Data Driven Approaches for Large-scale Knowledge Graph Construction

Yanghua Xiao

Knowledge Works at Fudan (kw.fudan.edu.cn)

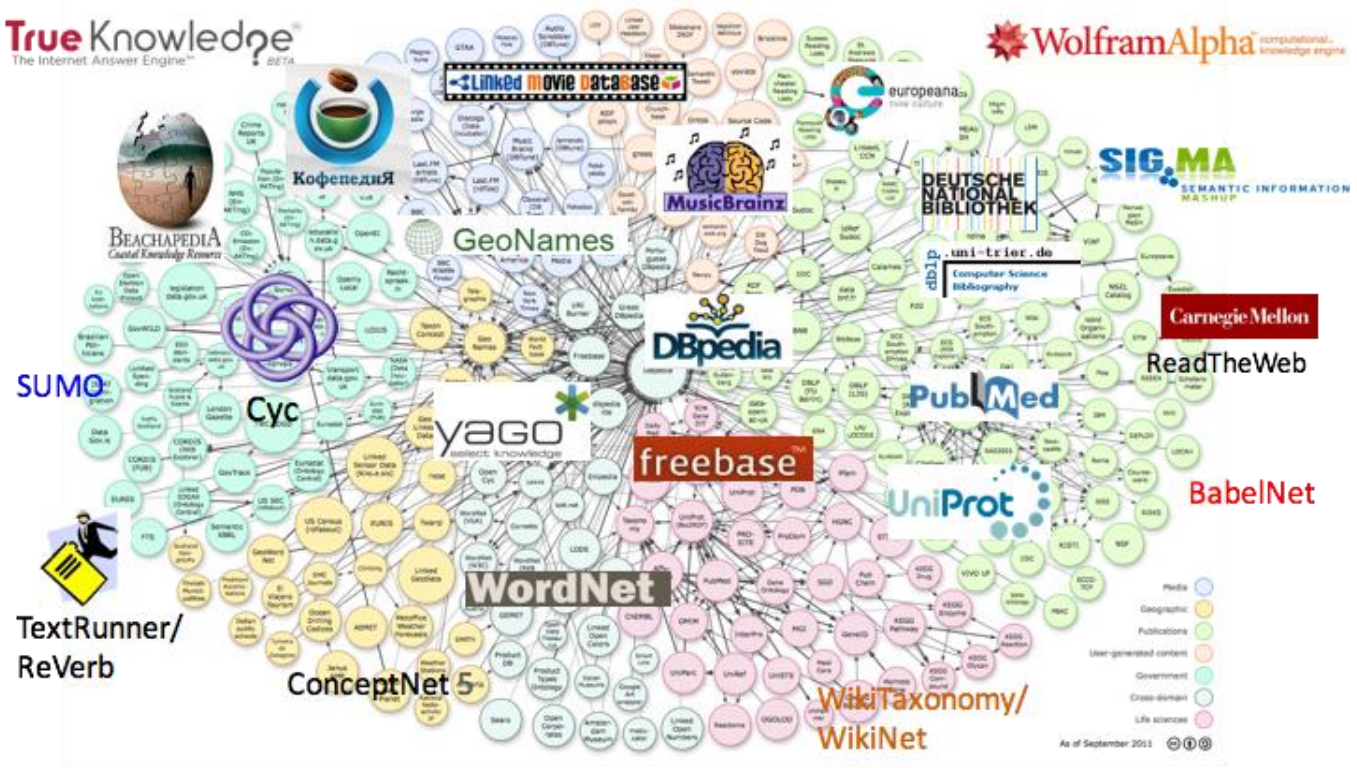
Fudan University

Knowledge Graph

- Knowledge graph is a large scale semantic network consisting of entities/concepts as well as the semantic relationships among them
  - Higher coverage over entities and concept
  - Richer semantic relationships
  - Usually organized as RDF
  - Quality insurance by Crowdsourcing

- Why Knowledge Graphs?
  - Understanding the semantic of text needs background knowledge
  - A robot brain needs knowledge base to understand the world

- Yago , WordNet, FreeBase, Probase, NELL, CYC, DBPedia….

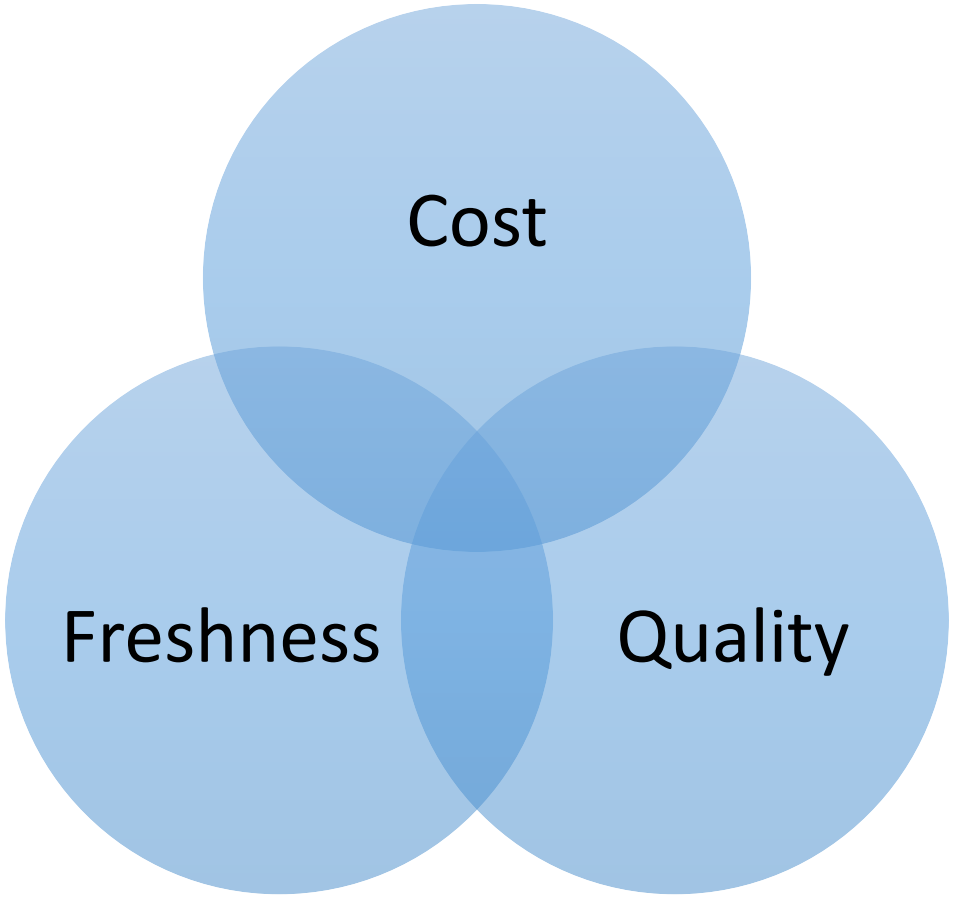


Data Driven vs Hand Crafted

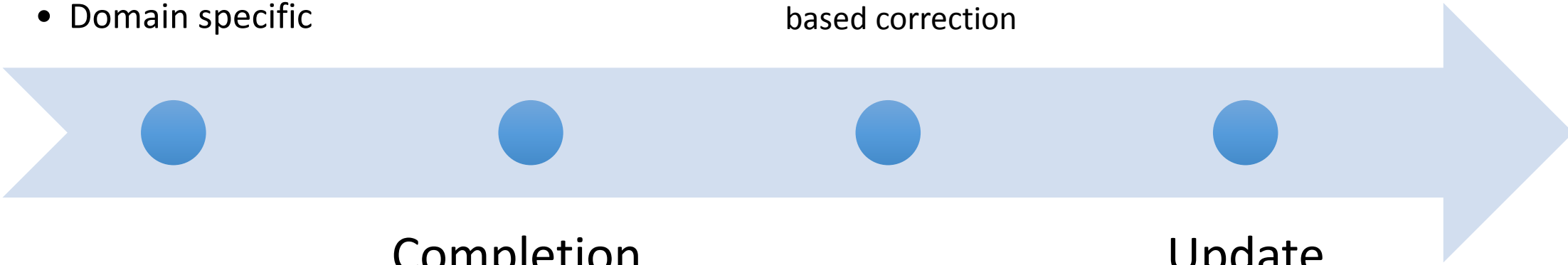
- Manually constructed knowledge graph
  - Examples: WordNet, Cyc
  - Size: **Small** **(Huge human cost)**
  - Quality: Almost perfect (Each relation is checked by expects)

- 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

Key issues in KG construction



Pipeline of KG construction



Extraction

- End-to-end
- Domain specific

Cost:
Reduce Human Efforts

Completion

- Collaborative filtering based completion
- Transitivity inference based completion

Quality:
Find missing facts

Correction

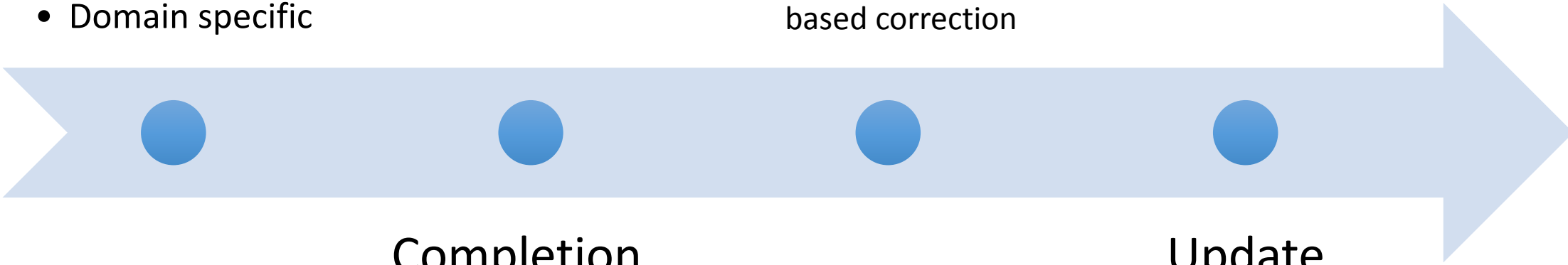
- Graph structure based correction

Quality:
Correct wrong facts

Update

Freshness:
Reduce latency

Pipeline of KG construction



Extraction

- End-to-end
- Domain specific

Cost:
Reduce Human Efforts

Correction

- Graph structure based correction

Quality:
Correct wrong facts

Completion

- *Collaborative filtering based completion*
- Transitivity inference based completion

