From hyperlinks to semantic links

A Probabilistic Approach

ABSTRACT

In this paper, we propose an approach for detectiing the implicit relations between 2 entities.

1. INTRODUCTION

Semantifying web documents is a huge task. Recently, more and more hyperlinked documents emerge. There are almost semi-structured information everywhere, but unfortunately the links lacks an explanation of why they are connected, and why human may consider them related[]. The types of hyperlink are article to article, article to knowledge base(or encyclopedia-like databases). Each link has two ends the original one(the web page/ entity page contains it) and the oriented one(the web page/ entity page it points to), and each end has 2 possible types, an anchor url or an anchor entity. Hence, in all there are 4 possible combinations, which consists the 4 type of links. An anchor entity can be named entities such as movies, products, people names, locations etc. In this case we explain the relation between the entity and the original article. As for an anchor url, an automatic target entity extraction process[2011] can be performed, and then it become another semantifying problem between entities problem.

In this paper we focus on the problem of semantifying the first type of hyperlinks since other kinds of links can finally be deduced to this problem.

We argue that the context of an entity in an article is informative[] and deserve a higher occurrence[] and can produce fresh and latest relation of an entity, which can be useful in updating the KB[reverb]. In our approach the relations is not necessarily to be exist, however the cluster of the relation[relation clustering] it belongs to should be conceptually right. For instance, the relationship `artist of` will be correct in the tuple of `(Leonardo Da Vinci, artist of, Mona Lisa)` [`(Human, artist of, Painting)`] and will be never correct in the tuple of `(Automobile, artist of, Painting)` [e.g.`(BMW i8,artist of, Mona Lisa )`] With rich context and large knowledge base, we can easily derive the fresh context relations and the concept of each entity,

On the other hand, we consider the co-occurrence of the 2 entities, based on the assumption of important relationship will be observed in various of documents[]

For these reasons, we purpose a relation explanation method leveraging the concept and co-occurrence of and entity, to explain the relation between the target entity and the related entity, thus semantifying the hyperlink.

2. RELATED WORKS

2.1 Link entities to Database

Linking entities to database, especially to Wikipedia, has been widely studied. Entity linking to Wikipedia [10, 9, 5] exploit Wikipedia as thesaurus and link web documents to it. In our work, instead of linking entities to the correspond one in KB, we extract the target entity [1] and explain the semantic relation of the entity towards target entity.

2.2 Relation Discovery

to be read [12, 11, 7]

Recently, different efforts are devoted to relation Discovery [3, 13, 8] are studied on graph based approaches and text based approaches [6][reverb]. However, these approached will be limited largely by the incompletion of Knowledge Base, and cannot discover new type of relations. In our work, we focus on semantic relations of entity by leveraging the concepts of entity. extracted from knowledge bases to link entities with a probability.

2.3 Information Extraction

Attribute acquisition methods Among domain-dependent approaches, we can mention approaches that focus on products. In this domain, attributes have been used to improve product search and recommendation [18, 22], but also to enable data mining [27]

Attribute retrieval provides another granularity in Web search. This can interest communities that propose a more focused access to information or communities that envision aggregating pieces of information such as aggregated search [19, 15]. Wong et al. [27] combine tags and textual features in a Conditional Random Fields model to learn attribute extraction rules, but they need a seed of relevant documents manually fed.

2.4 Short Text Conceptualization

3. FRAMEWORK

In this section, we present the problem and our solution to it.

3.1 Problem Statement

For a given pair of entity, we formalize the problem of explaining the relationship between into getting a ranking of all the possible attributes. Our target is to calculate the typicality of an attribute that can best define the possible relation of the 2 entities, which can be denoted as $P(a|(e_1, e_2))$. One possible way is to store all the entities and their relationships, so that we can find an accurate attribute for each entity pair. Unfortunately, it is almost impossible to achieve this since there are billions of entities and new entities emerges everyday. Thus we need to utilize the concepts of the entities. Eq. 1 breaks down our target into 2 parts: 1)Calculating the typicality of an attribute towards a concept pair. 2) Calculating the typicality of the concept pair towards an entity pair.

