works

Pipeline of KG construction

Extraction

- End-to-end
- Domain specific

Cost: Costly Human Efforts

Correction

- Graph structure based correction

Quality：
Wrong data

Completion

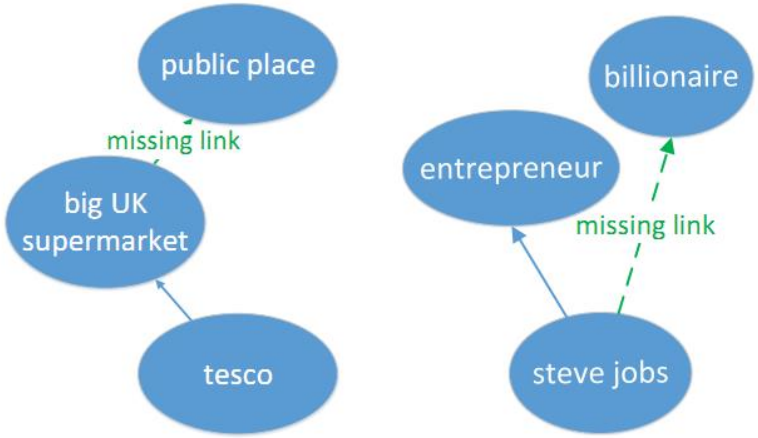
- Collaborative filtering based completion
- Transitivity inference based completion

Quality：
Missing data

Missing isA relationships

- Many valid isA relationships are missing in the taxonomy
  - "big UK supermarket" has no hypernyms in Probase
    - Data sparsity, the relationship between "big UK supermarket" and "public place" rarely appears explicitly
  - "steve jobs" does not connect to the concept "billionaire"
    - Common sense, it is too obvious to be mentioned in texts

- Missing isA relationships break the inference



Errors in automatically constructed lexical taxonomies

- Wrong isA relations in Probase:
  - Errors in corpus
    - "…make Paris such as exciting city…"
    - leads to '**exciting city**' isA '**Paris**'
  - Errors made by information extraction algorithms

- How to detect errors in automatically constructed lexical taxonomies?

Entity	*isA*	**Concept**	**Entity**	*isA*	**Concept**
exciting city	*isA*	paris	battery	*isA*	fuel cell
automobile	*isA*	lead acid battery	cause	*isA*	tsunami
music video	*isA*	youtube video	sweet	*isA*	glucose
world cup	*isA*	football	grape	*isA*	purple
college	*isA*	basketball	juice	*isA*	tomato

Table 1: Examples of incorrect isA relations in Probase

Challenges

Characteristics of data-driven taxonomies and challenges

- **Web-scale**.
  - They usually contain millions of terms and tens of millions of isA relationships.
  - It is a great challenge for the scalability of solutions.
- **Noise**.
  - Some existing isA relationships are wrong, and misleading.
  - In Probase, “germany” isA “latin american country”. We might infer that “france” isA “latin american country” too.
  - How to prevent the inference from the noisy relationships?
- **Ambiguity**.
  - A lexical taxonomy such as Probase does not distinguish the different senses of a term.
  - For example, “apple” has both hypernyms of “company” and “fruit” in Probase. We cannot use “apple” isA “company” to infer “pear” isA “company.”
  - In general, the multiple senses make the inference of truly missing hypernyms more difficult.

Find Missing isA via Transitivity

Transitivity in taxonomies

- One of the most important properties of the isA relationship: transitivity.
- In human-crafted taxonomies, transitivity is taken for granted
  - **Example 1 Is Einstein a scientist?**
- In data-driven lexical taxonomies, transitivity does not always hold
  - **Example 2 Is Einstein a profession?**
  - **Example 3 Is a car seat a piece of furniture?**

In a data-driven lexical taxonomy, when the transitivity holds?

Example 1 *Is* `Einstein` *a* `scientist`*?*
hyponym(`einstein`, `physicist`)
hyponym(`physicist`, `scientist`)
⇒ *hyponym*(`einstein`, `scientist`)

Example 2 *Is* `Einstein` *a* `profession`*?*
hyponym(`einstein`, `scientist`)
hyponym(`scientist`, `profession`)
⇏ *hyponym*(`einstein`, `profession`)

Example 3 *Is a* `car seat` *a piece of* `furniture`*?*
hyponym(`car seat`, `chair`)
hyponym(`chair`, `furniture`)
⇏ *hyponym*(`car seat`, `furniture`)

If we can determine in which cases transitivity hold, we can generate many missing isA relations.

a	b	c
albertsons	supermarket, ...	large store
ipod touch	mp3 player, ipods, ...	consumer electronics
television monitor	display device, ...	device
shampoo	cosmetic, cleaning agent ...	daily good
linkedin	social network, website, ...	web service

Challenges

- It is not a trivial task to tell whether transitivity holds in a data-driven lexical taxonomy

- Naive approach: enforce word sense disambiguation, just as WordNet does
  - Performing word sense disambiguation is costly in a huge lexical taxonomy
  - Dividing the meaning of a word into finite and discrete senses is not always possible
    - chair includes office chair, bench, stool, car seat, etc.