Quality:
Find missing facts

Jiaqing Liang, Yanghua Xiao, et a, Probase+: Inferring Missing Links in Conceptual Taxonomies, TKDE 2017

Update

Freshness:
Reduce latency

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 concepts, and 16M isA relations

Hearst pattern

NP such as NP, NP, …, and|or NP such NP as NP,* or|and NP

NP, NP*, or other NP

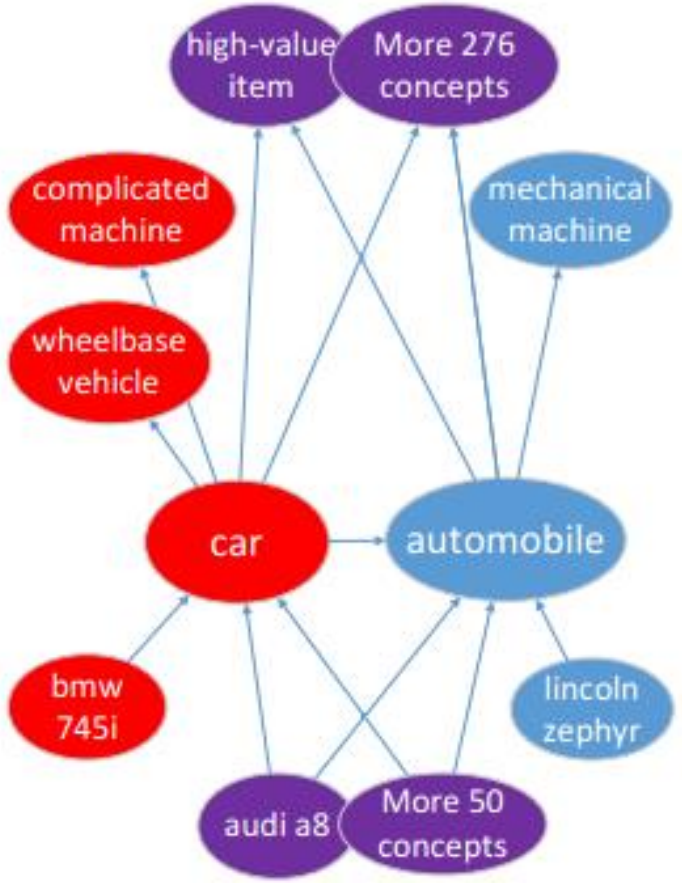
NP, NP*, and other NP

NP, including NP,* or | and NP NP, especially NP,* or|and NP

Missing isA relationships in Probase

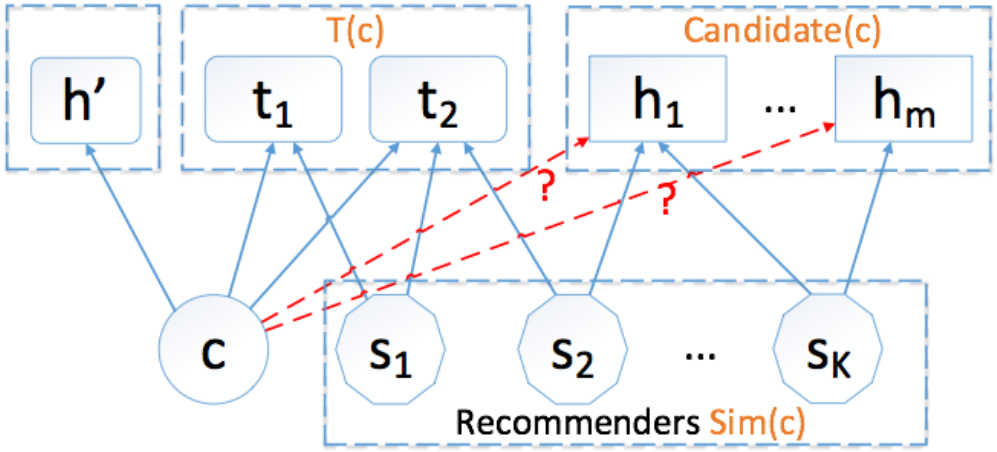
- “car” and “automobile” are synonyms
  - They should share hypernyms
  - “automobile” should beA “wheelbase vehicle”

- Missing isA relaiton hurts the understanding the concepts of entities
  - Is Lincoln zephyr a car?



Solution idea: CF based Missing isA inference

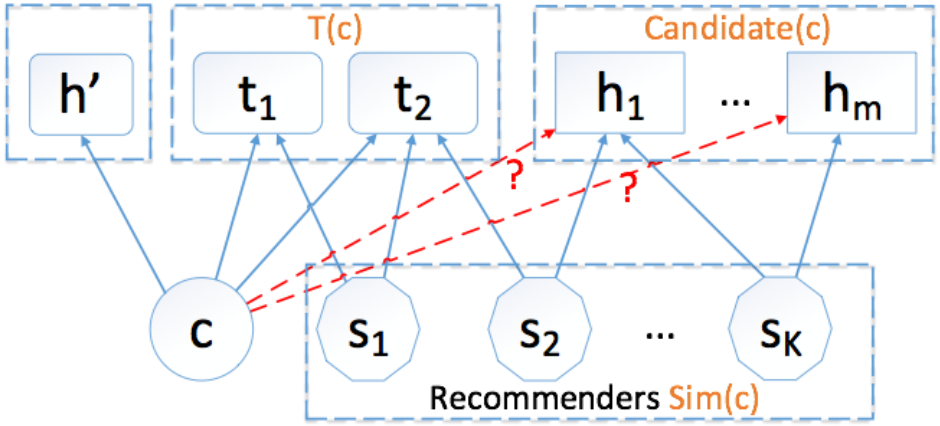
- User-based collaborative filtering!
  - Hypernyms --- Items
  - Concepts --- Users
  - Synonyms or Siblings --- Similar users
- Concepts with similar meanings tend to share hypernyms/hyponyms in an isA taxonomy
- To find missing hypernyms for a concept c
  - First find c's synonyms and siblings
  - Then we transport their hypernyms to c



Idea:
if most similar terms of c have h as the hypernym, c is likely to have the hypernym h.

Problems to be solved

- Effectiveness
  - Sparsity: How to deign an effective similarity metric?
    - Noisy-or model amplifying the weak signals
  - Weight aware: How to estimate a frequency for the new isA relation?
    - Build a regression model
  - Diversity: How to select the final hypernyms?
    - Dynamically tuning k for the top-k selection
- Efficiency
  - How to reduce the quadratic complexity of pairwise similarity computation?
    - Upper-bound pruning



Results

- Recover 5.1M missing edges，with precision 87%, recall 80%.
- Probase plus has accuracy 91%

TABLE 7
Precision of missing isA detection. Precision is defined as $\frac{\#Correct}{\#Sampled}$

Iter.	Samp.	Corr.	Prec.
1	2000	1746	87.3%
2	2000	1689	84.5%
3	2000	1672	83.6%
All	6000	5107	85.1%

TABLE 8
Recall of missing isA detection. Recall is defined as $\frac{\#Recovered}{\#Removed}$

Removed	Recovered	Recall
200	162	81.0%
400	309	77.3%
600	489	81.5%
800	664	83.0%

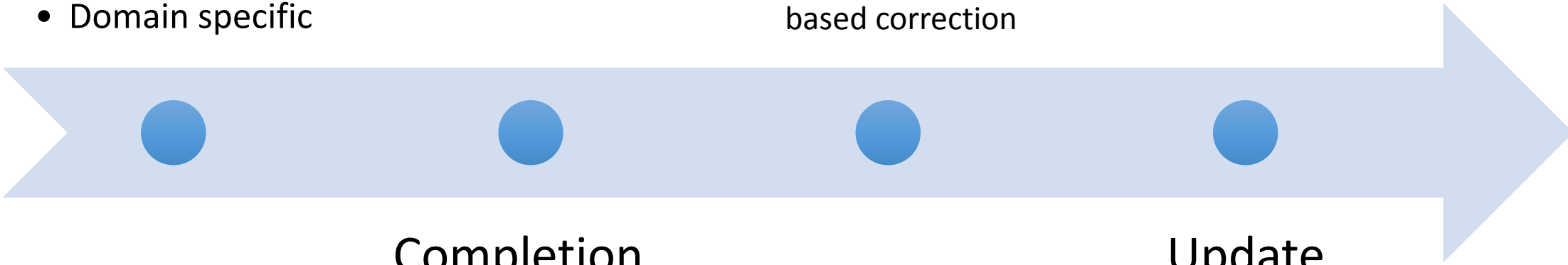
Precision and recall

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%**

Entity	Concept
steve jobs	billionaire
einstein	scientific pioneer
ipad	wireless device
ps3	electronic product
world war ii	disaster
doom 3	video game
taobao	website
wireshark	software program
education	personal information
temperature	anthropometric measurement
black tea	liquid
windows xp	pc operating system

Case study

Pipeline of KG construction



Extraction

- End-to-end
- Domain specific

Cost:
Reduce Human Efforts

Completion

- Collaborative filtering based completion
- *Transitivity inference based completion*