$$P(a|(e_1, e_2)) = \sum_{c_1 \in C_1, c_2 \in C_2} P(a|(c_1, c_2)) \times P((c_1, c_2)|(e_1, e_2)), \tag{1}$$

where $P((c_1, c_2)|a)$ represents the probability of given an attribute, how likely is a concept pair going to appear. We use Bayesian theorem to convert the first part in Eq.2:

$$\begin{aligned} P(a|(c_1, c_2)) &= \frac{p((c_1, c_2)|a) \times P(a)}{P((c_1, c_2))} \\ &= \frac{p((c_1, c_2)|a) \times P(a)}{\sum P((c_1, c_2)|a^*) \times P(a^*)}, \end{aligned} \tag{2}$$

where $P(a)$ is defined by Eq. 3, denoting how frequent attribute occurs.

$$P(a) = \frac{n(a)}{\sum n(a^*)}, \tag{3}$$

where $n(a)$ is the count of the attributes.

$P((c_1, c_2)|(e_1, e_2))$ denotes the likelyhood of the occurrence a concept conbination given the entity, one naive solution is as Eq. 4, following the intuition of finding typical concepts for the entity. We discuss how to calculate $P(c|e)$ for this task in detail in Section. 4.

$$P((c_1, c_2)|(e_1, e_2)) = P(c_1|e_1) \times P(c_2|e_2) \tag{4}$$

However, the joint distribution of the two concepts is indispensable for sense disambiguation. For example, we suppose the given entity pair is (apple, steve jobs). The potential left concepts are `fruit,company,...` and the right concepts are `entrepreneur, pc developer,...`. Obviously, we cannot combine `fruit` with `entrepreneur` in this case. Therefore, the Joint Distribution $JD(c_1, c_2)$ is introduced in Eq. 5

$$P((c_1, c_2)|(e_1, e_2)) = JD(c_1, c_2) \times P(c_1|e_1) \times P(c_2|e_2) \tag{5}$$

Combining all these together, the complete version of $P(a|(c_1, c_2))$ is described in Eq. 6

$$\begin{aligned} P(a|(e_1, e_2)) = &\frac{p((c_1, c_2)|a) \times P(a)}{\sum P((c_1, c_2)|a^*) \times P(a^*)} \\ &\times JD(c_1, c_2) \times P(c_1|e_1) \times P(c_2|e_2) \end{aligned} \tag{6}$$

3.2 Problem Solution

Thus, the main framework is divided into 2 parts, the online part and the offline part.

The offline part.

According to the above derivation, there are 2 aspect we need to calculate offline: $JD(c_1, c_2)$ and $P((c_1, c_2)|a)$. We leverage the plentiful (e_1, a, e_2) tuple in DBpedia and their concepts to compute

The online part.

For an entity pair request, we can calculate $P(a|(e_1, e_2))$ using Eq. 6, where we get a typicality score for each attribute, once we get $P((c_1, c_2)|a)$ and $JD(c_1, c_2)$. Hence, we get a ranking of the attributes for the entity pair.

Paper Orgnization.

The rest of the paper is organized as follows, Section 4 describe how to derive $P(c|e)$ leveraging the basicness of concept, Section 5 is devoted to the offline calculation of $JD(c_1, c_2)$ and $P((c_1, c_2)|a)$.

4. PROBABILITY RECALCULATION AFTER CONCEPTUALIZATION

4.1 Problem Statement

intro about probase

Given an entity e, from Probase, we can acquire its concepts' set C and for each $c_i \in C$, the frequency $n(c_i, e)$ can be accordingly derived, which means how many times the e isA c_i pattern can be observed from the original corpus.we can derive $P(c_i|e)$, where $c_i \in C_{probase}$,

$$P(c_i|e) = \frac{n(c_i, e)}{n(e)}$$

However, the concepts here has various forms as illustrated in Example 1. For our task, we only need relatively general concepts(a.k.a. head concepts).

EXAMPLE 1 (VARIOUS FORMS OF CONCEPTS). *Take the entity* `Mona Lisa` *as example, its concepts includes* `painting, famous painting, world's most famous painting`*, with corresponding frequency* `33,8,1`

We divide the $C_{Probase}$ into 2 parts, C_{simple} and C_l, where C_{simple} only contains one word and C_l are the rest. C_{simple} are generated by the head modifier detection. The problem here is to recalculate the probability $P(c_h|e)$ where $c_h \in C_{simple}$, literally, we should contribute all the counts of C_l to C_{simple}.