Problem statement and basic idea

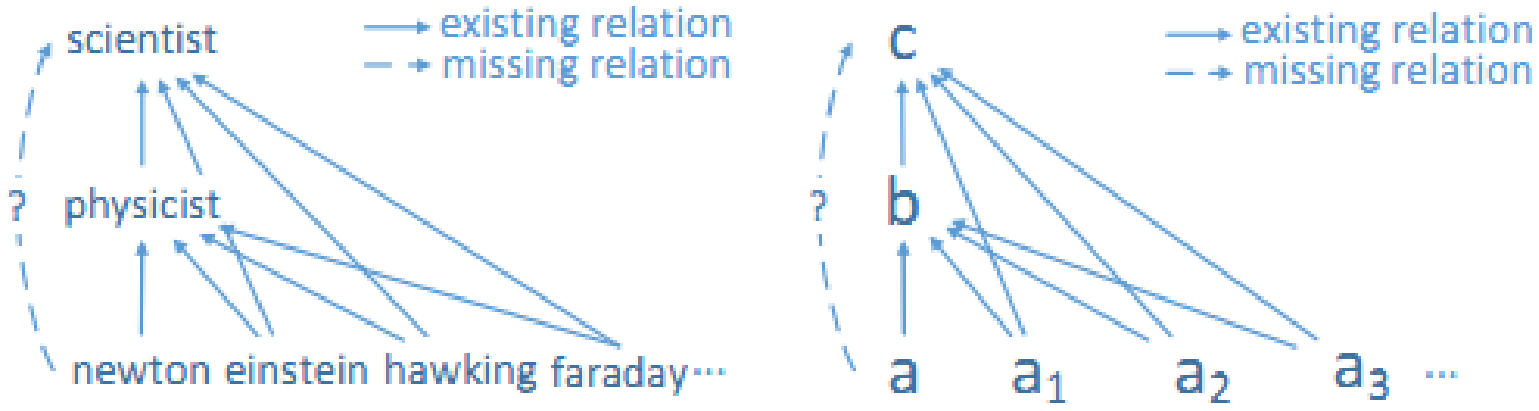
- Problem statement:
  - Input: for a given triple <A,B,C> in Probase satisfying that hyponym(A, B) and hyponym(B, C)
  - Output: judge whether hyponym(A, C) is correct or not
- Our idea
  - A supervised binary classification problem
- Our works:
  - How to build the Labeled dataset?
  - How to design effective Features?

Construction of the labeled dataset

- Objective
  - Collect ground truths about transitivity
- Source
  - WordNet contains hypernym-hyponym relations among synsets.
- Example
  - The word **tank** has two synsets in WordNet.
    - **tank** 1 = storage tank, **tank** 2 = army tank.
  - In WordNet,
    - hyponym(**water tank**, **tank** 1), hyponym(**tank** 1, **vessel**), hyponym(**tank** 2 , **military vehicle**)
  - Then <**water tank, tank, vessel**> is positive, <**water tank, tank, military vehicle**> is negative.
    - hyponym(**water tank, vessel**) holds because the two relations use the same sense of tank.
    - hyponym(**water tank, military vehicle**) is wrong, because the two relations use different senses of tank.
- 9.9k positive triples and 9.4k negative triples.

Features: inferring transitivity from siblings

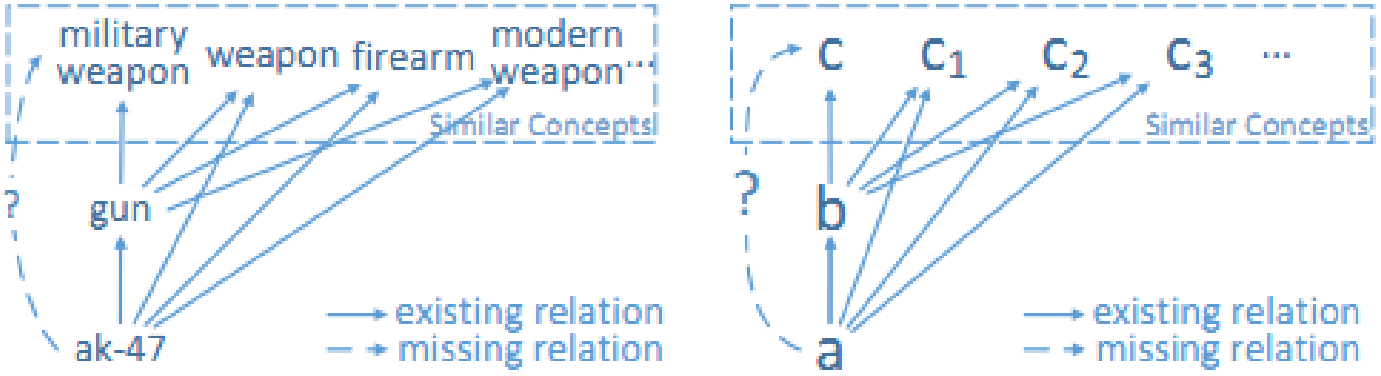
- Principle: similar instances have similar concepts



$$sib_r(t) = \frac{|hypo(\mathtt{b}) \cap hypo(\mathtt{c})|}{|hypo(\mathtt{b})|}, t = \langle \mathtt{a}, \mathtt{b}, \mathtt{c} \rangle$$

Features: inferring transitivity from similar concepts

- Principle: similar concepts have similar instances



$$sim(t) = \frac{\sum_{c_i \in hype(a,b)} sim_c(c, c_i)}{|hype(a,b)|}, t = \langle a, b, c \rangle$$

$$sim_c(c_1, c_2) = 1 - (1 - s_e(c_1, c_2)) \times (1 - s_o(c_1, c_2))$$

$s_e(c1, c2)/s_o(c1, c2)$ is the cosine similarity between c1 and c2's hypernym/hyponym vectors

positive examples in general have a significantly larger sim than negative examples

Pos/Neg	Triple	sim
Positive	⟨ physicist, scientist, profession ⟩	0.545
Positive	⟨ ak-47, gun, dangerous weapon ⟩	0.406
Negative	⟨ newton, scientist, profession ⟩	0.140
Negative	⟨ ak-47, gun, combat skill ⟩	0.035

Table 1: Examples of similarity feature

Features: inferring transitivity from sense number

- Principle: b's ambiguous degree affects the result
- Use WordNet for the sense number
  - b is not in WordNet: b is a rare word that is less likely to be ambiguous and has a unique sense
  - b is in WordNet: We can use the number of synsets of b as the count of b's senses, denoted by synsets(b)

$$s_b(t) = \begin{cases} synsets(\mathrm{b}) & \mathrm{b} \in \text{WordNet;} \\ 1 & \text{otherwise.} \end{cases}, t = \langle \mathrm{a}, \mathrm{b}, \mathrm{c} \rangle$$

  - In addition, we don't consider senses corresponding to entities

$$sc_b(t) = \begin{cases} synsets(\mathrm{b}) - \theta(\mathrm{b}) & \mathrm{b} \in \text{WordNet;} \\ 1 & \text{otherwise.} \end{cases}, t = \langle \mathrm{a}, \mathrm{b}, \mathrm{c} \rangle$$

    - where θ(b) denotes number of senses of b that correspond to entities.

