# An Inference Approach to Basic Level of Categorization

Zhongyuan Wang [♮,†,1] Haixun Wang [‡,2] Ji-Rong Wen [♮,3] Yanghua Xiao [#,4]

[♮]Renmin University of China, Beijing, China
[†]Microsoft Research, Beijing, China
[‡]Facebook, Menlo Park, CA, USA
[#]Fudan University, Shanghai, China

[1]wzhy@outlook.com [2]haixun@gmail.com [3]jirong.wen@gmail.com [4]shawyh@fudan.edu.cn

## ABSTRACT

Humans understand the world by classifying objects into an appropriate level of categories. This process is often automatic and subconscious. Psychologists and linguists call it as *Basic-level Categorization (BLC)*. BLC can benefit lots of applications such as knowledge panel, advertising and recommendation. However, how to quantify basic-level concepts is still an open problem. Recently, much work focuses on constructing knowledge bases or semantic networks from web scale text corpora, which makes it possible for the first time to analyze computational approaches for deriving BLC. In this paper, we introduce a method based on *typicality* and *PMI* for BLC. We compare it with a few existing measures such as *NPMI* and *commute time* to understand its essence, and conduct extensive experiments to show the effectiveness of our approach. We also give a real application example to show how BLC can help sponsored search.

## Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representations (procedural and rule-based), Semantic networks*

## General Terms

Theory

## Keywords

Semantic Network; Conceptualization; Basic Level of Categorization

## 1. INTRODUCTION

Humans understand the world by classifying objects into concepts. The concepts an object belongs to form a set of hierarchically organized categories, ranging from extremely general to extremely specific [23]. Furthermore, different levels of categorization reflect different levels of abstraction, which associate with different properties.


*CIKM'15,* October 19–23, 2015, Melbourne, Australia.

ACM 978-1-4503-3794-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806533 .

### 1.1 Basic-level category

A human being usually maps an object to an appropriate level of category, and regard the object as equivalent to other objects in the category. For example, one would say `I have a house, a car, and a dog` instead of `I have asset, vehicle, and mammal.` Similarly, when someone sees an `iPhone 6`, he most likely thinks of *high end smartphone* or *Apple's product*, instead of *item* or *popular cellular wireless network phone*. For humans, this process of categorization is often automatic and subconscious, and psychologists call the process *Basic-level Categorization (a.k.a. Basic-level Conceptualization, or **BLC** for short).*

BLC is important because it provides rich information with little cognitive efforts [23]. When a person obtains the basic-level category of an unfamiliar object, he will associate the object with the known properties of the basic category, and he suddenly understands almost everything about the object. The fact that he can do this with *little cognitive efforts*[1] has intrigued and pushed researchers to better understand BLC. One of the most important properties of BLC was obtained by Rosch et al. [24]. We summarize their findings in Table 1, which shows that, at the basic level, *perceived similarity among category members is maximized and perceived similarities across contrasting categories is minimized.* As a result, the basic-level category of an object is usually in the middle of the object's taxonomic hierarchies.

| Category Level | Informative? | Distinctive? |
|---|---|---|
| Superordinate | No | Yes |
| Basic-level | Yes | Yes |
| Subordinate | Yes | No |

**Table 1: Differences among category levels**

As an example, consider the term `Microsoft`, which can be categorized into a large number of concepts, for example, *company*, *large company*, *Redmond IT giant*, etc. Let us take a closer look at the following three concepts:

1. *company*
2. *software company*
3. *largest OS vendor*

Both 1 and 3 are highly related to `Microsoft` in the sense that when we think of `Microsoft`, we think of it as a

[1]It is shown that three-year-olds already master basic-level categorization perfectly [16].

*company*, and when we think of *largest OS vendor*, we think of `Microsoft`. However, neither of them is an appropriate basic-level concept for `Microsoft`. To see this, assume we want to find objects that are similar to `Microsoft`. If we go through *company*, we may find objects such as `McDonald's` and `ExxonMobil`, which have not much similarity to `Microsoft`. If we go through *largest OS vendor*, we may not be able to find any reasonable object other than `Microsoft`. On the other hand, if we go through *software company*, we may find `Oracle`, `Adobe`, `IBM`, which are a lot more similar to `Microsoft`. Thus, *software company* is a more appropriate basic-level concept for `Microsoft`, or in other words, properties associated with *software company* are more readily applied to `Microsoft`, which is also the reason why through *software company* we can find many objects that are similar to `Microsoft`.

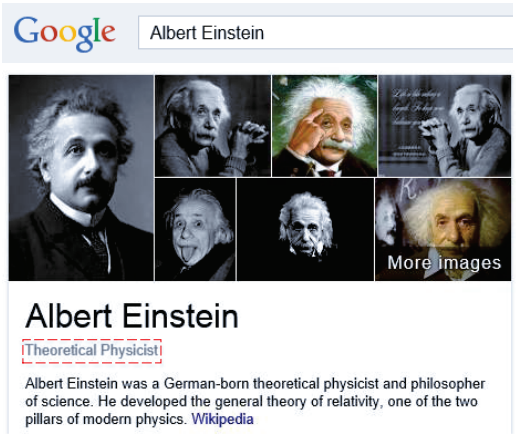
Unfortunately, although much work has been done on this topic, we still do not have a clean formula to infer the basic-level category for any given object. In other words, we do not know how to pick *software company* from thousands of concepts that `Microsoft` belongs to. Psychologists have used word association tests on human subjects to infer concepts that may correspond to the basic-level categories [24]. But such approaches do not scale. On the other hand, an increasing number of applications in the field of information retrieval, natural language understanding, and artificial intelligence require a computational approach for BLC. Short text conceptualization [25, 15, 14, 30] in one of the fundamental techniques. It maps a short text to the concept space, and can benefit web table understanding [28], query intent detection [29], query recommendation [27], etc. BLC is the basic unit of conceptualization, and is crucial for short text conceptualization.