Why head concepts?.

The relationship between entities are determined by the simple concepts. For example, the `founder` relationship between `Apple Inc.` and `Steve Jobs` are determined by the head concepts they possessed (e.g. `company` and `entrepreneur`, regardless of the modifiers such as `technology` in the concept `technology company`.) The number of entities can be very large, but the number of top concepts and the relationship between them are limited, literally, we can find all the possible relationship between concepts instead of store all the long-tailed entities and their relations, which indicates the rationality of doing conceptualization. The reason why we do head modifier detection instead of directly using isA edge in probase, is that even if the long concept c_l has an isA edge towards a certain concept $c_h{}'$, it still sometimes not include the head concept of the long concept which is very plausible as is demonstrated in Example 2

EXAMPLE 2 (HEAD CONCEPTS VS ORIGINAL CONCEPTS). *Take* `famous painting` *as example. Its original concepts are* `image, treasure`*, which are reasonable but not plausible, since their occurrence are 2 and 1 respectively. However, the most plausible concept* `painting` *is not among the concepts.*

The main steps.

Given an entity e from *Probase*, we can get its concepts from probase. First we do head modifier detection based on syntax[], since the concepts in Probase all follows English grammar, this approach already produces a good result. Next, we recalculate the probability of $P(c_h|e)$ by aggregating the contribution from c_l. The essentiality of doing so is illustrated in Example 3. Finally, we provide a method to take the original isA relation from Probase into consideration.

EXAMPLE 3 (ESSENTIALITY OF AGGREGATION). `steve jobs` *The concept*`well-known name` *has four occurrences however* `name` *has only 2. There are other modifiers for the same head, so that the typicality of the head will be largely underestimated.*

4.2 Baseline

After head modifier detection, we have a set of $c_h \in C_{simple}$, among all the $c_{l_j} \in C_l$, there are 2 cases in the probase determined by whether the c_{l_j} has an isA edge towards c_h or not. The intuition of doing so is illustrated in the Example 4:

EXAMPLE 4 (CONTRIBUTING LONG CONCEPTS). *Assume that* `Mona Lisa is a painting` *and* `Mona Lisa is a famous painting` *are observed respectively* `33 times and 8 times` *from different documents, we will get the knowledge that* `Mona Lisa is a painting` *occurs* `41 times` *instead of* `33 times`*.*

Hence, the most straight forward approach is to contribute the corresponding long concepts to the simple ones as follows:

$$\hat{P}(c_h|e) = \frac{n(c_h, e) + \sum_{f_{HM}(c_l)=c_h} n(c_l, e)}{\sum n(c_h^*, e)}$$

where $f_{HM}()$ is a function that takes a long concept and produce a head concept.

4.3 Combined Model with Original IsA

When obtaining $\hat{P}(c_h|e)$, we are in fact judging the typicality of a concept. In this section, we take the original Probase IsA relation into consideration. In Example. 5, we can observe some reasonable results produced by the Probase isA relationship.

EXAMPLE 5 (RESONABLE ISA RELATION). *There exists several original IsA concepts of the long concepts that are also reasonable. For example* `topaz`*(a kind of yellow gemstone) has the concept* `precious stones`*, and* `precious stones` *has an edge towards* `material` *which is reasonable.*

Based on the how to treate c_l, we have the following 2 cases:

c_l appear as an concept In this case the counts that its entities produced should be take into consideration into c_h, deriving the following Case A.

c_l appear as an instance In this case c_l is observed from the corpus as an instance at the left side of an isA sentence, it should be treated the same as other entities e, deriving the following Case B.