Quality:
Find missing facts

Correction

- Graph structure based correction

Quality:
Correct wrong facts

Update

Freshness:
Reduce latency

Jiaqing Liang, Yi Zhang, **Yanghua Xiao***, Haixun Wang, Wei Wang and Pinpin Zhu, On the Transitivity of Hypernym-hyponym Relations in Data-Driven Lexical Taxonomies, (***AAAI 2017***)

Motivation

- We can use transitivity to find many missing isA relations
  - Example 1
- But it is not trivial, there are wrong cases
  - Example 2 & 3
- If we can determine in which cases transitivity hold, we can generate many missing isA relations
  - There are some examples, **a isA c** are found missing isA relations

Example 1 *Is* `Einstein` *a* `scientist`*?*
hyponym(`einstein`, `physicist`)
hyponym(`physicist`, `scientist`)
$\Rightarrow$ *hyponym*(`einstein`, `scientist`)

Example 2 *Is* `Einstein` *a* `profession`*?*
hyponym(`einstein`, `scientist`)
hyponym(`scientist`, `profession`)
$\not\Rightarrow$ *hyponym*(`einstein`, `profession`)

Example 3 *Is a* `car seat` *a piece of* `furniture`*?*
hyponym(`car seat`, `chair`)
hyponym(`chair`, `furniture`)
$\not\Rightarrow$ *hyponym*(`car seat`, `furniture`)

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

Problem Statement and Basic Idea

- Objective:
  - Input: for a given triple <A,B,C> in Probase satisfying that A isa B, B isa C
  - Output: Judge whether A isa C is correct or not

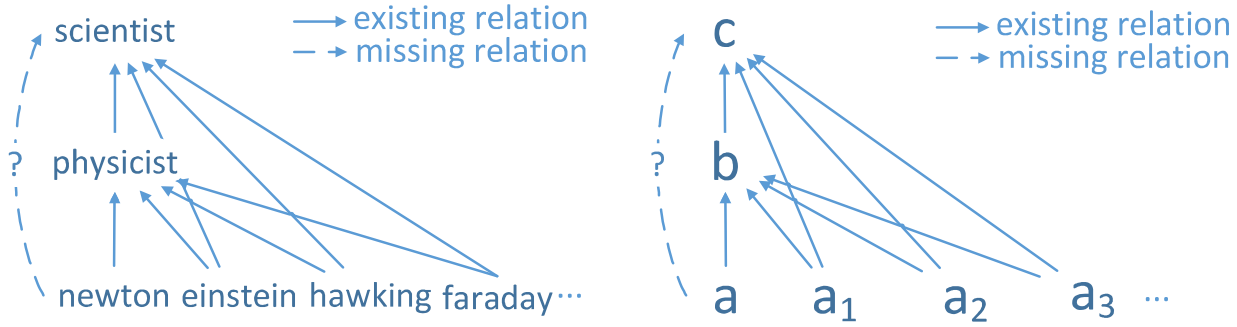
- Our idea
  - A supervised binary classification problem
- Our works:
  - How to build the Labeled data set?
  - How to design effective Features?

Labeled Data Set

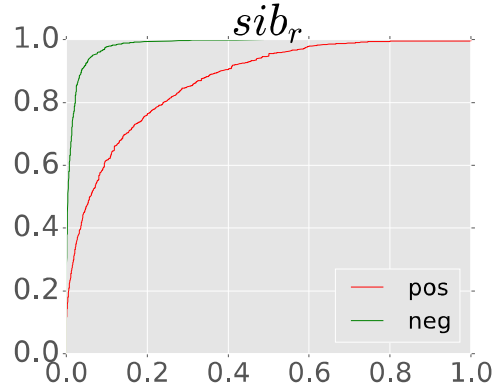
- From WordNet
  - if A isa B(synset1), B(synset1) isa C, then <A,B,C> is positive
  - if A isa B(synset1), B(synset2) isa C, then <A,B,C> is negative
  - About 10k positive samples and 9k negative samples
- Example
  - **tank** has two synsets in WordNet.
    - **tank1=storage tank, tank2=army tank.**
  - In WordNet, we find the following relations:
    - **water tank** isa **tank1, tank1** isa **vessel**,
    - **tank2** isa **military vehicle**
  - Then we construct two samples:
    - <**water tank, tank, vessel**>: POSITIVE
      - Because the **tank** is **tank1**
    - <**water tank, tank, military vehicle**>: NEGATIVE
      - Because **water tank** isa **tank1**, **tank2** isa **military vehicle**, here are two different **tanks**

Inference Mechanisms

Inference from similar instances

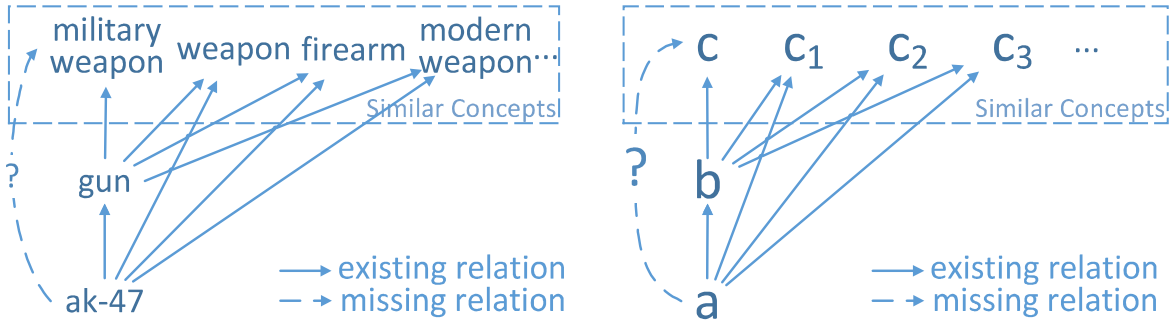


$$sib_r(t) = \frac{|hypo(\mathtt{b}) \cap hypo(\mathtt{c})|}{|hypo(\mathtt{b})|}, t = \langle \mathtt{a}, \mathtt{b}, \mathtt{c} \rangle$$

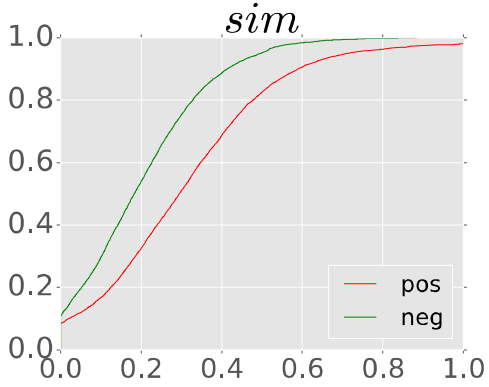


(a) sib_r

Inference from similar concepts



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



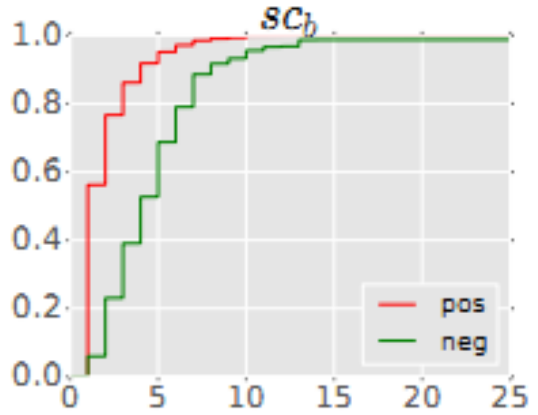
(c) sim

Other signals: Sense(b)

- Sense: sc_b The number of senses of B in WordNet
  - If B not in WordNet, sc(B) = 1
  - Else sc(B) = #synsets that is not in instance level

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

  - Here theta(b) is the number of instance senses
    - For example, the sense **People's Republic of China** is in instance level (it has no instances), will be excluded from sc_b



#	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				

Results

- Top-seven features can help achieve 90% precision
- Randomized decision trees method got best results

Model	Accuracy	Precision	Recall	F1
rbf SVM	0.8009	0.8101	0.8114	0.8107
linear SVM	0.8054	0.8125	0.8186	0.8155
5-NN	0.8634	0.8788	0.8584	0.8685
15-NN	0.8510	0.8748	0.8361	0.8550
50-NN	0.8335	0.8735	0.7990	0.8346
weighted 5-NN	0.8764	0.8926	0.8695	0.8809
weighted 15-NN	0.8683	0.8912	0.8537	0.8720
weighted 50-NN	0.8534	0.8879	0.8253	0.8554
random forest	**0.9187**	**0.9276**	**0.9168**	**0.9221**

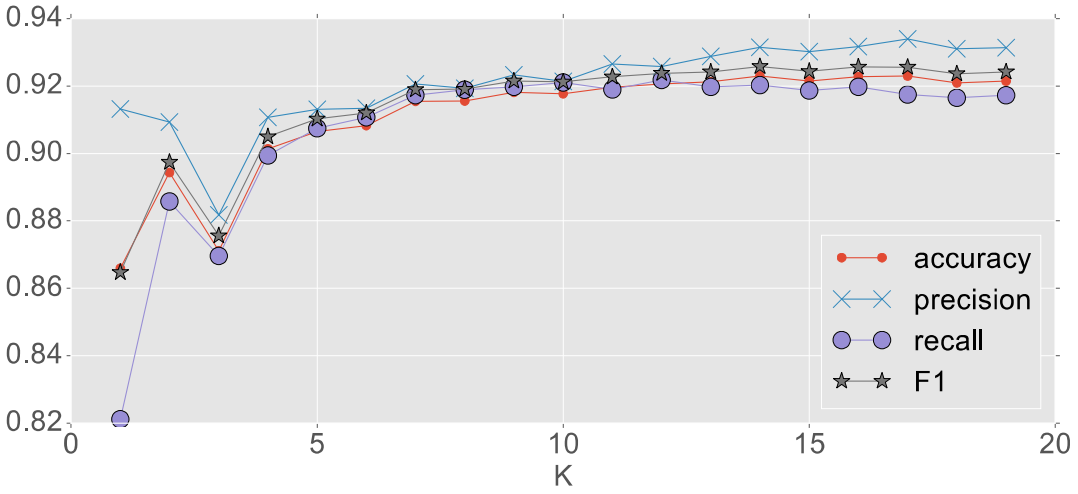


Figure 5: Performance of top-K features

Missing isA relation inference

- For <A,C> pair that has no isA relation, we need to determine whether A isA c holds or not
- For the <A, C> pair, there are many <A, B_i, C>s s.t. A isa B_i, B_i isa C hold, there are many
  - 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 ti = <A, B_i, C>
    - A isa C if and only if most ti are predicted to be positive
  - Weighted voting
    - Sum up the classification scores over each B_i

Finally, find 3.86 million new isA relations with 93% F1-score.