## 1.2 Applications

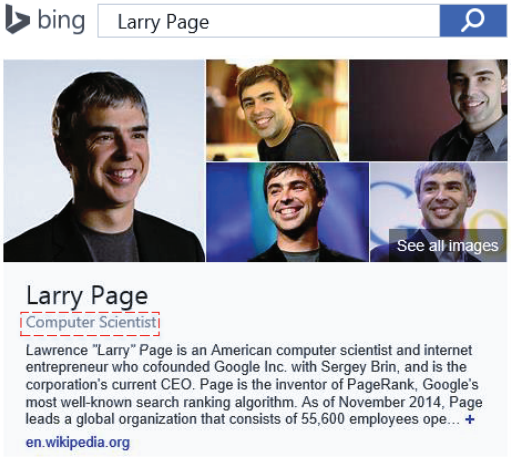
For many applications, including query understanding and ads matching, finding an object's basic-level category is important. Here, we give two real life applications that may serve to demonstrate the need of BLC.

- **Knowledge panel**: Search engines display a *knowledge panel* for queries that contain well-known entities. Fig. 1(a) shows the content of Google's knowledge panel when people search for `Albert Einstein`. Similarly, Bing also provides knowledge panel (As Fig. 1(b) shows). It is worth noting that `Albert Einstein` is labeled as a *theoretical physicist*, which, compared to *world famous German-born American physicist* or just *physicist*, is arguably the "just-right" concept to describe `Albert Einstein`. Unfortunately, there is no good inferencing mechanism to derive such "just-right" concepts, and as a result such labels are currently hand-created by human editors.

- **Advertising and recommendation**: Recommending entities similar to a given entity is important in many applications. As an example, for users who show interest in `Samsung Galaxy S5`, we may want to recommend `HTC One M8` or `iPhone 6`, rather than `Samsung LED TV`. As we know, an entity's basic-level category usually contains many entities that are similar to the entity. In the above example, entities in the category of *popular smartphone* or *high end smartphone* may be good candidates. Thus, knowing the basic-level category enables us to provide good recommendations. Currently, good recommendations are usually based on signals from click logs, but click logs are not always available, especially for new entities.




(a) Google's knowledge panel




(b) Bing's knowledge panel

**Figure 1: Examples of the Application for BLC**

## 1.3 Computational approaches for BLC

The rise of applications such as those mentioned above is pushing for computational approaches for BLC. It has become clear that a better understanding of the human cognitive process helps us build machines that understand the human world. In recent years, a large variety of knowledge bases [3, 26, 9, 32, 5] have been constructed for text understanding and a variety of other tasks. Some of these knowledge bases define a rich concept space, and provides a mapping from a term to the concept space. However, there is no mechanism to determine the basic-level concept for the term, which makes it difficult for machines to "understand" the term, which in turn hampers the impact of such knowledge bases in natural language understanding and other applications.

In this paper, we focus on computational approaches for deriving BLC. Based on previous studies [24], an object's

basic-level concept is considered to be in the middle of the taxonomic hierarchies. In other words, basic-level concepts are a trade-off between general concepts and specific concepts. It is also a trade-off between the accuracy of classification and the power of prediction. Based on this observation, we introduce a method based on *typicality* and *PMI* for BLC. We compare our approach with a few existing measures including graph commute time to understand its essence. We also conduct extensive experiments to show the effectiveness of our approach.

## 1.4 Paper Organization

The rest of the paper is organized as follows. Section 2 introduces background knowledge. Section 3 focuses on computational approaches of BLC. We discuss typicality, PMI, and introduce our own approach. Section 4 compares our approach with existing approaches. We conclude in Section 5.

# 2. KNOWLEDGE BASES

A large variety of knowledge bases, including lexcial knowledge bases and encyclopedic knowledge bases, have been constructed for various applications. Some of them, such as WordNet [10], Wikipedia [31], Cyc [18], and Freebase [3], are created by human experts or community efforts. Others, such as KnowItAll [9], NELL [5], and Probase [32], are created by data-driven approaches. Because information in data-driven knowledge bases is usage based, it is particularly useful for natural language understanding. More specifically, data-driven-based knowledge bases are special in the following aspects:

1. Data-driven knowledge bases contain more fine-grained concepts than human-crafted ones. For example, Freebase has thousands of human crafted concepts, while Probase has millions of concepts (shown in Fig. 2).

2. Information in data-driven knowledge bases is not black or white, but is associated with various weights and probabilities such as typicality (i.e., how typical is an instance for a concept), etc.

In this paper, we are concerned with basic-level categorization. Many basic-level concepts, such as *high-end smartphone*, *software company*, and *theoretical physicist*, are fine-grained concepts that are not present in manual-crafted knowledge bases. Furthermore, inferring basic-level concepts requires the statistical information associated with the knowledge. Therefore, we choose data-driven knowledge bases for inferring basic-level concepts.

We use Probase[2] [32] to provide us fine-grained concepts and their statistics, but our techniques can be applied to other knowledge bases which meet above requirements, such as KnowItAll and NELL. Probase is acquired from 1.68 billion web pages. It extracts *isa* relations from sentences matching Hearst patterns [12]. For example, from the sentence `... presidents such as Obama ...`, it extracts evidence for the claim that `Obama` is an instance of the concept *president*. The core version of Probase contains 3,024,814 unique concepts, 6,768,623 unique instances, and 29,625,920 isA relations.