Therefore, to calculate $P(c_h|e)$, there are three cases.:

Case A.1 $e \xrightarrow{isA} c_h$ The entity has has an isA edge towards one or more simple concept, which gives the original $P_{org}(c_h|e) =$

Case A.2 $e \xrightarrow{isA} c_l \overset{Head}{\dashrightarrow} c_h$ The solid edge here refers to the isA relationship in *Probase* and the dashed one refers to the edge generated by head modifier detection. Example 6 pointed out that there won't be necessarily an isA edge from `famous painting`(c_l) to `painting`(c_h), however c_l is obviously a hyponym of c_h. In this case, we have to re-calculate the $P(c_h|c_l)$. In the original probase approach, we use Eq. 7 to calculate the probability.

$$P(c_h|c_l) = \frac{n(c_h, c_l)}{\sum n(c_h{}^*, c_l)} \tag{7}$$

However, $n(c_h, c_l)$ is lower than expected due to the reason demonstrated in Example. 6. Therefore, we alternatively utilize the $\sum e^*, n(c_l)$ as the occurrence of c_l, following the assumption that *whether c_l is typical towards its c_h is independent from*

$$P_{head}(c_h|e) = \frac{\sum_{c_h=f_{HM}(c_l*)} n(c_l*, e)}{n(e)}$$

Case B $e \xrightarrow{isA} c_l \xrightarrow{isA} c_h$ *In this case, we need to calculate the following equation*

$$P(c_h|e) = \sum_{c_l^* \in C_l} P(c_h|c_l^*, e) \times P(c_l^*|e)$$

, where $P(c_l^*|e)$ can be obtained from Probase and

$$P(c_h|c_l, e) = \frac{n(c_h, c_l, e)}{n(c_h, e)} \tag{8}$$

We assume that the occurrence of e does not affect $P(c_h|c_l)$ equivalently speaking, $P(c_h|c_l)$ is independent *from e, thus Eq. 8 can be simplified*

$$P(c_h|c_l, e) = P_{probase}(c_h|c_l) = \frac{n(c_h, c_l)}{n(c_h)}$$

which can be obtained from Probase.

Example 6 (Why aren't head relationship observed). *There are less chance of occurring* `Famous painting is a painting` *in the corpus, since human takes it for granted and will seldom express it in such a way, so that there won't be necessarily an isA edge from* `famous painting` *to* `painting` *in the KB, while we insist it is necessary.*

Considering Case A.1 and Case A.2 we get the baseline. When calculating typicality, we should consider the both cases of c_l, thus combining Case A and B through a linear combination.

Finally $P(c_h|e)$ is calculated using the following equation:

$$P(c_h|e) = \alpha\hat{P}(c_h|e) + (1-\alpha)\sum_{c_l^*\in C_l}[P(c_h|c_l^*)] \times P(c_l^*|e) \tag{9}$$

We consider only 2 layers of isA relationship for 2 reasons. The first one is that more layers will lead to noisy concepts such as `issue, factor, element`, which are concepts for almost eveything, Secondly, discussing the transitive relation between concepts is beyond the scope of this paper.

5. FIND ALIAS FOR ATTRIBUTES

This section is devoted to calculating $JD(c_1, c_2)$ of $P((c_1, c_2)|(e_1, e_2))$ in Eq. 5 and $P((c_1, c_2)|a)$ of $P(a|(c_1, c_2))$ in Eq. 2.

We first specify the relationship between them. Actually, they have the following relationship:

$$JD(c_1, c_2) = \sum P((c_1, c_2)|a^*) \times P(a^*) \tag{10}$$

problem here!!! the numerator and the denominator multiplies and get 1, is the relationship here correct??

Definition 1 (Definition of $JD(c_1, c_2)$).

$$JD(c_1, c_2) = \frac{1}{Z_\theta} \times e^{-\frac{1}{f(c_1)\times f(c_2)\times g(c_1,c_2))}}$$

, where $f(c_i) = \frac{\sum P(c_i|e^*)}{,}(i = 1, 2)$ *denotes the importance of a concept itself,* $g(c_{h1}, c_{h2}) = \sum_{(e_{1i},a,e_{2j})\in KB} P(c_{h1}|e_{1i}) \times P(c_{h2}|e_{2j})$ *denotes the joint ratio of the concepts.*

5.1 Calculating $P((c_{h1}, c_{h2})|a)$

Example 7 (Calculating $P((c_{h1}, c_{h2})|a)$). *As illustrated in Fig. 2, the process of calculating* $P((c_{h1}, c_{h2})|a)$ *is as follows*

insert a graph Given a set of concept pairs (c_{h1},c_{h2}), where $c_{h1} \in C_1$ and $c_{h2} \in C_2$, we want to find a set of attributes A, where for each $a \in A$: we can form a (c_{h1},a,c_{h2}) pair which best describe the relationship between c_{h1} and c_{h2}.