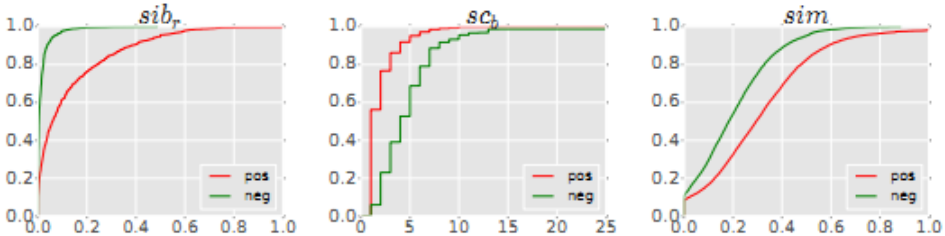
Missing relation inference

- For <A,C> pair that has no relation, we need to determine whether hyponym(A,C) holds or not
- For the <A, C> pair, there are many <A, B_i, C>s s.t. hyponym(A, B_i), hyponym(B_i ,C) hold
  - Classifier of term pairs
    - Train a model directly for <A, C>
    - Use mean pooling to aggregate the feature vectors from different triples.
  - Majority voting
    - For all triples t_i = <A, B_i, C>
    - hyponym(A,C) if and only if most t_i are predicted to be positive
  - Weighted voting
    - Sum up the classification scores over each B_i

Effectiveness of features

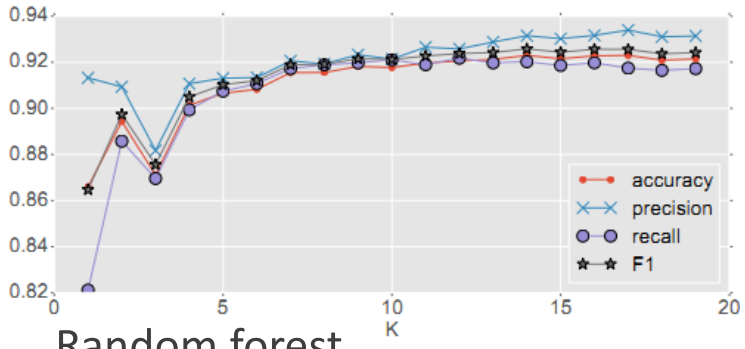
- We use χ^2 and information gain to evaluate the effectiveness of the features used in the classifier.
- We also give CDFs for the top three features ranked by χ^2

#	Feature	χ^2	IG%	#	Feature	χ^2	IG%
1	sib_r	844.50	20.44	11	PMI_{ab}	39.09	4.64
2	sc_b	461.64	23.39	12	v_a	37.07	0.62
3	sim	235.99	4.90	13	$freq_{ab}$	25.61	0.53
4	v_b	158.78	9.40	14	s_a	16.94	0.38
5	$freq_b$	82.09	6.73	15	v_c	4.90	1.32
6	sib	72.02	1.43	16	u_a	0.74	0.07
7	PMI_{bc}	70.20	5.12	17	$freq_c$	0.44	0.48
8	u_b	58.41	8.70	18	$freq_a$	0.08	0.05
9	$freq_{bc}$	53.74	1.32	19	u_c	0.08	1.42
10	sc_c	45.34	1.78				



(a) sib_r (b) sc_b (c) sim

- sib_r (t), sc_b (t), and sim(t) are the top features
- They can clearly separate the positive from negative.
- The top 11 features dominate the performance



Random forest

Results of Probase completion

- The comparison of the three strategies and some examples

Method	Accuracy	Precision	Recall	F1
Binary classifier	88.7%	90.2%	88.2%	89.2%
Majority voting	91.2%	**92.6%**	90.4%	91.5%
Weighted voting	**92.4%**	90.1%	**96.0%**	**93.0%**

- Weighted voting has the best F1
- Weighted voting added 3.86M edges to Probase, with 92% precision (sampling test)

a	b	c
albertsons	supermarket, ...	large store
ipod touch	mp3 player, ipods, ...	consumer electronics
television monitor	display device, ...	device
shampoo	cosmetic, cleaning agent ...	daily good
linkedin	social network, website, ...	web service

Wrong IsA Relation Detection

Errors in automatically constructed lexical taxonomies

- Probase: a lexical taxonomy automatically extracted from web corpora, consisting of tens of millions of isA relations
  - "… famous basketball players such as Michael Jordan … " →
  - "**Michael Jordan**" isA "**famous basketball player**."

- Wrong isA relations in Probase:
  - Errors in corpus
    - "…make Paris such as exciting city…" leads to '**exciting city**' isA '**Paris**'
  - Errors made by information extraction algorithms

Entity	*isA*	Concept	Entity	*isA*	Concept
exciting city	*isA*	paris	battery	*isA*	fuel cell
automobile	*isA*	lead acid battery	cause	*isA*	tsunami
music video	*isA*	youtube video	sweet	*isA*	glucose
world cup	*isA*	football	grape	*isA*	purple
college	*isA*	basketball	juice	*isA*	tomato

Table 1: Examples of incorrect isA relations in Probase

- How to detect errors in automatically constructed lexical taxonomies?