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%**

Pipeline of KG construction

Extraction

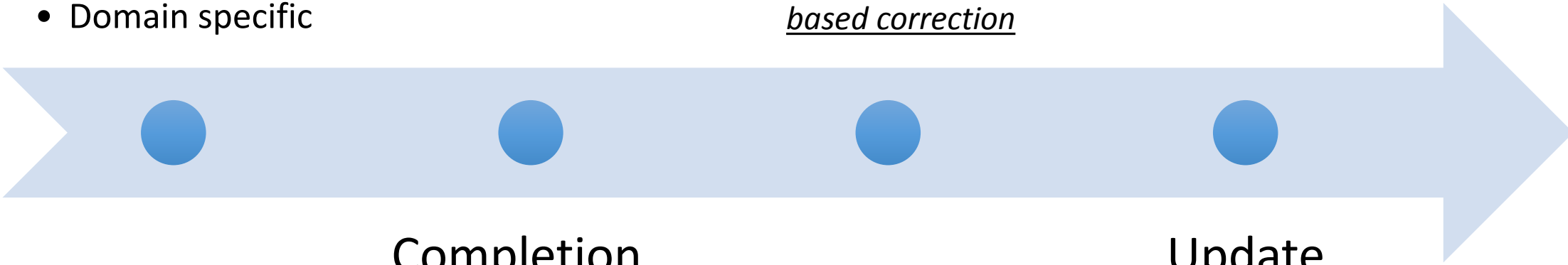
- End-to-end
- Domain specific

Cost:
Reduce Human Efforts

Correction

- *Graph structure based correction*

Quality:
Correct wrong facts



Completion

- Collaborative filtering based completion
- Transitivity inference based completion

Quality:
Find missing facts

Update

Freshness:
Reduce latency

Jiaqing Liang, **Yanghua Xiao***, Yi Zhang, Seung-Won Hwang and Haixun Wang, Graph-based Wrong IsA Relation Detection in a Large-scale Lexical Taxonomy, (***AAAI 2017***)

Problem

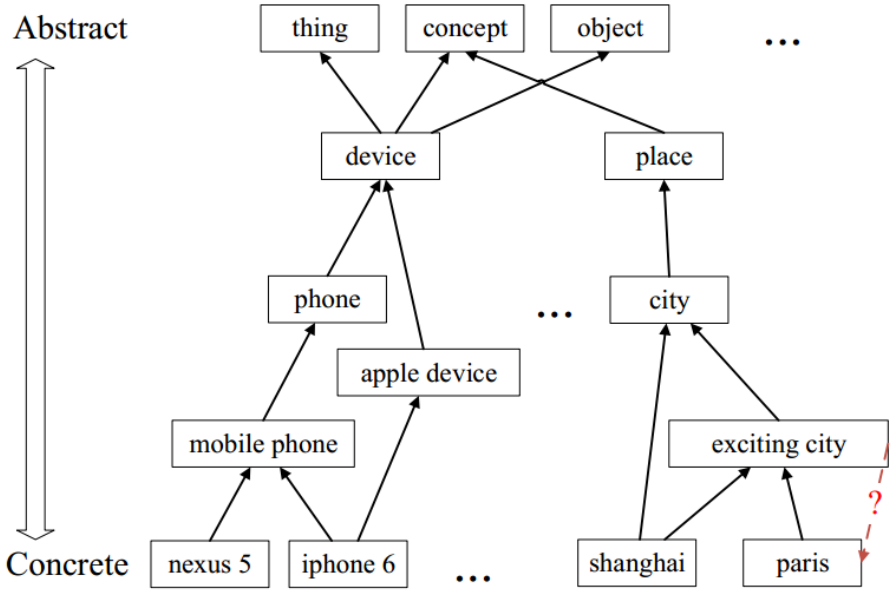
- Wrong isA relations problem in auto-constructed taxonomies
  - Wrong isA relations always exist
    - Natural languages are complex, there are many long sentences and complex sentences.
    - NLP processes can't achieve 100% accuracy, such as stemming, POS tagging etc.
    - The extracting algorithm can't achieve 100% accuracy too.

- Wrong isA relations in Probase
  - The accuracy of Probase is about 92% (original Probase paper)
  - Some wrong isA examples:
    - In the table

Entity	isA	Concept
exciting city	isA	paris
germany	isA	latin american country
packaging material	isA	plastic
music video	isA	youtube video
battery	isA	fuel cell
high traffic area	isA	noise

Observation

- How to detect wrong isA relations?
  - For example: ”paris isA exciting city”, “exciting city isA paris” are conflicts，forming a cycle
- The reason of conflicts may be
  - Errors in the corpus
  - Mistakes made by information extraction algorithms.
    - “... make Paris such **as** exciting city” leads to exciting city *isA* Paris by algorithms that use the such as pattern for extraction.
- Statistical study reveals that cycle tend to contain errors



An ideal taxonomy should be cycle free

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

Cycles in Probase tend to contain errors

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

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

Table 3: Effectiveness of w_f

Reliability Metric- Difference of #Hyponyms

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

- For 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
  - Expression: the higher, the more reliable

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

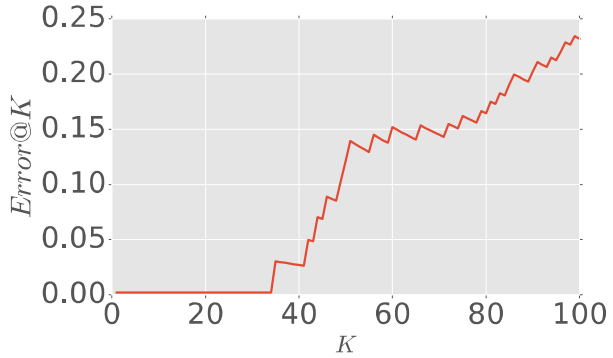


Figure 3: Effectiveness of P_h

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

- Metrics:
  - w(x isA y) = freq(x isA y)
  - w(x isA y) = freq(x isA y) * P_hypo

Model 2: Agony Model

- Agony Model: Find an level assignment such that

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

Penalty function: First, the more errors incurred, the higher the penalty is. Second, the more reliable the edge is in terms of other signals (such as edge weight), 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.

- A dual problem of minimum-cost flow problem, solved by a network flow algorithms

Agony+ Optimization

- The Agony model removes too many edges
- Basic idea:
  - After we remove some edges, some “inversed” edges will not be in a cycle any more
- Agony+
  - Sort all inversed edges by the l(y)-l(x) and weight with ascending order
    - i.e. edges with high wrong probability 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)

Results

Setting	Time	*# result*	Precision	*# truly wrong*
Baseline	3min	281.1K	71.0%	199.5K
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%)

The maximal precision of our models is 91.3%

Setting	Pearson's r (avg $\pm$ std)
Baseline	0.568 ± 0.38
MFAS#1	0.669 ± 0.38
MFAS#2	0.651 ± 0.40
Agony#1	0.639 ± 0.42
Agony#2	$\mathbf{0.692 \pm 0.37}$
Agony+#1	0.623 ± 0.39
Agony+#2	0.684 ± 0.38

Table 5: Evaluation on level assignment

Our model can generate a level assignment that is highly correlated with the levels estimated from Wordnet

Pipeline of KG construction

Extraction

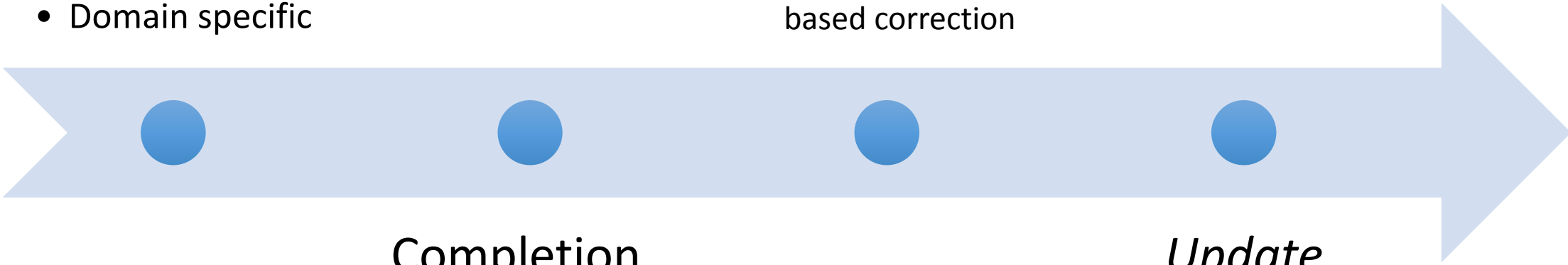
- End-to-end
- Domain specific

Cost:
Reduce Human Efforts

Correction

- Graph structure based correction

Quality:
Correct wrong facts



Completion

- Collaborative filtering based completion
- Transitivity inference based completion

Quality:
Find missing facts

Update

Freshness:
Reduce latency

Jiaqing Liang, Sheng Zhang, **Yanghua Xiao***, How to Keep a knolwedge base synchronized with its sources (**IJCAI *2017***)

Update problem of KBs

- Most of these knowledge bases tend to be outdated, which limits the utility of these knowledge bases
- Many facts in these KBs are changed:
  - American president: from Obama to Trump
  - Trump's profession: from Businessman to President
- It is important to let KB up-to-date
  - Machines can only understand that the topic of an article mentioning Donald Trump is probably related to politics **only when** the KB has the latest fact that **Trump is a president**
  - New entities are continuously emerging such as **iphone 8**.