[2]Probase data is publicly available at http://probase.msra.cn/dataset.aspx

| Term | Concept |
|---|---|
| apple | fruit |
| apple | company |
| apple | food |
| apple | movie |
| apple | guitarist |
| apple | music track |
| apple | book |

**Table 2: The Term `Apple` and Its Concepts**

# 3. BASIC LEVEL OF CATEGORIZATION

In this section, we describe computational approaches for BLC. We start with two popular measures, namely typicality and PMI, and we show why they are insufficient to derive the basic-level category. Based on the insights from these measures, we introduce our approach, and also the intuition why our approach is a better mechanism for BLC.

## 3.1 Preliminary

Scoring is very useful for machines when they leverage knowledge bases. It can make machines do reasoning like human beings. Without scores, knowledge facts in the knowledge base or semantic network are not easy to use. E.g. for the term `apple`, it may belongs to lots of categories, such as *fruit, company, book, movie,* and *music track*. For human beings, probably they will think of *fruit* or *company* when they see `apple`. For machines, if they do not have scores, they will treat *music track* as important as *fruit* or *company*. This makes machines cannot reason over the knowledge base as human beings. What is worse, according to our observations, all kinds of terms or phrases may be names of *book*, *movie*, or *music track*, etc. Without scores, machines will think all texts are related, because they all belong to these concepts. This leads to many errors when machines understand texts.

## 3.2 Typicality

Typicality is an important measure for understanding the relationship between an object and its concept. Each category (concept) may contain one or more typical items, which are ideal or average examples that people extract from seeing real examples [23]. For example, given a concept *bird*, people are more likely to think of `robin` than `penguin`, although `penguin` is also a *bird*. Similarly, when someone mentions `Barack Obama` without a context, we are more likely to think of the concept of *president* than *author*, although `Barack Obama` is actually a best-selling *author* of several books.

More specifically, we use $P(e|c)$ to denote the typical grade of an instance $e$ in concept $c$, and we use $P(c|e)$ to denote the typical grade of a concept $c$ for instance $e$. We call $P(e|c)$ and $P(c|e)$ typicality. For our previous examples, we have $P(\texttt{robin}|bird) > P(\texttt{penguin}|bird)$ and $P(president|\texttt{Obama}) > P(author|\texttt{Obama})$.

How to compute typicality is an interesting and challenging problem. Mervis et al. [21] found that frequency on its own does not predict typicality. For example, `chicken` is a *bird*, and it appears frequently in day-to-day conversations or in texts. But it is much less typical than some much less frequently encountered or discussed birds, such as `robin`. On the other hand, how often an item is thought of as being a member of a category can be used to measure its typi-

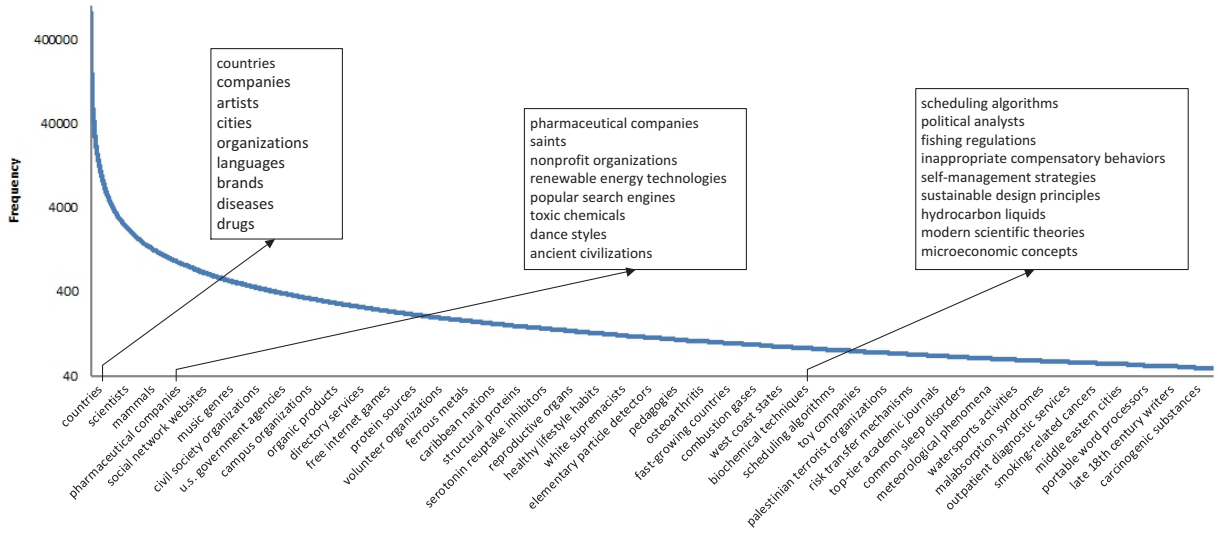



**Figure 2: Frequency distribution of the 3 million concepts**

cality [1]. For example, by leveraging Hearst patterns [12], we find `chicken` is used as an example of *bird* 130 times in a corpus, while `robin` is used 279 times. This result is consistent with the fact that `robin` is more typical than `chicken` as a *bird*. Based on the above observation, in our work, we derive typicality from co-occurrences of concept and instance pairs:

$$P(e|c) = \frac{n(c,e)}{\sum_{e_i \in c} n(c,e_i)}$$
$$P(c|e) = \frac{n(c,e)}{\sum_{e \in c_i} n(c_i,e)} \quad (1)$$

where $n(c,e)$ is the co-occurrence of concept $c$ and instance $e$ in Hearst patterns' sentences from Web documents.