For any $(entity_1, attribute, entity_2)$ tuple, later denoted as (e_1, a, e_2) , where e_1 and e_2 are also referred to as `domain` and `range` of the attribute. We can conceptualize e_1 and e_2 using the method in section 4, and get a set of concept

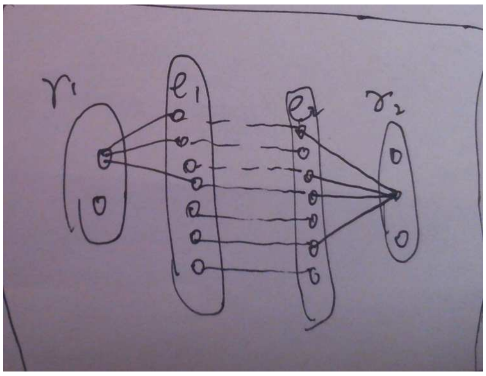

Figure 2: Calculating $P((c_{h1}, c_{h2})|a$ for

C_1, C_2, accompanied with a set of probabilities $P(c_{h1}|e_{1i})$, $P(c_{h2}|e_2)$,where $c_{h1} \in C_1, c_{h2} \in C_2$.

To construct the Entity Attribute Graph, we only need topK concepts to form (c_{h1}, c_{h2}) pair, K trough case study is around 5, so we here set K=10.

Thus for any attribute a, given a pair of entity (e_{1i}, e_{2j}), we can define: should i use joint ratio here?

$$\begin{aligned} P_{(e_{1i},e_{2j})}((c_{h1}, c_{h2})|a) &= P_{before}(c_{h1}|a) \times P_{after}(c_{h2}|a) \\ &= P(c_{h1}|e_{1i})P(e_{1i}|a) \times P(c_{h2}|e_{2j})P(e_{2j}|a) \end{aligned} \tag{11}$$

where we use $P_{(e_{1i},e_{2j})}((c_{h1}, c_{h2})|a)$ to denote observing a single pair (e_{1i}, e_{2j}), how likely is a combination of (c_{h1}, a, c_{h2}) to occur.

Consequently,

$$P((c_{h1}, c_{h2})|a) = \sum_{e_{1i}\in E_1, e_{2j}\in E_2} P_{(e_{1i},e_{2j})}((c_{h1}, c_{h2})|a) \tag{12}$$

where E_1, E_2 denoting the whole set of domain entity and range entity,The $P(e_{1i}|a)$ and $P(e_{2j}|a)$ here has only 2 values 1 and 0, depending on whether e_1 occurs before a or e_2 occurs after a. Apparently, only (e_{1i}, a, e_{2j}) occurs will give the equation a non-zero value, therefore, Eq. 12 is finally equal to Eq. 13.

$$\begin{aligned} P((c_{h1}, c_{h2})|a) &= \sum_{(e_{1i},a,e_{2j})\in KB} P_{(e_{1i},e_{2j})}((c_{h1}, c_{h2})|a) \\ &= \sum_{(e_{1i},a,e_{2j})\in KB} P(c_{h1}|e_{1i}) \times P(c_{h2}|e_{2j}) \end{aligned} \tag{13}$$

The process of calculating is demonstrated in Example. 7

To Construct the Entity Attribute Graph, we calculate $P((c_{h1}, c_{h2})|a)$ for each attribute.

Note that we only consider the attributes whose range is an entity, and ignore those numerical values or date-and-time values such as $(MonaLisa, Year, 1503)$.

For each (c_{h1}, a, c_{h2}) tuple, we can calculate $P((c_{h1}, c_{h2})|a)$ for each

5.2 Find the best alias

We then Use an arg max model use KL divergence? to minimize D_{KL} to solve the problem.