Naïve approaches

- Using frequency
  - Each isA relation in Probase has a frequency observed in corpus
  - Principle: **smaller frequency usually means lower reliability**
  - Problem: many false positives
    - 78% of isA relations with frequency 1 are correct.

- Using external knowledge
  - Idea: Employing external knowledge bases to eliminate the conflicts and improve the quality of the taxonomy
  - Problem: low overlap between different KBs
    - Probase has 2.7 million concepts, Yago has only 0.48 million types and DBpedia has only 700 types

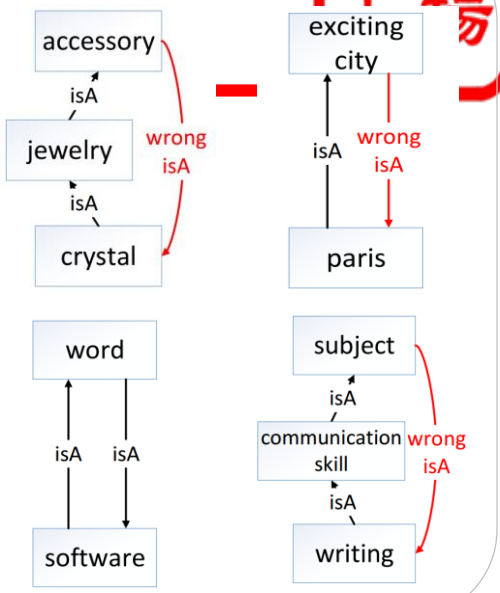
Intuition of our approach

- Observation: There are many errors following this pattern:
  - *An abstract concept* isA *a specific entity*

- The wrong relation and the correct relations tend to form cycles
- An ideally correct taxonomy should be a DAG

Hypothesis: Cycles are important sources of locating ***suspicious*** relations

Examples of cycles

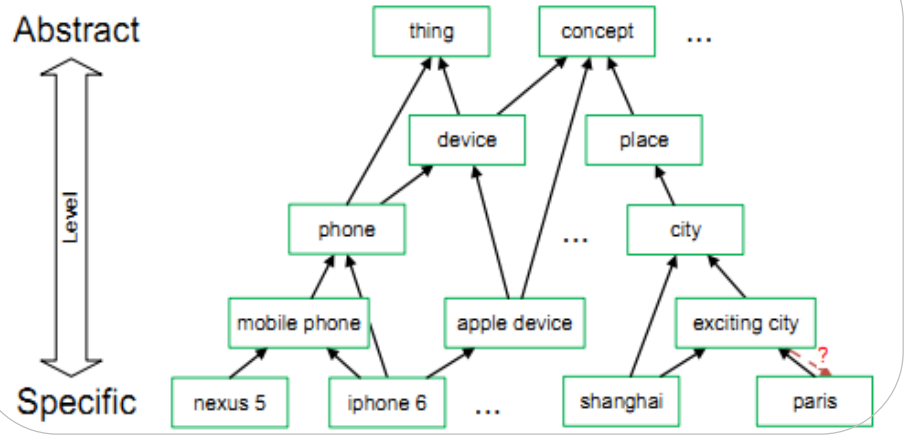


Statistical tests

Size	Have error	Null model	z-score	p-value
2	97%	15%	22.96	<0.0001
3	96%	24%	16.86	<0.0001

More than 95% small cycles contain errors!

An ideal taxonomy



A general model

- Input: a graph G(V, E)
- Output: a wrong edge set E’
- Constraint:
  - G(V, E - E’) is a DAG
  - minimize$\sum_{e \in E'} w(e)$, where w(e) means e’s reliability

- Rationality:
  - The output, wrong isA relation set E’, should contain relations with low reliability
  - Correct edges (edges with high reliability) should be preserved
  - Break cycles with low reliability edges
  - The sum of reliability in E’ should be as low as possible

Reliability metric- Edge frequency

- The edge frequency in Probase (the edge weight in original Probase)
  - Edges with high frequency are more reliable than edges with low frequency
    - China isA country : 10723 times → **reliable**
    - exciting city isA paris: 1 times → **unreliable**
  - Test: Sample and manually judge
    - It is effective

w_f range	Accuracy
1	78%
2-10	86%
11-100	94%
> 100	100%

Table 3: Effectiveness of w_f

- However, 7 million edges' frequency are 1, so that they can't compare to each other

Reliability metric- Difference of #Hyponyms

- Rationality
  - An entity should have no hyponyms
  - A less specific concept should have fewer hyponyms than general concepts

- For an edge X isA Y, if X has many hyponyms but Y has few hyponyms, the edge is unreliable
  - juice (173 hyponyms) isA tomato (69 hyponyms) → **unreliable**
  - exciting city (29 hyponyms) isA paris (9 hyponyms) → **more unreliable**
  - The higher, the more reliable

$$P_h(X\ isA\ Y) = \log\left(1 + \frac{hypo(Y)}{hypo(X)}\right)$$

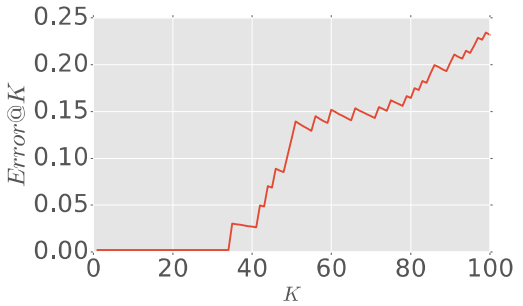


Figure 3: Effectiveness of P_h

Model 1: Minimum feedback arc set

For a G(V, E), find a subset E' of the edge set such that

- G(V, E – E') is a DAG
- $\sum_{e \in E'} w(e)$ is minimized

- This is a classical weighed MFAS problem: NP-Hard

- Approximate greedy algorithm:
  - 1 Randomly choose a cycle
  - 2 Remove the edge in the cycle with minimum weight
  - 3 Back to Step 1, until there is no cycles
  - 4 Try to add back edges removed one by one in the weight descending order, keeping acyclic

- Metrics:
  - #1 $w(e) = freq(e)$
  - #2 $w(e) = freq(e) * P_h(e)$

Model 2: Agony Model

Agony Model: Find a level assignment (l) such that

$$\arg\min_{l} \sum_{(x,y)\in E_R} d(x,y)w(x,y) \qquad d(x,y) = l(x) - l(y) + 1.$$

$$E_R = \{(x,y)|(x,y) \in E, l(x) \geq l(y)\}$$

Penalty function: First, the more errors incurred, the higher the penalty is. Second, the more reliable the edge is, the higher the penalty is.