## 3.3 Using Typicality for BLC

On the first look, typicality can be used directly for BLC: For a given instance $e$, how likely is the concept $c$ that maximizes $P(c|e)$ be the basic-level concept for $e$? Or, how likely is the concept $c$ that maximizes $P(e|c)$?

In this section, we argue that neither is a good candidate for the basic-level concept. Let us go back to the `Microsoft` example we mentioned in Section 1. It is easy to see that *company* is a very typical concept for `Microsoft`, that is, when we think of `Microsoft`, we think of it as a *company*. On the other hand, `Microsoft` is a very typical instance of the concept *largest OS vendor*, that is, when we talk about *largest OS vendor*, we are likely talking about `Microsoft`. In other words, $c_1$ = *company* has very large typicality $P(c_1|e)$ and $c_3$ = *largest OS vendor* has very large typicality $P(e|c_3)$, where $e$ = `Microsoft`. But neither is an appropriate basic-level concept for `Microsoft`, for the reasons we described in Section 1. In fact, the two concepts represent two extremes: *company* is a very general concept for *Microsoft*, while *largest OS vendor* is a very specific concept. These two extremes have the following characteristics:

- For the purpose of classifying an instance into the right category, general concepts tend to maximize the accuracy. For example, if we classify instances to general concepts such as *item* or *object*, it may always be correct.
- For the purpose of making prediction, specific concepts tend to have greater power. For example, *largest OS vendor* is able to predict more about *Microsoft* (its properties) with a higher confidence than *company*.

In other words, general concepts may be correct answers to a given instance, but they cannot distinguish different kinds of instances. On the other hand, specific concepts preserve more useful information about instances, but their coverage is limited. What we look for is a compromise of the two.

## 3.4 Using Typicality with Smoothing for BLC

The typicality score we discussed above tends to give higher score to "extreme" concepts, that is, concepts either very general or very specific. The reason is obvious: (1) when $e$ is given, $P(c|e)$ is proportional to the co-occurrence of $c$ and $e$, so it tends to map $e$ to some general concepts; (2) when $e$ is given, $P(e|c)$ tends to return those specific concepts that only contain $e$.

Usually, to address the problem of "extreme" values, we can use the technique of smoothing. We define *smoothed typicality* as follows:

$$P(e|c) = \frac{n(c,e) + \varepsilon}{\sum_{e_i \in c} n(c,e_i) + \varepsilon N_{instance}} \quad (2)$$

where $N_{instance}$ is the total number of instances, and $\varepsilon$ is a small constant which assumes every (concept, instance) pair has a very small co-occurrence probability in the real world, no matter whether we observe it or not.

Smoothed typicality reduces extreme values. For example, assume originally we have $P($`Microsoft`$|$*largest OS vendor*$)$ = 1. In other words, `Microsoft` is the only instance in the category of *largest OS vendor*. Assume $\varepsilon = 0.001$ and $N_{instance} = 1M$, we derive smoothed typicality according to Eq 2 as $P(e|c) = (1 + 0.001)/(1 + 0.001 * 10^6) = 0.001$, which is much smaller than the original value. This translates to boosting the typicality of other concepts such as

*software company*, which makes it possible for us to choose a more appropriate basic-level category.

However, the smoothing approach has two problems. First, the results are very sensitive to $\varepsilon$, which is difficult to tune. Second and more importantly, smoothing helps avoid extreme values using a rationale totally inconsistent with ours. For example, it assumes that there are many *largest OS vendors* for which we do not observe, and as a result it discounts $P(\texttt{Microsoft}|\textit{largest OS vendor})$ to 0.001. But the fact is, `Microsoft` is indeed one of the very few *largest OS vendor*, and reducing its typicality to 0.001 does not make sense. Thus, smoothing does not address the fundamental challenges in finding the basic-level category.

## 3.5 Using PMI for BLC

Pointwise Mutual Information ($PMI$) [20] is a common measure of the strength of association between two terms. We may consider using the $PMI$ between concept $c$ and instance $e$ to find the basic-level concepts, that is, we consider $c = \arg\max_c PMI(e, c)$ as $e$'s basic-level concept, where $PMI(e, c)$ is defined as

$$PMI(e, c) = \log \frac{P(e, c)}{P(e)P(c)} \quad (3)$$

This leads to

$$PMI(e, c) = \log \frac{P(e|c)P(c)}{P(e)P(c)} = \log P(e|c) - \log P(e) \quad (4)$$

For a given $e$, $\log P(e)$ is a constant. Then, ranking concepts by $PMI$ is equivalent to ranking by typicality $P(e|c)$. In other words, PMI is equivalent to typicality. But as we have already shown in Section 3.3, using typicality to find basic-level concept does not work.

In order to make $PMI$ less sensitive to occurrence frequency and at the same time more easily interpretable, Bouma et al [4] proposes a normalized pointwise mutual information ($NPMI$). The $NPMI$ between concept $c$ and instance $e$ is defined as:

$$NPMI(e, c) = \frac{PMI(e, c)}{-\log P(e, c)} = \frac{\log P(e|c) - \log P(e)}{-\log P(e, c)} \quad (5)$$

As we can see from Eq 5, similar to $PMI$ and typicality, $NPMI$ also tends to produce concepts either too general or too specific. More specifically,

- when $P(e, c)$ is large (close to 1), $P(e, c)$ dominates $NPMI$ because $-logP(e, c)$ tends to go to 0. This leads to top basic-level concepts too general.
- when $P(e, c)$ is small, $\frac{1}{-\log P(e,c)}$ does not change much when $P(e, c)$ changes, thus, $PMI$ dominates $NPMI$ in this case. As mentioned above, this leads to top basic-level concepts too specific.