Synchronization of KB with its data sources

- Good News: the data sources (encyclopedia websites) are up-to-date by many volunteers
- Core problem: How to keep a KB synchronized with online encyclopedia.
  - Way 1: download the latest dump of the encyclopedia website
    - Some website do not provide dump services
    - The dump service is usually update monthly
    - We need to download GBs of data
  - Way 2: crawl each entities from the latest encyclopedia website
    - We need to visit the website too many times (tens of millions of articles)
    - We need to download GBs of data
    - The website will ban us

Key idea

- Key point: update every entity is not only wasteful but also unnecessary
  - Most entities have *stable* properties and are *seldom* changed
    - Basic concepts like orange
    - Historical persons like Newton
  - Only few very hot entities are subject to change
    - such as Donald Trump.
- A smarter strategy:
  - Distinguish the entities subject to change (hot entities) from others with stable properties
  - And then only updating hot entities

- Key: how to estimate the update frequency of an entity in an encyclopedia website.

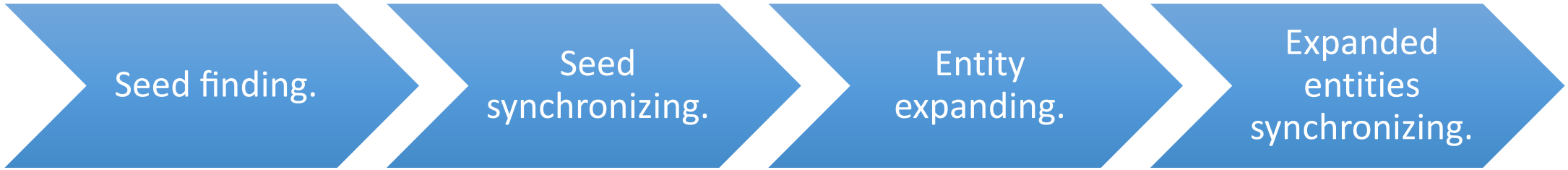
Problem statement

- Select at most K entities (entity set R) to crawl
- And maximize the number of crawled entities that have a newer version than in our knowledge base

$$\arg \max_{R,|R|<K} |\{x|x \in R, t_n(x) > t_s(x)\}|$$

  - $t_n(x)$ is the last update time of x in the online encyclopedia,
  - $t_s(x)$ is the last synchronization time of x
    - if x is a new entity, $t_s(x) = -\inf$

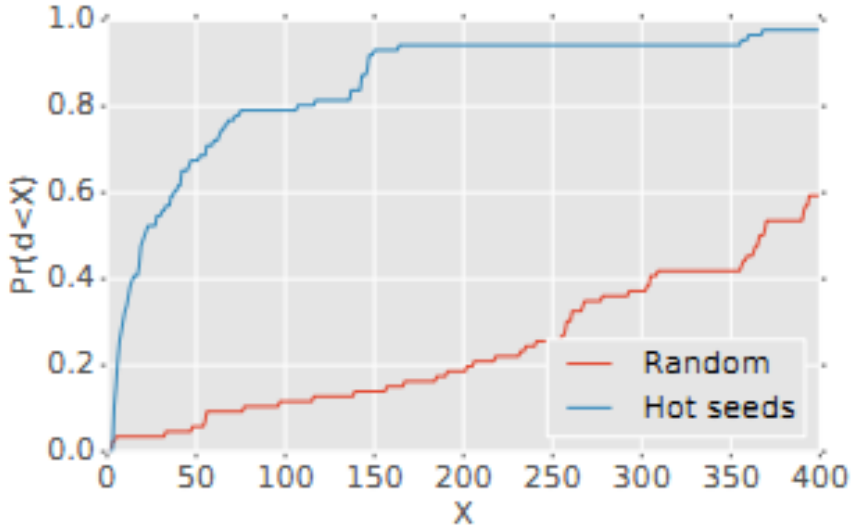
Solution framework



- We find hot entities from Web.
- We synchronize (update or insert) these seed entities.
- We find more related entities via the hyperlinks in the latest pages.
- We synchronize these expanded entities according to their priority, which is provided by a predictor.

Step 1&2: Seed finding & synchronizing

- *Principle 1: If an entity appears frequently on Webs (search engines, online communities or online news), its facts are likely to change.*
- Sources:
  - hot news titles, top search keywords of search engines, hot topics of online communities
- Experiment:
  - d: #days since last update
  - 80% hot seeds are updated in 100 days
  - 10% random entities are updated in 100 days



Step 3: entity expanding

- *Principle 2: An entity that is semantically related to a recently updated entity is more likely to be updated recently.*
  - Examples： Donald Trump’s wife becomes the First Lady after Donald Trump becomes American president.

- Experiment:
  - We find that 269/687 of expanded entities (about 40%) are updated in one month.
  - The ratio is statistically significantly (with a z-score 18.03) larger than random samples (less than 3%).
  - However, synchronizing all the expanded entities is still wasteful.

Step 4: expanded entities synchronizing

- *Principle 3: Any entity that is not in the current KB should has the highest priority to be synchronized.*
- *Principle 4: The entity that has a large expected update times in encyclopedia websites after its last synchronization, deserves a high priority to be synchronized.*

- Priority: $\mathbf{E}[u(x)] = P(x) \times (t_{now} - t_s(x))$
  - expected update times after the last synchronization (E[u(x)])
  - P(x): expected update frequency produced by our predictor
  - $t_s(x)$: the last synchronization time of x, -inf if x is non-existed

Expected update frequency predictor: baseline

- Assumption:
  - The number of updates in an interval (denoted by N) for an entity in encyclopedia websites follows the Poisson distribution.
  - For each entity, we can estimate the parameter λ

$$P(N = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

- However, we use K-S test for ~100k entities and find that the Poisson assumption is not correct
  - First, some new entities have a short history
    - The parameter estimation is inaccurate.
  - Second, the update frequency of an entity might change with time.
    - Trump is not so popular

    before his presidential campaign

Time unit	Amount	Confidence	Test passed	Proportion
Month	94873	0.050	4253	4.48%
Month	94873	0.010	6182	6.52%
Month	94873	0.005	6796	7.16%
Week	94873	0.050	313	0.33%
Week	94873	0.010	436	0.46%
Week	94873	0.005	490	0.52%

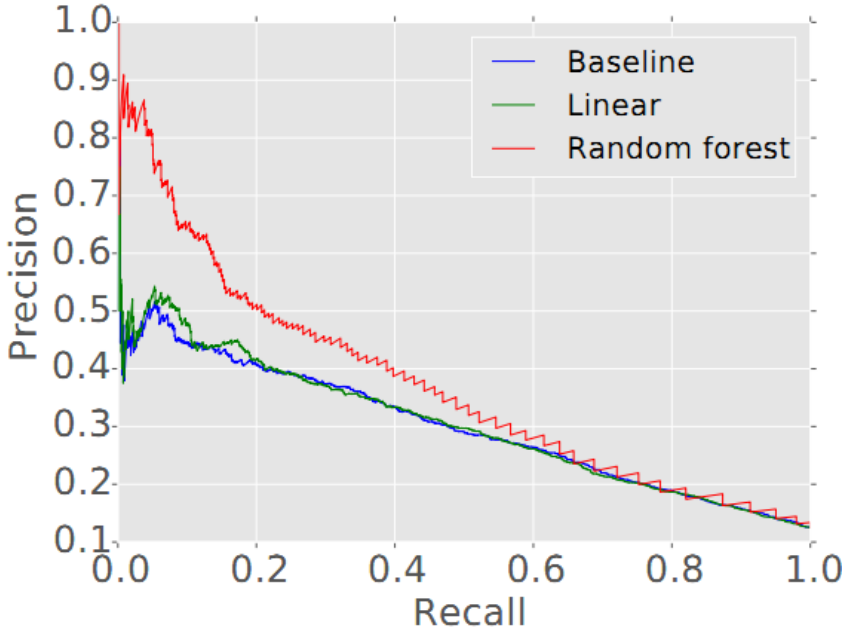
Expected update frequency predictor: regressor

- Machine learning
  - Model:
    - Linear regression
    - Random forest regression
  - Features:
    - In the table, the chi^2 and IG show their effectiveness
  - Labeled data:
    - Select a time stamp T (we set T as one month before now)
    - Use the KB snapshot at T for features
    - Predict the update times for each entities in time span (T,T+30]

#	Feature	χ^2	IG(10^{-3})
1	#Weeks of existence	41.8	19.1
2	#Total updates	**481.1**	**55.9**
3	#Times viewed by users	203.5	46.2
4	#All hyperlinks	460.9	35.8
5	#Hyperlinks to entities	444.9	32.1
6	Page length	131.9	32.9
7	Main content length	202.1	19.1
8	Historical update frequency	287.6	54.7