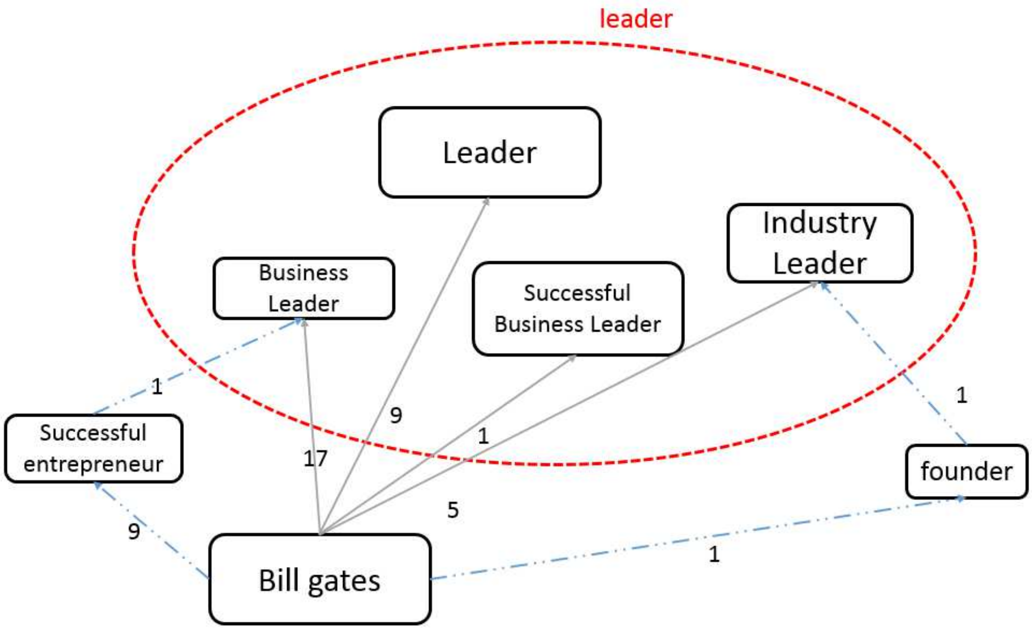

Figure 1: calculating $P(c_h|\texttt{BillGates})$

Given (e_1, e_2), our goal is to find the best attribute for it. We denote it as:

$$\arg\max P((e_1, e_2)|a)$$

where

$$P((e_1, e_2)|a) =$$

6. EXPERIMENT

6.1 Experiment Setup

We

6.2 Evaluation

In

6.3 Head Concept Vs Original Concept

6.4 Find alias

6.4.1 compare

Compare $P((\gamma_{1i}, \gamma_{2i}|a))P(\gamma_{1i}|a) \times P(\gamma_{2i}|a)$

6.4.2 Sense Disambiguation

We can solve the problem of sense disambiguation problem well by applying this method since there are many entities belongs to the same concept and we only consider topK (γ_1, γ_2) pairs that has high typicality $P((\gamma_1, \gamma_2)|a)$, so that the weird (γ_1, γ_2) patterns as manifest in Example. 8 can be easily filtered.

cut the figure smaller

EXAMPLE 8 (SENSE DISAMBIGUATION). *Consider the following* $(e1, a, e2)$ *tuple* `(iphone, manufacturer, apple)`. *Suppose it is our query, where* `apple`*'s sense can either be a kind of* `fruit` *or a* `company`. *Fig. 3 is a heatmap for all the concepts pairs* (γ_1, γ_2) *of attributes* `manufacturer`. *The horizontal axis represents the* e_1 *and the vertical axis stands for* e_2. *The darker the blue is, the higher typicality it will be. In Fig. 3, We can observe that the top concepts of* e_2 *in the heatmap are* `company, manufacturer,...` *and top 10 pairs also does not include* `fruit`. *The intuition for this is that there exists thousands of* $(e1, a, e2)$ *tuple such as* `(BMW_Z4,manufacturer,BMW),(PlayStation_4,manufacturer,Sony)`