## 3.6 Using $Rep(e, c)$ for BLC

Neither typicality nor $PMI$ produces basic-level concepts. Based on previous studies [24], we know basic-level concepts are concepts neither too general nor too specific. To this end, we make an intuitive compromise and define a scoring function $Rep$ as follows:

$$Rep(e, c) = P(c|e) \cdot P(e|c) \quad (6)$$

For an instance $e$, we use the above score to find its basic-level concept:

$$blc(e) = \arg\max_c Rep(e, c) \quad (7)$$

Intuitively, our measure tries to boost such a concept $c$ for a given instance $e$: i) concept $c$ is $e$'s typical concept, and ii) instance $e$ is also $c$'s typical instance.

Moreover, if we take the logarithm of our scoring function in Eq 6, we get

$$\log Rep(e, c) = \log \frac{P(e, c)^2}{P(e)P(c)} = PMI(e, c) + \log P(e, c) \quad (8)$$

This in fact corresponds to $PMI^2$, which is a normalized form of $PMI$ in the $PMI^k$ family [7]. $PMI^2$ is proposed in an attempt to investigate how to improve upon $PMI$ by introducing the $P(e, c)$ inside the logarithm. It normalizes the upper bound of $PMI$ [4]. Therefore, $PMI^2$ can reduce the extreme values, and help to boost concepts in the middle of the taxonomy.

One important property of an instance's basic-level category is that the category is more likely to contain its similar instances. In our `Microsoft` example in Section 1, we mentioned that going through too general concepts (such as *company*) or too specific concepts (such as *largest OS vendor*) we cannot find instances similar to `Microsoft`. If instead we go through a likely basic-level category (such as *software company*) we find many similar instances. This inspires a graph traversal approach to finding basic-level categories. In the rest of this section, we show that a graph traversal approach is equivalent to our approach of maximizing the scoring function $Rep$ in Eq 7.

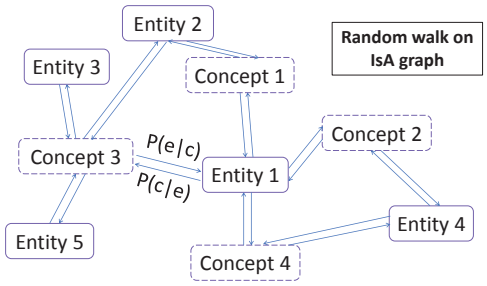



**Figure 3: Random Walk on the IsA Network**

As we mentioned above, given the instance $e$, the basic-level concept $c$ should be one of $e$'s typical concepts, in other words, $c$ should have "shortest distance" with $e$. Similarly, given this basic-level concept $c$, $e$ should have "shortest distance" with $c$. Therefore, for an instance $e$, we consider the process of finding its basic-level concepts as *a process of finding concepts that have shortest expected distance to* $e$. Intuitively, traversing from node $e$, it is very likely we arrive at the concept, and traversing from the concept, it is very likely we come back to $e$. This corresponds to a random walk problem of finding nearest nodes reached by a given node, as shown in Fig. 3. We use commute time [19] as a measure of the distance between two nodes in a graph. Commute time is defined as the expected number of steps that a random

walk starting at node $i$, going through node $j$ once, and returning to $i$ again. The commute time between an instance $e$ and a concept $c$ is:

$$\begin{aligned} Time(e,c) &= \sum_{k=1}^{\infty}(2k) * P_k(e,c) \\ &= \sum_{k=1}^{T}(2k) * P_k(e,c) + \sum_{k=T+1}^{\infty}(2k) * P_k(e,c) \\ &\geq \sum_{k=1}^{T}(2k) * P_k(e,c) \\ &\quad +2(T+1) * (1 - \sum_{k=1}^{T} P_k(e,c)) \end{aligned} \tag{9}$$

where $P_k(e,c)$ is the probability of starting from $e$ to $c$ and back to $e$ in $2k$ steps.

As we are only interested in concepts within a short commute, we may just ignore concepts with commute time larger a threshold of $T$ steps. Let us constrain the random walk within 4 steps, then we have:

$$\begin{aligned} Time'(e,c) &= 2 * P(c|e)P(e|c) + 4 * (1 - P(c|e)P(e|c)) \\ &= 4 - 2 * P(c|e)P(e|c) \\ &= 4 - 2 * Rep(e,c) \end{aligned} \tag{10}$$

This shows that the commute time $Time'(e,c)$ has an inverse relationship with the scoring function $Rep$ we defined in Eq 6. Thus, our simple, easy-to-compute scoring method is equivalent to a graph traversal approach of finding the basic-level category under this random walk assumption. But for the complete commute time, our experiments show that the representativeness score is better than it.

# 4. EXPERIMENTS

To evaluate the effectiveness of scores proposed in this paper, we conduct our experiments on the Probase [32] semantic network. The dataset we use is called "core" version. It contains 3,024,814 unique concepts, 6,768,623 unique instances, and 29,625,920 edges among them.

## 4.1 Experiment Setting

Conceptualization is one of the essential processes for text understanding. It maps instances in the text to the concept space. In this evaluation, we select top queries from Bing search logs between January 1st, 2015 and January 31st, 2015, then we use the typicality $P(c|e)$, $P(e|c)$, and our approach $Rep(e,c)$ to rank concepts corresponding to these instances. Finally, we generate 1,683 *(instance, concept)* pairs as labeling candidates.