- Basic idea: A level arrangement of a directed graph implies a DAG. Thus, any backward edges can be identified as wrong edges.
- It is a dual problem of minimum-cost flow problem, solved by a network flow algorithm

Agony+ optimization

- The Agony model removes too many edges
- Basic idea:
  - After we remove some edges, some backward edges will not be in a cycle any more
- Agony+
  - Sort all backward edges by the l(y) - l(x) and weight with ascending order
    - i.e. edges with large level difference has high priority removed
  - Remove each edge one by one
  - If one edge is not in a cycle any more, this edge will be skipped (will not be removed)

Evaluation results on Probase & WikiTaxonomy

- Precision & recall:
  - The Agony+#2 model achieves the highest precision and a relatively higher recall in Probase
- Running time:
  - Our methods can process web-scale taxonomies in acceptable time.
  - The performance of MFAS model is better than that of Agony (Agony+)
- Our method removes near 74 thousand wrong relations with high precision in Probase.

WikiTaxonomy results

Setting	Time	*# result*	Truly wrong	Precision
MFAS	~1sec	108	100	92.6%
Agony	~1sec	112	102	91.1%
Agony+	~1sec	108	101	**93.5%**

Our solutions are effective in both Probase and WikiTaxonomy

Probase results

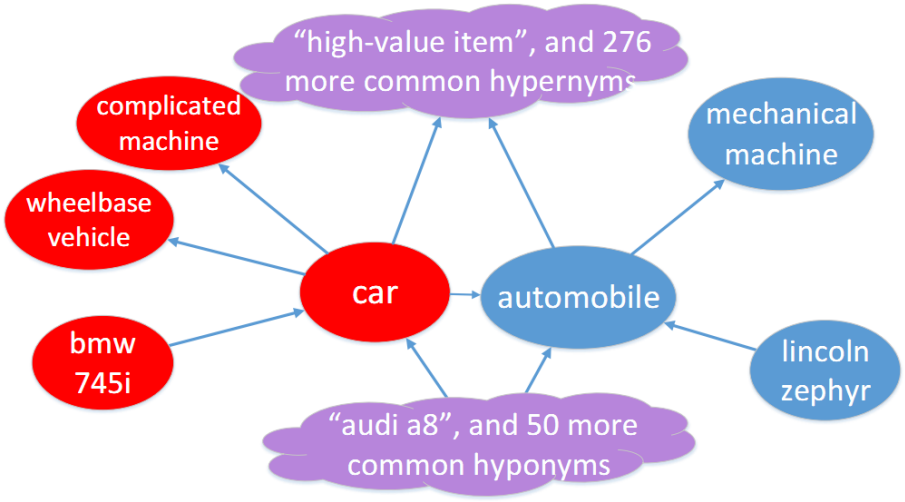
Setting	Time	*# removed*	Precision	*# truly wrong*
Baseline	3min	281.1K	71.0%	199.5K
Baseline+	1.1h	260.7K	72.3%	188.5K (94.5%)
MFAS#1	1.9h	67.1K	86.0%	57.7K (28.9%)
MFAS#2	10.6h	68.7K	90.7%	62.3K (31.2%)
Agony#1	43h	89.5K	83.7%	74.9K (37.5%)
Agony#2	89h	102.3K	84.7%	86.7K (43.4%)
Agony+#1	43h	55.0K	85.7%	47.1K(23.6%)
Agony+#2	89h	74.2K	**91.3%**	67.7K(33.9%)

Inferring Missing Links via Recommend Systems

Our solution: basic idea

- A collaborative filtering (CF) based inferencing mechanism to find missing relationships in Probase
  - **Terms with similar semantics tend to share hypernyms/hyponyms in an isA taxonomy.**
  - "car" and "automobile" are similar in meaning; they share many hypernyms and hyponyms; many links are missing, like "automobile" isA "wheelbase vehicle."

- CF is a natural choice to solve our problem.
- CF is general and flexible enough to allow us to optimize each basic component



Our CF-based framework

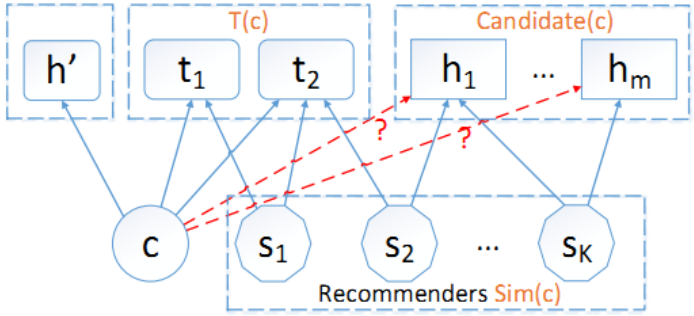
- An iterative framework
- For each term **c**
  - Find top-K terms that are similar to **c**
  - Set each hypernym (**h**) of these top-K terms but **c** as a candidate hypernym
  - Rank candidate hypernyms by aggregating the votes from the top-K similar terms by a scoring function
  - Add the candidates with a score larger than a threshold as the missing hypernyms of **c**

- **T(c)** is the intersection of **c**'s existing hypernyms and **c**'s similar terms' hypernyms
  - Used for the frequency of newly discovered isA relations and the threshold