Experiments for the predictor: hold-out evaluation

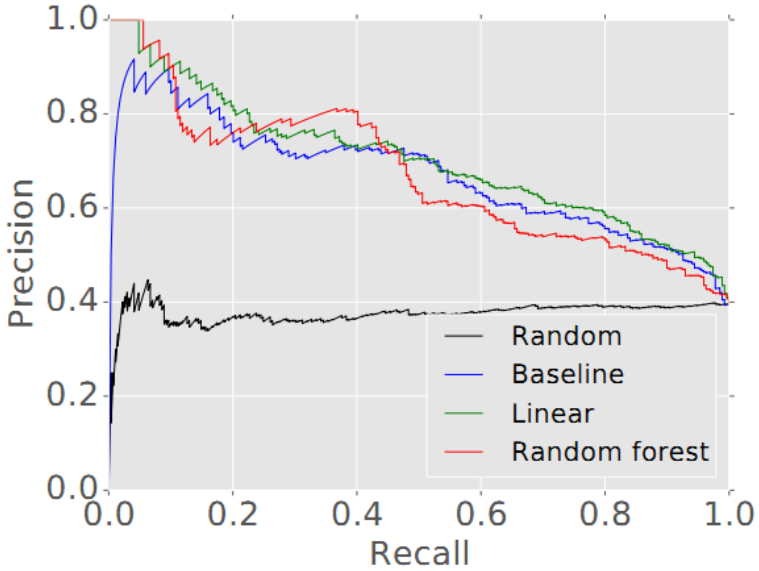
- Use 90% labeled entities for training, 10% labeled entities for testing
  - Our models outperform the baseline



Model	MSE	AUC
Baseline	0.0400	0.2992
Linear	0.0367	0.3021
Random forest	**0.0315**	**0.3692**

Experiments for the predictor: real updating

- Our two models generate better rankings than the random ordering and the baseline
- Random forest model in general has a better performance when we are only interested in the top sublist
- We prefer random forest model, since we will drop tailed entities



Metric	Random	Baseline	Linear	RF
MAP	0.379	0.670	**0.704**	0.671
nDCG	0.800	0.916	**0.941**	0.933
AUC	0.376	0.666	**0.700**	0.667
Precision@20	0.400	0.850	0.900	**0.950**
Recall@20	0.030	0.063	0.067	**0.071**
F1@20	0.055	0.118	0.125	**0.131**
Precision@100	0.350	0.730	0.750	**0.790**
Recall@100	0.130	0.271	0.279	**0.294**
F1@100	0.190	0.396	0.407	**0.428**

Experiments for the system

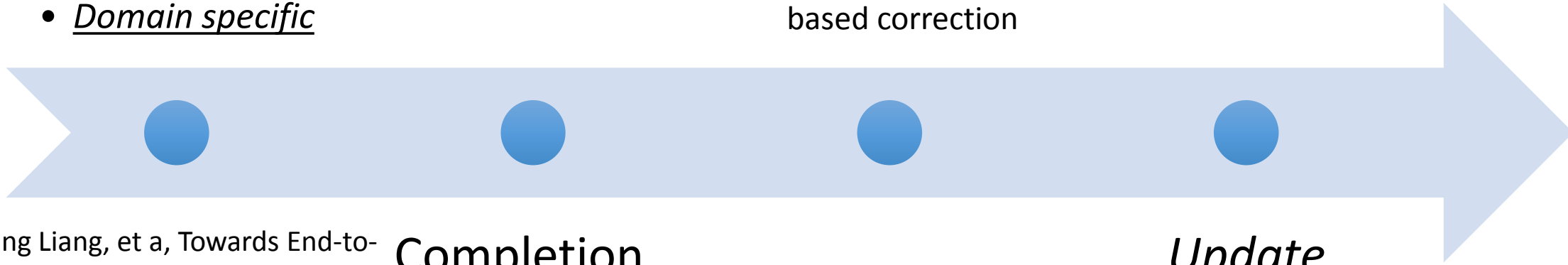
- We deployed the system on our knowledge base, a DBpedia-like Chinese knowledge base extracted from BaiduBaike.
- We set K (upper access limit) as 1000 so that the crawling will not be banned by the website.
- Our system crawl 1000 entities in one day, and 68.7% of them have newer versions.

Total visits	Success updates	Success ratio
50	46	92.0%
100	90	90.0%
200	175	87.5%
500	398	79.6%
1000	687	68.7%

Our update system for knowledge base

- The system is deployed on our knowledge base CN-DBpedia
  - http://kw.fudan.edu.cn/cndbpedia/
- The data source is the largest Chinese encyclopedia BaiduBaike
  - https://baike.baidu.com/
- The system only updates few entities per day, most of them actually contain newly updated facts.

Pipeline of KG construction



Extraction

- *End-to-end*
- *Domain specific*

Cost:
Reduce Human Efforts

Correction

- Graph structure based correction

Quality:
Correct wrong facts

Completion

- Collaborative filtering based completion
- Transitivity inference based completion

Quality:
Find missing facts

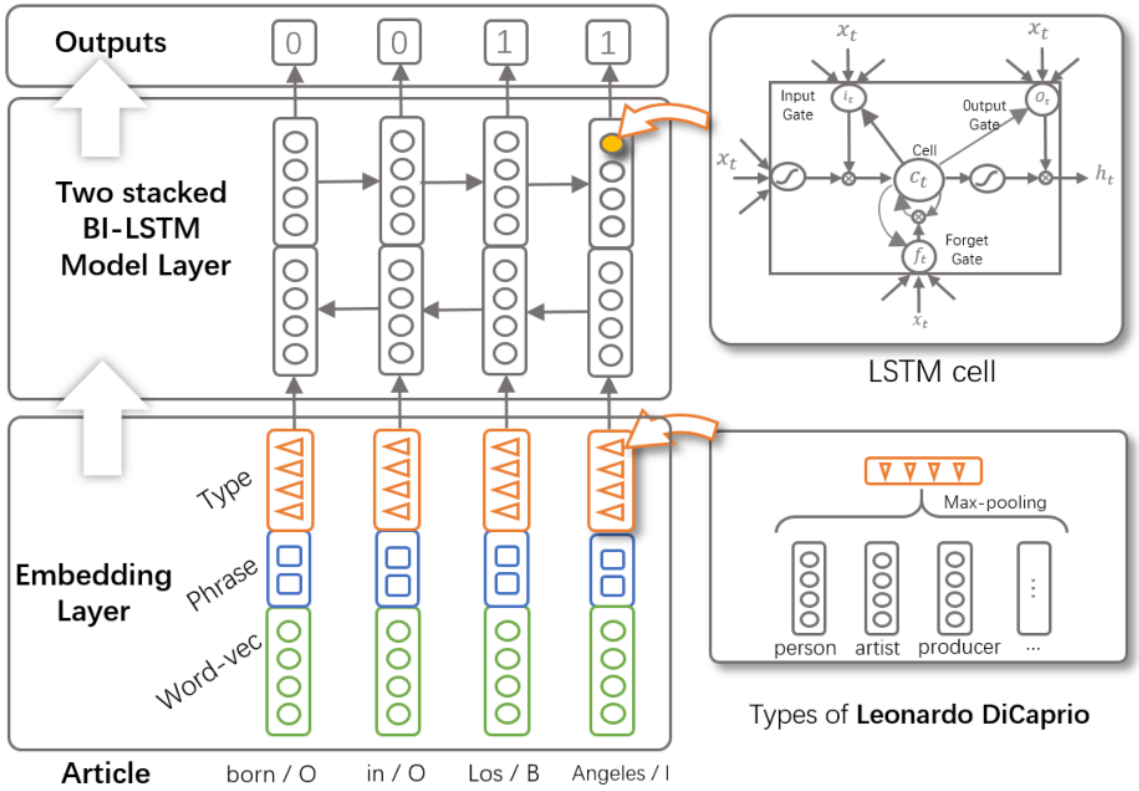
Update

Freshness:
Reduce latency

- Jiaqing Liang, et a, Towards End-to-End Knowledge Graph Construction via a Hybrid LSTM-RNN Framework (under review)
- Wanyun Cui, Transfer Learning for Domain-Specific NLP Applications with Progressive Neural Networks (under review)

End-to-end knowledge extraction

- Previous extraction approach has high human cost
  - Data labeling
  - Feature engineering
- Our idea
  - Using distant supervision for automatic data labeling
  - LSTM based end-to-end extraction



Results

Predicate	#Sample	w2v only			w2v+phrase			w2v+phrase+type		
		Prec.	Reca.	F1	Prec.	Reca.	F1	Prec.	Reca.	F1
activeYearsStartYear	51,078	0.786	0.708	0.745	0.792	0.695	0.741	0.705	0.809	0.754
artist	70,754	0.867	0.813	0.839	0.874	0.861	0.867	0.893	0.864	0.878
birthPlace	150,787	0.719	0.736	0.728	0.746	0.729	0.738	0.762	0.706	0.733
class	2,172	0.903	0.787	0.841	0.869	0.833	0.851	0.924	0.851	0.886
country	334,118	0.888	0.938	0.912	0.889	0.940	0.914	0.888	0.951	0.918
deathPlace	36,164	0.599	0.478	0.532	0.613	0.466	0.529	0.639	0.417	0.505
family	106,920	0.956	0.963	0.959	0.959	0.972	0.965	0.960	0.970	0.965
foundedBy	3,437	0.663	0.355	0.462	0.836	0.544	0.659	0.834	0.527	0.645
genre	9,306	0.699	0.353	0.469	0.689	0.441	0.538	0.642	0.493	0.558
genus	116,174	0.937	0.957	0.947	0.948	0.979	0.963	0.947	0.979	0.963
isPartOf	192,418	0.889	0.952	0.919	0.905	0.955	0.929	0.888	0.972	0.928
location	131,124	0.808	0.711	0.756	0.793	0.784	0.788	0.806	0.783	0.794
occupation	6,399	0.441	0.163	0.238	0.547	0.239	0.332	0.576	0.277	0.374
order	4,939	0.867	0.748	0.803	0.918	0.768	0.836	0.928	0.768	0.840
populationTotal	63,626	0.946	0.986	0.965	0.948	0.986	0.966	0.957	0.976	0.967
position	2,316	0.863	0.711	0.780	0.831	0.791	0.811	0.853	0.691	0.764
postalCode	3,268	0.979	0.967	0.973	0.967	0.976	0.971	0.979	0.982	0.980
previousWork	7,271	0.609	0.391	0.477	0.698	0.523	0.598	0.695	0.570	0.626
producer	46,795	0.749	0.559	0.640	0.684	0.757	0.718	0.732	0.706	0.719
recordLabel	34,414	0.853	0.828	0.840	0.885	0.852	0.868	0.872	0.898	0.885
starring	44,621	0.755	0.829	0.790	0.831	0.875	0.852	0.846	0.855	0.851
subsequentWork	3,435	0.639	0.202	0.307	0.611	0.371	0.462	0.598	0.415	0.490
team	55,110	0.803	0.823	0.813	0.854	0.859	0.857	0.850	0.872	0.861
type	15,275	0.783	0.535	0.635	0.809	0.538	0.647	0.780	0.559	0.651
writer	31,240	0.784	0.679	0.728	0.788	0.813	0.800	0.817	0.789	0.803
TOTAL	1,523,161	0.834	0.819	0.826	0.846	0.851	0.848	0.849	0.853	0.851