We also compare our approaches with several baselines: $MI$, $NPMI$, $PMI^3$. Since $PMI$ is reduced to typicality $P(e|c)$ in our scenario, we do not treat it as another baseline separately. The formulas of these baselines are as follows:

**Mutual Information (MI)**:
$MI(e,c) = \sum P(e,c) \log \frac{P(e,c)}{P(e)P(c)}$

**Normalized Pointwise Mutual Information (NPMI)**: please refer to Equation 5.

**PMI$^3$** [4]: $PMI^3(e,c) = \log \frac{P(e,c)^3}{P(e)P(c)}$

## 4.2 Metrics

We now discuss how we evaluate the results of different approaches. As there is not ground-truth ranking for conceptualization results. We submitted a labeling task to UHRS (Universal Human Rating System), which is a crowdsourcing and human annotation platform. Each *(instance, concept)* pair is labeled by three workers. Therefore we got 5,049 labeled records. We took the majority of the labels as the final label. If all three annotators did not agree with each other, this pair was sent to the fourth annotator to label, and so on. In this labeling task, the workers did not our labeling goal. They just strictly follow the guideline shown in Table 3.

| Label | Meaning | Examples |
|---|---|---|
| Excellent | Good matched concepts at the basic level | (bluetooth, wireless communication protocol) |
| Good | A little general or specific | (bluetooth, accessory) |
| Fair | Too general or specific | (bluetooth, feature) |
| Bad | Non-sense concepts | (bluetooth, issue) |

**Table 3: Labeling Guideline for Conceptualization**

Then we employ $precision@K$ and $nDCG$ to evaluate the results of different approaches. The $precision@K$ is used for evaluating the **correctness** of concepts, and $nDCG$ is used to measure the **ranking** of concepts.

For $precision@K$, we treat *Good/Excellent* as score 1, and *Bad/Fair* as 0. We calculate the precision of top-K concepts as follows:

$$Precision@K = \frac{\sum_{i=1}^{K} rel_i}{K} \tag{11}$$

where $rel_i$ is the score we define above.

For $nDCG$, we treat *Bad* as 0, *Fair* as 1, *Good* as 2, and *Excellent* as 3. Then we calculate $nDCG$ as follows:

$$nDCG_K = \frac{rel_1 + \sum_{i=2}^{K} \frac{rel_i}{\log i}}{ideal\_rel_1 + \sum_{i=2}^{K} \frac{ideal\_rel_i}{\log i}} \tag{12}$$

where $rel_i$ is the relevance score of the result at rank $i$, and $ideal\_rel_i$ is the relevance score at rank $i$ of an ideal list, obtained by sorting all relevant concepts in decreasing order of the relevance score.

## 4.3 NDCG & Precision

We show the comparison from two aspects: without smoothing, and with smoothing. Fig. 4 shows the top-20 results for different scoring approaches.

Without smoothing, as Fig. 4(a) shows, the score $Rep(e,c)$ is much better in precision than others. $Rep(e,c)$ is the best for top-2, top-5, and top-10. $PMI^3$ is also good for top-1, but it is worse for other cases. So it is not stable. For the ranking of top concepts, as Fig. 4(c) shows, $Rep(e,c)$ outperforms all other methods in all cases.

With smoothing, we try different values of $\varepsilon$, from *1e-3* to *1e-7*. We observe that the smoothing technique has a significant effect on the typicality $P(e|c)$, as shown in Fig. 5. For $P(e|c)$, its optimized $\varepsilon$ is *1e-4*. With this smoothing setting, as shown in Fig. 4(b) and Fig. 4(d), the typicality $P(e|c)$ outperforms all other methods in both *precision* and $nDCG$

when K is 2, 3, 5, 10, and 15. However, as we mentioned in Section 3.4, the rationale of smoothed typicality is totally inconsistent with our problem. Its top concepts may be good, but not for the rest of concepts. Besides, it is quite sensitive to $\varepsilon$, which is hard to tune. On the other hand, $Rep(e, c)$ is still quite good, and it wins the *precision*@1 when its smoothing $\varepsilon$=*1e-4*.

Based on this evaluation, we have the following conclusions:

- The score $Rep(e, c)$ performs best for conceptualization task overall, and more importantly, it is robust.
- The smoothed typicality $P(e|c)$ is very sensitive to $\varepsilon$. It may have a good performance for top results after a sophisticated tuning on the smoothing value $\varepsilon$.

## 4.4 Examples

Some examples are shown in Table 4. We find top concepts ranked by $P(c|e)$ tend to be general. E.g. the top 1 concept of `battery` is *item*. Obviously, *item* is not a good concept to predict the instance's features. On the other hand, concepts ranked by $P(e|c)$ (without smoothing) tends to be specific. Though these concepts have greater power on prediction, they have limited coverage. This makes them useless in the computation. E.g. when we compare two related instances with extremely specific concepts, they may have little overlapping information. In Table 4, concepts ranked by $Rep(e, c)$ and $P(e|c)$ with smoothing($\varepsilon$ = *1e-4*) are the best. They are the trade-off between general concepts and specific concepts.

However, as we mentioned in Section 3.4, $\varepsilon$ is difficult to tune, and the rationale of smoothed typicality is inconsistent with the problem we want to resolve in this paper. For example, *largest OS vendor* should not be assumed to contain many other instances for which we do not observe. Therefore, smoothed typicality may only be good for top concepts, instead of all concepts.