Algorithm 1 CF-based Missing isA Relationship Finding

Input: Taxonomy T, parameters K, θ

while $iteration < max\_iteration$ **do**
  for term $c \in T$ **do**
    $Sim(c) \leftarrow$ top-K similar terms of c;
    $Candidate(c) \leftarrow [\bigcup_{s \in Sim(c)} hype(s)] - hype(c)$;
    Rank candidates in $Candidate(c)$ by a scoring function f;
    Update T by attaching c to any $x \in Candidate(c)$ s.t. $f(x) \geq \theta$;
  end for
  $iteration \leftarrow iteration + 1$;
end while



The challenges of the framework

- **Similarity metric.**
  - We need an effective semantic similarity metric to find similar terms.
  - Since Probase has ambiguous words or phrases, and noisy or missing isA relationships, it is not easy.
- **Relationship frequency**.
  - In Probase, each isA relationship is associated with a frequency that the relationship is observed from corpora.
  - The frequency is critical for the successful usage.
  - We still need great efforts to estimate an appropriate frequency for the predicted missing relationships.
- **Parameter tuning.**
  - K (used for the selection of the similar concepts of c)
  - Θ (used for the selection of final missing hypernyms).
- **Efficiency**.
  - Probase has millions of terms and tens of millions of relationships.
  - A straightforward solution is not efficient.

Similarity metrics

- Our similarity metric = f(Jaccard metric, Random walk metric)

has high precision, because in a sparse taxonomy, significant overlap of direct neighbors is a strong signal indicating similarity

has high recall, because it explores a much larger neighborhoods.

Similarity metrics: Jaccard similarity

- We use Jaccard similarity as a direct measure of similarity of two terms.
- Two terms that are similar to each other often have many common hypernyms/hyponyms
- We compute the Jaccard similarity of two terms' hypernym set and hyponym set.
- We further use a noisy-or model to combine the two scores

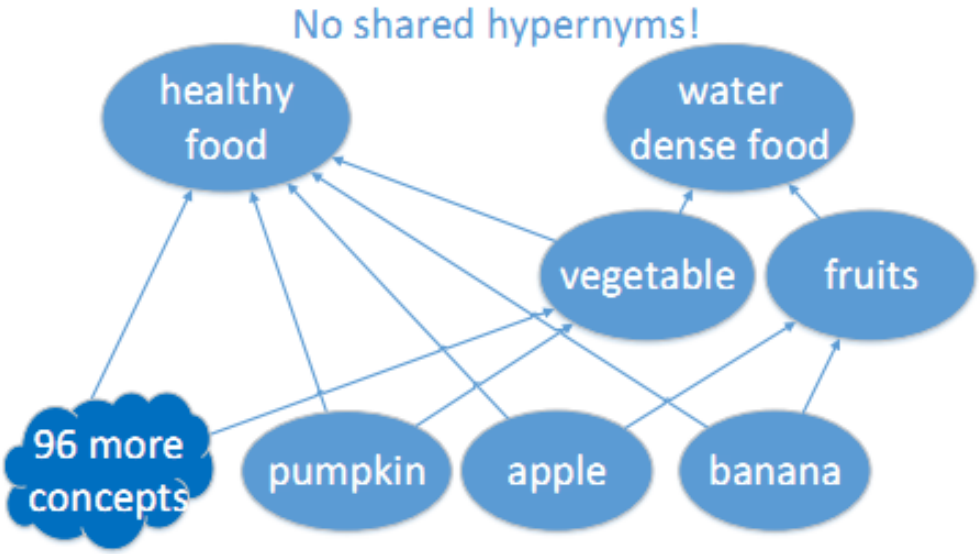
- Rationale of noisy-or:
  - Each individual similarity signal tends to be weak due to the fact that there are many missing links
  - J_o maybe too small

$$Jacc(U,V) = \frac{|U \cap V|}{|U \cup V|} = \frac{|U \cap V|}{|U| + |V| - |U \cap V|}$$

$$j_e(c_1, c_2) = Jacc(hype(c_1), hype(c_2))$$
$$j_o(c_1, c_2) = Jacc(hypo(c_1), hypo(c_2))$$
$$jacc(c_1, c_2) = 1 - (1 - j_e(c_1, c_2)) \cdot (1 - j_o(c_1, c_2))$$

Similarity metrics: random walk similarity

- Jaccard similarity only looks at a term's immediate neighbors (its hypernyms and hyponyms)
  - "water dense food" and "healthy food" have only few common neighbors in Probase and their Jaccard similarity is 0.004
  - But they are similar and they actually have many indirect neighbors



Similarity metrics: random walk similarity

- Use random walk to get the feature vectors of each term
  - Let N be the number of terms in the taxonomy.
  - For each term c, we construct a vector V(c) of 2N dimensions
  - It is the result of concatenating the random walk vectors starting at c along two different edge directions
- Random walk:
  - Each item of $v_c^{(i)}$ is the probability that a certain term is reached starting from c after i steps of random walk
  - We use $v_c^{(L)}$ for the final feature vector
    - i.e. only walks L steps
    - We find that tiny updates happen after L = 2, thus we use L = 2

$$\vec{v}_c^{(i)} = \frac{1}{2}\vec{v}_c^{(0)} + \frac{1}{2}M\vec{v}_c^{(i-1)}$$

- And finally, we use cosine of V(c1) and V(c2) as the RW-similarity of c1 and c2