Held-out automatic evaluation

Table 1: DBpedia held-out evaluation results

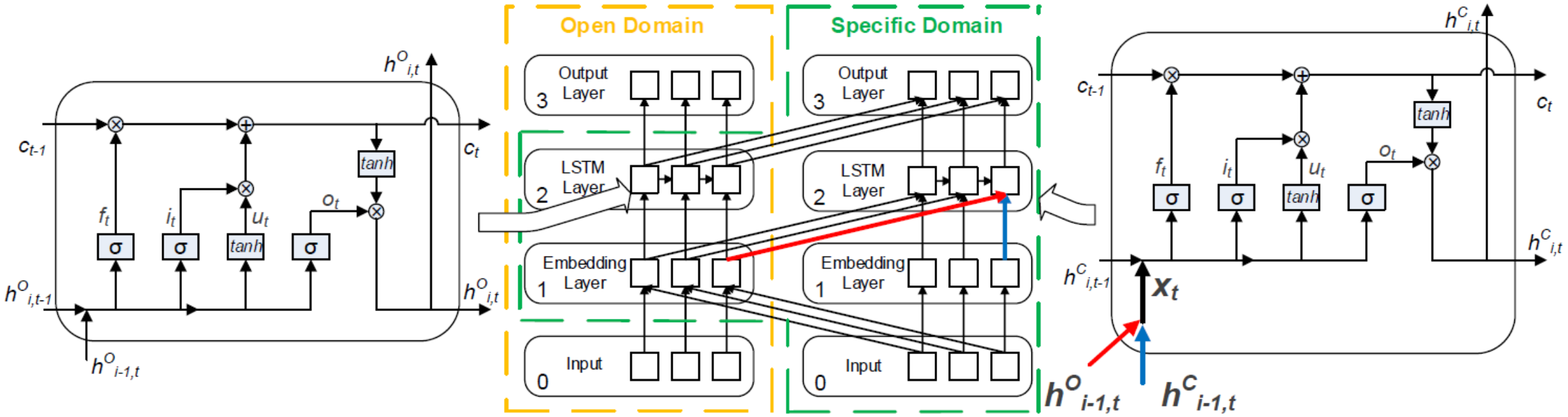
Method	Precision	Recall	F1
Rule-based	0.235	0.023	0.042
CRF baseline	0.268	0.656	0.381
Single forward LSTM	0.820	0.781	0.800
Single backward LSTM	0.817	0.828	0.822
Bi-direction LSTM	0.834	0.819	0.826
W2v+Phrase	0.846	0.851	0.848
W2v+Phrase+Type	**0.849**	**0.853**	**0.851**

Predicate	DS	Our Model
people/person/place_of_birth	0.78	**0.98**
people/deceased_person/place_of_death	**0.81**	0.79
people/person/nationality	0.72	**0.90**
location/location/contains	**0.84**	0.76
film/director/film	0.44	**0.66**
music/artist/origin	0.71	**0.81**

Manul evaluation

Domain-Specific knowledge extraction

- Models for the open domain usually incur more errors in specific domains.
- Training a model for specific domain faces the challenge of the **insufficiency of the specific domain's labeled data**.
- Our idea: transfer knowledge from the open domain to specific domains
- To fully utilize open domain knowledge, multi-level knowledge should be transferred
  - Entity level: word embedding
  - Sentence level: RNN-LSTM
  - Knowledge transfer -> Weight matrix transfer



- The two overlapping layers store the knowledge (parameter matrices) of open domain.
  - Their parameter matrices are freezed during the specific domain training.
- Lateral connections are used for transferring open domain knowledge to the specific domain.
  - The LSTM layer and output layer receive transferred knowledge from embedding layer and LSTM layer, respectively.

- LSTM layer $x_t = h^S_{1,t} + W_1 h^O_{1,t}$

$$
\begin{aligned}
i_t &= \sigma(W^{(i)} x_t + U^{(i)} h^S_{2,t-1} + b^{(i)}), \\
f_t &= \sigma(W^{(f)} x_t + U^{(f)} h^S_{2,t-1} + b^{(f)}), \\
o_t &= \sigma(W^{(o)} x_t + U^{(o)} h^S_{2,t-1} + b^{(o)}), \\
u_t &= tanh(W^{(u)} x_t + U^{(u)} h^S_{2,t-1} + b^{(u)}), \\
c_t &= i_t \odot u_t + f_t \odot c_{t-1}, \\
h^S_{2,t} &= o_t \odot tanh(c_t)
\end{aligned}
$$

Our new system

- CN-DBpedia 2.0: An Never-Ending Knowledge Base Extraction System
  - Multiple domain fusion
  - Fully pipelined
  - Actively updated
  - Real-time Updated
  - Almost fully automated
  - Never ending extraction
  - Highly configurable
  - Easily adaptable for new domains

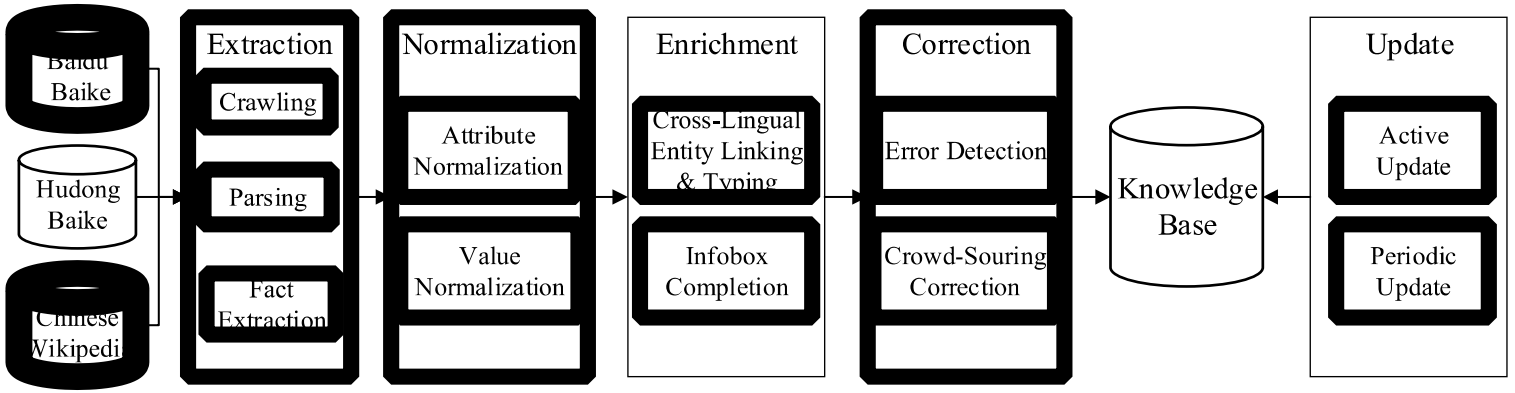


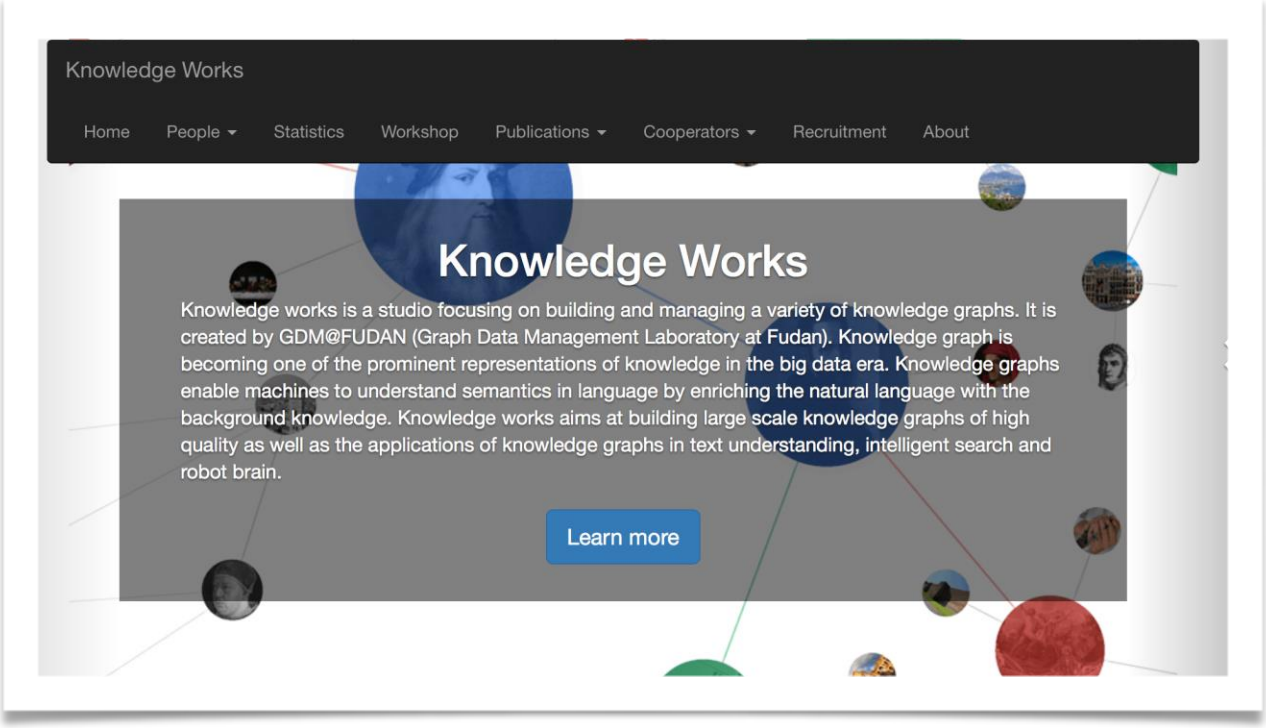
Fig. 1. System Architecture of `CN-DBpedia`.