## 4.5 Application

With the scores proposed in this paper, the knowledge base becomes usable for machines. In this subsection, we showcase one application leveraging the basic-level concepts. Definitely, more applications can be easily developed for different scenarios.

Sponsored search is one of the most successful business models for search engine companies. It matches user queries to relevant ads. In reality, each ad is associated with a list of keywords. Advertisers bid for keywords, and also specify matching options for these keywords. One option is *exact match* where an ad is displayed only when a user query is identical to one of the bid keywords associated with the ad. Another option is *smart match* which is based on semantic relevance. Exact match targets exact traffic, but it is limited since queries are various. Currently, smart match is the default option provided by mainstream search engines.

In smart match, search engines map the query to bid ad keywords. Since both are short texts, traditional bag-of-words approaches do not work well in this scenario. Therefore, we can leverage the knowledge base for this task in smart match.

For each short text, we first identify instances from the short text (query, or bid keyword), and map it to basic-level concepts with the score $Rep(e, c)$. Then we simply merge the concept vectors of different instances in the short text, and get a single concept vector as the representation of this short text. Finally, we can calculate the semantic similarity between queries and bid keywords by comparing their concept vectors (E.g. using cosine similarity function).

We conduct our experiments on real ads click log of Bing (from January 1st, 2015 to January 31st, 2015). The experiment process is as follows: first, we calculate the semantic similarity score of (query, bid keyword) pairs for each record in the log; Second, we divide all scores into 10 buckets; Third, we aggregate the click number and impression number in the same bucket.

The overall results are illustrated in Fig. 6(a). X-axis represents the bucket number (E.g. *bucket 1* means the similarity score is between 0 and 0.1, and so on). Y-axis is the general click-through rate (CTR) of this bucket, where $CTR = \frac{\#\ of\ clicks}{\#\ of\ impressions}$. From the figure, we observe that: (1) once our semantic similarity score is low, the CTR is low; (2) once our score is high, the CTR is high; (3) the highest CTR is about 3 times of the lowest CTR. This means our semantic similarity score can be a strong feature for the query and ads matching.

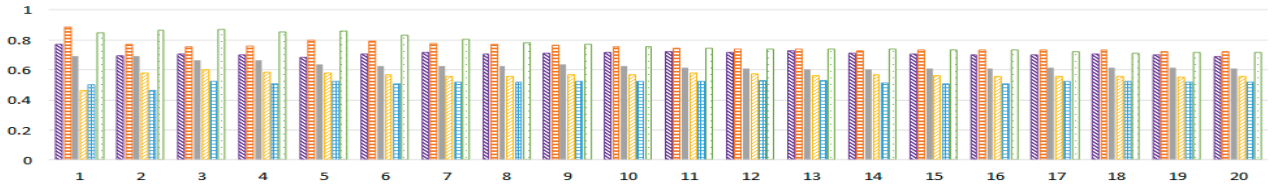
We further analyze our results by query frequencies. We separate all queries to 10 deciles in the order of frequency, and each decile has the same total volume of traffic. Generally speaking, decile 1 to 3 are head queries, decile 4 to 6 are torso queries, and decile 7 to 10 are tail queries. Usually, head queries can be covered by exact match. Therefore we mostly focus on torso queries and tail queries. Results are shown in Fig. 6(b) and Fig. 6(c). For both torso and tail queries, the correlation between our semantic similarity score and CTR is preserved. This is quite good because long queries usually are lack of click signals, and the semantics can fill this gap.

# 5. RELATED WORK

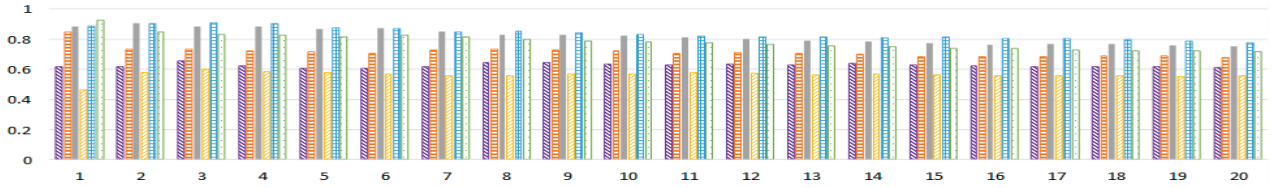
The goal of BLC is for text understanding. Currently, there are two major efforts on semantic analysis and text representation. One is implicit knowledge mining, the other is explicit knowledge mining.

In terms of implicit knowledge mining, some efforts focus on implicit semantic analysis and topic modeling, such as traditional latent semantic analysis (LSA) [8], probabilistic latent semantic indexing (PLSI) [13], and latent Dirichlet allocation (LDA) [2]. They can be also used for basic-level concept reasoning. However, the semantics mined by these approaches cannot be interpreted by human being. Besides, their computation is quite heavy, which makes them cannot be easily applied for large scale data. Other efforts try to get the term embedding by using deep learning techniques [6, 22]. This embedding can be treated as the representation for a given term in high-dimensional semantic space, and benefit lots of applications. But ususally the embedding is not good for those low frequency terms, and it is not easy to tune once its overall quality is bad.