$$rw(c_1, c_2) = \frac{V_{c_1} \cdot V_{c_2}}{\|V_{c_1}\|\|V_{c_2}\|}.$$

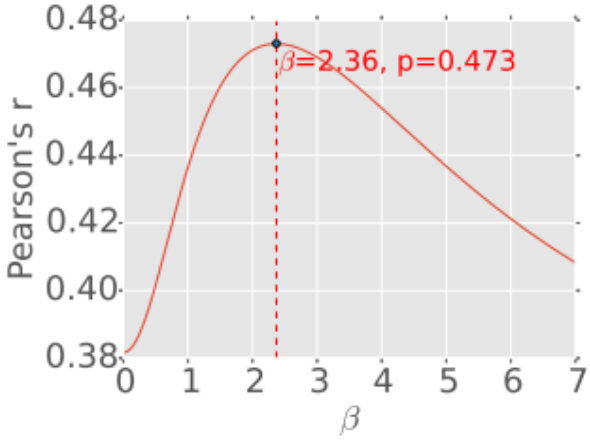
Similarity metrics: combine two metrics

- F_β:
- We set $\beta = 2.36$

$$cs(c_1, c_2) = F_\beta(jacc(c_1, c_2), rw(c_1, c_2)) = \frac{(1+\beta^2) \cdot jacc(c_1, c_2) \cdot rw(c_1, c_2)}{\beta^2 \cdot jacc(c_1, c_2) + rw(c_1, c_2)}$$

- We use WordSim353-similarity to find this best β
  - For each term-pair in this dataset, we compute our combined metrics with different setting
  - Then we compute the Pearson's r between our scores and human scores.

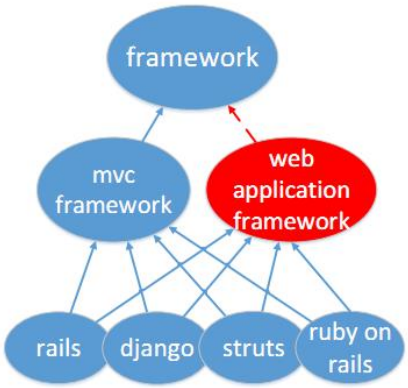
- $F_{2.36}$ is better than other combinations

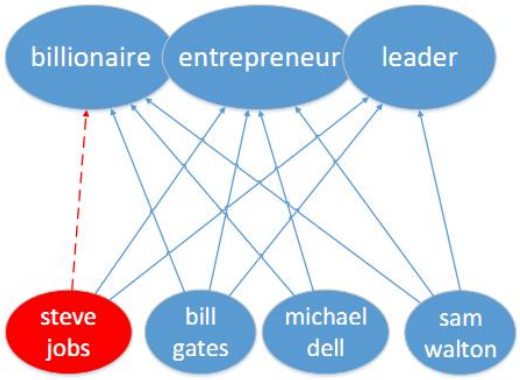


Metric	Pearson's r
Jaccard	0.382
RW Vector	0.339
Geometric Mean	0.448
Arithmetic Mean	0.389
Harmonic Mean(F_1)	0.436
$F_{2.36}$	**0.473**

Similarity: case studies

- Our metric is able to find synonyms
  - “car” and “automobile” ‘s similarity is 0.33
- Our metric is able to find similar concepts
  - “web application framework” and “mvc framework” ‘s similarity is 0.16
- Our metric is able to find similar entities
  - “steve jobs” and “bill gates” ‘s similarity is 0.37
- Here are more cases that finding top-5 similar terms.



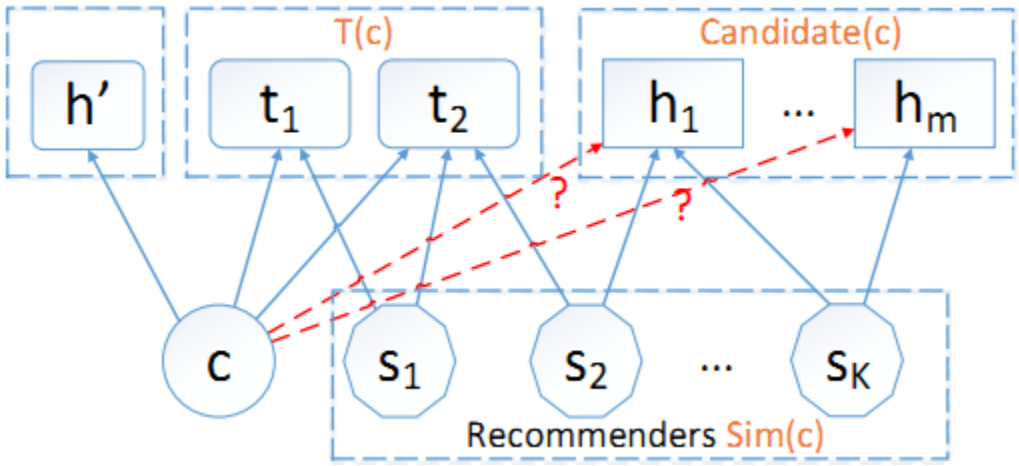


Term	Similar terms
facebook	twitter, linkedin, myspace, flickr, digg
python	perl, ruby, visual basic, tcl, basic
haskell	erlang, standard ml, scheme, ocaml, lisp
iphone	ipod touch,apple iphone, smartphones, psp, smart phone
microsoft windows	mac os,windows xp,windows, windows 95,mac os x
warcraft	starcraft,warcraft iii,company of heroes, age of empires, half life

Candidate ranking

- Let Sim(c) as the top-k similar terms of term c, and Candidate(c) is the hypernyms of terms in Sim(c) that are not a hypernym of c now.
- We have to give a ranking score for each candidate h in Candidate(c)

- Deterministic
- Probabilistic
- Regression



Candidate ranking

- Score 1: A Deterministic Approach
  - *Just sums over all vote weights from recommenders (Sim(c))*
  - Here cs is the similarity score, δ(s, h) = 1 if s isA h else 0

$$r(h) = \frac{\sum_{s \in Sim(c)} \delta(s,h) \cdot cs(c,s)}{\sum_{s \in Sim(c)} cs(c,s)}$$