Our Lab

- **Kowledge Works@FUDAN**

- http://Kw.fudan.edu.cn
- Knowledge works is a studio focusing on building and managing large scale knowledge graphs of high quality as well as the applications of knowledge graphs in text understanding, intelligent search and robot brain.

- **Graph Data Management Lab@FUDAN**
- http://gdm.fudan.edu.cn
- **GDM@FUDAN** focuses on studying and developing effective and efficient solutions to manage and mine these graph data, aiming at understanding real graphs and supporting real applications built upon large real graphs. Recently, we are especially interested in knowlege graphs and its application.



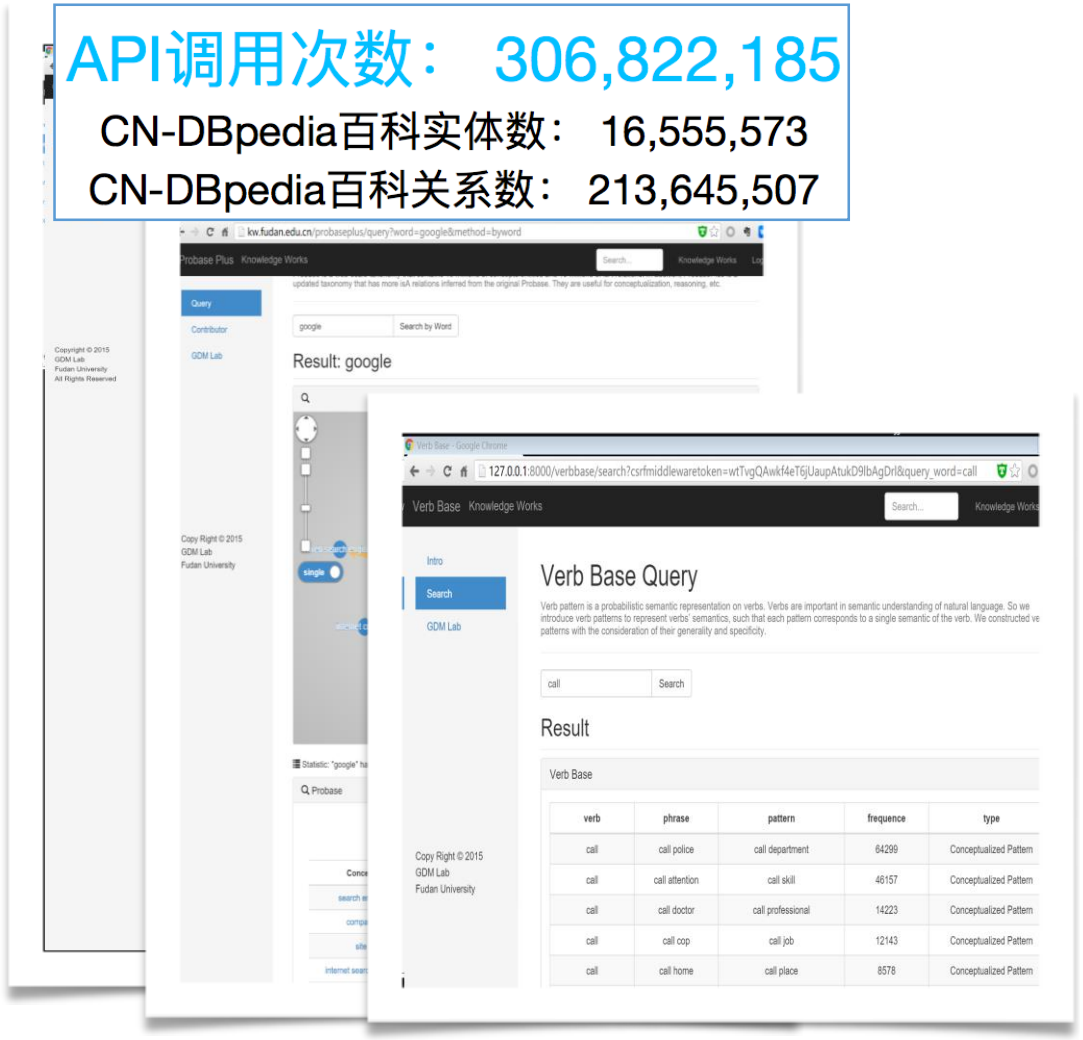
Knowledge Graph Service

1. CN-DBPedia CN-DBpedia is an effort to extract structured information from Chinese encyclopedia sites, such as Baidu Baike, and make this information available on the Web. CN-DBpedia allows you to ask sophisticated queries against Chinese encyclopedia sites, and to link the different data sets on the Web to Chinese encyclopedia sites data

2. Probase Plus Probase is a web-scale taxonomy that contains 10 millions of concepts/entities and 16 millions of isA relations. In addition, ProbasePlus is a updated taxonomy that has more isA relations inferred from the original Probase. They are useful for conceptualization, reasoning, etc

3. Verb Base

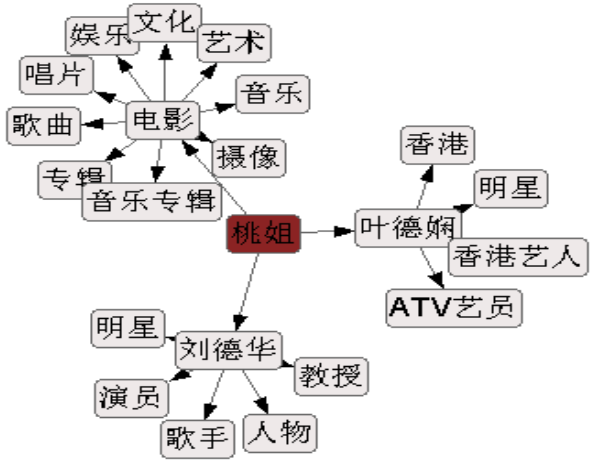
Verb pattern is a probabilistic semantic representation on verbs. We introduce verb patterns to represent verbs' semantics, such that each pattern corresponds to a single semantic of the verb. We constructed verb patterns with the consideration of their generality and specificity.



Our Mission: The construction, management and application of large scale knowledge graphs

Knowledge Graph

a kind of semantic network that consists of entities/concepts as well as their semantic relationships. Higher coverage over entities and concepts, more abundant semantic relationships, constructed in an more automatic way, higher accuracy is expected.



The key of intelligent information processing.

KG has shown its potential power in solve problems such as search intent understanding, relationship explaining, user profiling. It is of great business value in intelligent search, intelligent software, cybernetic security and intelligent business.

The key to build a machine that think like human

KG provides necessary background knowledge to enable machine to understand language and think like human.

Research Outline

Graph Analytic

1、 Models for symmetry (Physical Review E 2008)
2、 Graph Simplification (Physical Review E 2008)
3、 Complexity/distance measurement (Pattern Recognition 2008, Physica A 2008)
4、 Graph Index Compression (EDBT2009)
5、 Graph anonymization (EDBT2010)

Big Graph Management

1、 Big graph systems(SIGMOD12)
2、 Overlapping community search (SIGMOD2013)
3、 Local Community search (SIGMOD2014)
4、 Big graph partitioning (ICDE2014)
5、 Shortest distance query (VLDB2014)
6、 Fast graph exploration (VLDB 2016)

Knowledge Graph Construction

1、 IsA taxonomy completion (TKDE2017)
2、 Implicit isA relation inference (AAAI2017)
3、 Error isA correction (AAAI2017)
4、 Cross-lingual type inference (DASFAA2016)
5、 Synchronization of KG (IJCAI 2017)
5、 End-to-end knowledge harvesting
6、 Domain-specific knowledge harvesting

Natural Language Understanding by KG

1、 Understanding bag of words (IJCAI2015)
2、 Understanding a set of entities (IJCAI 2017)
3、 Understanding verb phrase (AAAI2016)
4、 Understanding a concept (IJCAI 2106)
5、 Understanding short text (EMNLP2016)
6、 Understanding natural languages (IJCAI2016, VLDB2017)

Knowledgable Search/Recommendation

1、 Recommendation by KG (WWW2014、 DASFAA2015)
2、 User profiling by KG (ICDM2015、 CIKM2015)
3、 Categorization by KG (CIKM 2015)
4、 Entity suggestion with conceptual explanation (IJCAI 2017)
5、 Entity search by long concept query

Thanks