Figure 3: (γ_1, γ_2) plot for attribute `Manufacturer`

Table 1: Rerank comparation

Entity	aggregated Top 10			original head concepts		
	agg head	agg count	agg prob	org head	org count	org prob
shanghai	city	1311	0.829222	city	644	0.407337
	region	46	0.029096	region	27	0.017078
	area	42	0.026565	metropolis	23	0.014548
	metropolis	26	0.016445	megacities	15	0.009488
	port	20	0.01265	market	15	0.009488
	market	19	0.012018	location	15	0.009488
	centre	18	0.011385	port	9	0.005693
	location	17	0.010753	locality	6	0.003795
	megacities	15	0.009488	locale	5	0.003163
	center	11	0.006958	seaport	4	0.00253
bill gates	leader	46	0.140244	billionaire	37	0.112805
	billionaire	44	0.134146	entrepreneur	28	0.085366
	entrepreneur	41	0.125	philanthropist	23	0.070122
	philanthropist	30	0.091463	celebrity	15	0.045732
	celebrity	20	0.060976	leader	9	0.027439
	person	16	0.04878	innovator	6	0.018293
	figure	11	0.033537	personality	5	0.015244
	innovator	8	0.02439	expert	5	0.015244
	luminary	8	0.02439	folks	4	0.012195
	individual	7	0.021341	icon	4	0.012195
samsung	company	1030	0.376875	company	816	0.298573
	brand	829	0.30333	brand	561	0.205269
	manufacturer	238	0.087084	client	42	0.015368
	maker	112	0.040981	firm	39	0.01427
	player	60	0.021954	rival	38	0.013904
	phone	60	0.021954	player	33	0.012075
	giant	51	0.018661	phone	30	0.010977
	firm	49	0.017929	conglomerate	19	0.006952
	name	49	0.017929	corporation	19	0.006952
	conglomerate	42	0.015368	partner	12	0.004391
mona lisa	painting	56	0.4	painting	33	0.235714
	masterpiece	21	0.15	masterpiece	16	0.114286
	work	20	0.142857	work	10	0.071429
	film	6	0.042857	film	5	0.035714
	image	5	0.035714	image	3	0.021429
	artwork	4	0.028571	picture	3	0.021429
	portrait	4	0.028571	treasure	2	0.014286
	piece	4	0.028571	song	2	0.014286
	picture	3	0.021429	icon	2	0.014286
	figure	3	0.021429	artwork	1	0.007143

other than `(iphone, manufacturer, apple)` *tuple, which results in a reasonable distribution.*

7. CONCLUSION

8. REFERENCES

[1] N. Dalvi, R. Kumar, and M. Soliman. Automatic wrappers for large scale web extraction. *Proceedings of the VLDB Endowment*, 4(4):219–230, 2011.

[2] I. Dinur and S. Safra. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pages 439–485, 2005.

[3] L. Fang, A. D. Sarma, C. Yu, and P. Bohannon. Rex: explaining relationships between entity pairs. *Proceedings of the VLDB Endowment*, 5(3):241–252, 2011.

[4] M. Gupte, P. Shankar, J. Li, S. Muthukrishnan, and L. Iftode. Finding hierarchy in directed online social networks. In *Proceedings of the 20th international conference on World wide web*, pages 557–566. ACM, 2011.

[5] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM, 2011.

[6] T. Hasegawa, S. Sekine, and R. Grishman. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 415. Association for Computational Linguistics, 2004.

[7] N. Konstantinova. Review of relation extraction methods: What is new out there? In *Analysis of Images, Social Networks and Texts*, pages 15–28. Springer, 2014.

[8] G. Luo, C. Tang, and Y.-l. Tian. Answering relationship queries on the web. In *Proceedings of the 16th international conference on World Wide Web*, pages 561–570. ACM, 2007.

[9] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM, 2007.

[10] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM, 2008.

[11] N. Nakashole, G. Weikum, and F. Suchanek. Discovering and exploring relations on the web. *Proceedings of the VLDB Endowment*, 5(12):1982–1985, 2012.

[12] N. Nakashole, G. Weikum, and F. Suchanek. Discovering semantic relations from the web and organizing them with patty. *ACM SIGMOD Record*, 42(2):29–34, 2013.

[13] D. Shahaf and C. Guestrin. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632. ACM, 2010.

[14] N. Tatti. Faster way to agony. In *Machine Learning and Knowledge Discovery in Databases*, pages 163–178. Springer, 2014.