- Score 2: A Probabilistic Approach
  - *A candidate h is likely to be a c's hypernym if h likely to be hypernyms of terms in Sim(c).*
  - Here n(s isA h) is the frequency of s isA h in Probase
  - P(s) is the prior probability of s, which is calculated as the ratio of n(s) to the sum of frequency over all terms (for normalization)

$$P(h|s) = \frac{n(s \text{ isA } h)}{\sum_x n(s \text{ isA } x)}$$

$$r(h) = \frac{\sum_{s \in Sim(c)} P(h|s) \cdot P(s) \cdot cs(c,s)}{\sum_{s \in Sim(c)} P(s) \cdot cs(c,s)} = \frac{\sum_{s \in Sim(c)} P(h,s) \cdot cs(c,s)}{\sum_{s \in Sim(c)} P(s) \cdot cs(c,s)}$$

Candidate ranking

知識工場

- Score 3: A Regression Model
  - The above models cannot assign a frequency to each newly found edge
  - We use regression model for a frequency-like ranking score
  - Let T(c) be the hypernyms of Sim(c) that has a link to c,

$$T(c) = [\bigcup_{s \in Sim(c)} hype(s)] \cap hype(c)$$

  - We compute a K-dim feature vector for each h
  - Then we can do regression, using for training

$$\mathbf{x}(h_i) = [g(n(c_k \text{ isA } h_i))]_{c_k \in Sim(c)}$$

$$\{(\mathbf{x}(h_i), \mathbf{y}(h_i))\}_{h_i \in T(c)}$$

  - And do predicting for Candidate(c)

$$\mathbf{y}(h_i) = g(n(c \text{ isA } h_i))$$

$$g(x) = \log(1 + x)$$

Scalability

- The complexity of our solution is dominated by the computation of top-K similar terms for every term in Probase
  - A straight-forward solution costs at least $O(N^2l)$ time
  - N is the number of terms in Probase and O(l) is the cost of computing the similarity of two terms

Method	Time Complexity	Description
Naive Method	$O(N^2 l)$	$N \propto 10^7$; l ranges from 10^3 to 10^5
Concept Filtering	$O(n^2 l)$	$n \propto 10^5$
Concept Pair Filtering	$O(nml)$	$m \propto 10^4$
Upper bound Pruning	$O(nm \log m + nmd + nkl)$	$O(nl)$ time to generate summary; practically $d \propto 10^2$ and $k < 0.5m$

Experiment

- Overall: comparisons of taxonomies
  - Our improved version: Probase+
  - Probase+ is a more comprehensive conceptual taxonomy.
  - It is the largest taxonomy as far as we know.
  - Overall, Probase+ has nearly 91% accuracy

Taxonomy	#Concepts	#isA Relations	Accuracy
WordNet	82,115	84,428	100.0%
WikiTaxonomy	76,808	105,418	86.5%
Probase	10,378,743	16,285,393	92.8%
Probase+	**10,378,743**	**21,332,357**	**90.9%**

Experiment

- Exp 1: Performance of missing isA relationship detection
  - Our solution can find missing links in a web-scale taxonomy in acceptable time.
  - The upper bound pruning accelerates our method

Iteration	Time(without pruning)	Time(with pruning)
1	42 hours	18 hours
2	$>$ 48 hours	22 hours
3	$>$ 48 hours	25 hours
total	$>$ 136 hours	65 hours

Experiment

- Exp 4: Precision and recall
  - Precision: We ask volunteers to judge randomly-sampled 2000 detected missing links in each iteration
  - Recall: We remove some correct edges already in Probase, and try to recover them
  - The precision is consistently about 85%.
  - We can recover about 80% removed edges.

Precision of missing isA detection. Precision is defined as $\frac{\#Correct}{\#Sampled}$

Recall of missing isA detection. Recall is defined as $\frac{\#Recovered}{\#Removed}$

Iter.	Samp.	Corr.	Prec.	Removed	Recovered	Recall
1	2000	1746	87.3%	200	162	81.0%
2	2000	1689	84.5%	400	309	77.3%
3	2000	1672	83.6%	600	489	81.5%
All	6000	5107	85.1%	800	664	83.0%

Experiment

- Exp 5: Case studies
  - Here are some found missing links
  - Many missing concepts are obviously correct but rarely mentioned in corpus
    - steve jobs" is obviously a "billionaire"
  - Most newly found hypernyms are specific concepts like "weight loss food," "dark-colored plant food."

Examples of missing isA

Entity	Concept
steve jobs	billionaire
battery	reusable product
einstein	scientific pioneer
wireshark	software program
ipad	wireless device
taobao	website
ps3	electronic product
sudoku	mental game
world war ii	disaster
mcdonalds	fast-food chain

Case studies

Entity	Concept
black tea	dark-colored plant food
black tea	weight loss food
black tea	famous herb
black tea	asian flavor
black tea	longevity power food
doom 3	video game
doom 3	first-person shooter
doom 3	violent game
doom 3	online video game
doom 3	online game

References

- Wu W, Li H, Wang H, et al. Probase: A probabilistic taxonomy for text understanding[C]//Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. ACM, 2012: 481-492.
- Hearst M A. Automatic acquisition of hyponyms from large text corpora[C]//Proceedings of the 14th conference on Computational linguistics-Volume 2. Association for Computational Linguistics, 1992: 539-545.
- Liang J, Xiao Y, Zhang Y, et al. Graph-Based Wrong IsA Relation Detection in a Large-Scale Lexical Taxonomy[C]//AAAI. 2017: 1178-1184.
- Liang J, Zhang Y, Xiao Y, et al. On the Transitivity of Hypernym-Hyponym Relations in Data-Driven Lexical Taxonomies[C]//AAAI. 2017: 1185-1191.
- Liang J, Xiao Y, Wang H, et al. Probase+: Inferring Missing Links in Conceptual Taxonomies[J]. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(6): 1281-1295.

Thank YOU！

Our LAB: Knowledge Works at Fudan

http://kw.fudan.edu.cn