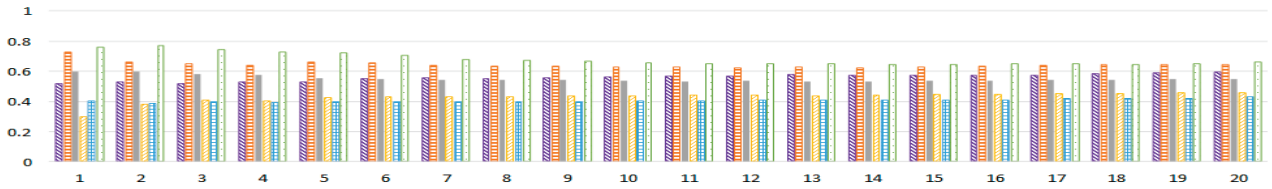
In terms of explicit knowledge mining, explicit semantic analysis (ESA) [11] and conceptualization [25, 15] are the representative work. The former represents the meaning of texts by Wikipedia's article titles. But it is still not the natural way of human thinking, and the "concepts" of a given instance are related articles instead of its categories. The latter is the scope of this paper. But previous work [25, 15] pays attention to short-text level conceptualization, while



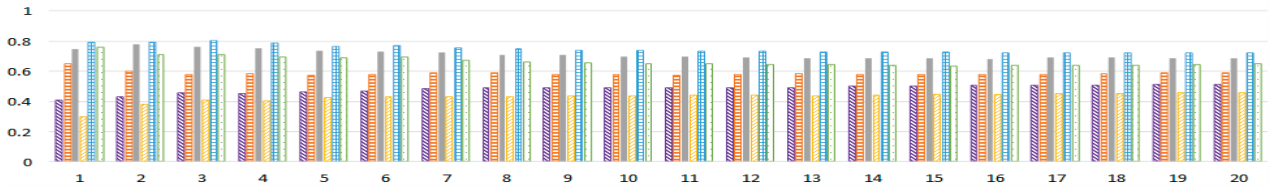
(a) Precision Without Smoothing



(b) Precision With Smoothing ($\varepsilon$ = *1e-4*)



(c) nDCG Without Smoothing



(d) nDCG With Smoothing ($\varepsilon$ = *1e-4*)

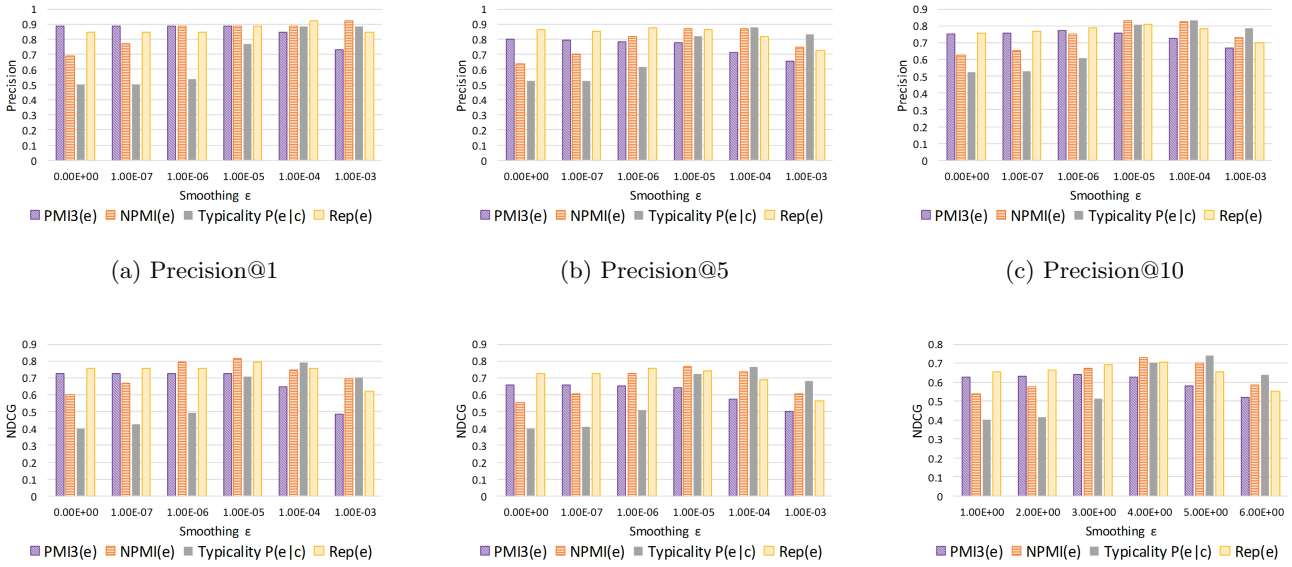

**Figure 4: Precision and nDCG Comparison**

our paper focuses on a more fundamental part: instance-level conceptualization. A good instance-level conceptualization mechanism can further improve short-text level conceptualization.

For scores discussed in this paper, typicality and basic-level conceptualization are actively studied in cognitive science and psychology at first. E.g. psychologist Gregory Murphy's highly acclaimed book [23] discusses the typicality and the basic level of concepts from the perspective of psychologists, which is the basis of these two scores proposed in this paper. Other work [17, 32] proposes typicality scores for some special scenarios. Their scoring functions cannot be easily extended to semantic networks. E.g. Lee et al. [17] leverage instances as intermedia for calculating typicality $P(a|c)$ between an attribute $a$ and a concept $c$. Instead, our typicality is more generic and easy to be adopted by more applications. Besides, we also discuss a smoothing approach and *Rep* score, which have been proved that they are much better than the standard typicality in instance-level conceptualization. Furthermore, we compare our BLC score function with PMI and commute time theoretically, which reveals the essence of BLC score.

## 6. CONCLUSION

In this paper, we discuss computational approaches for *Basic-level conceptualization (BLC)*. We propose our method based on *typicality* and *PMI*. We make deep analysis and compare it with some popular measures to understand its essence. We conduct extensive experiments and show the effectiveness of our approach. We also use a real example to demonstrate how our score helps current ads system.



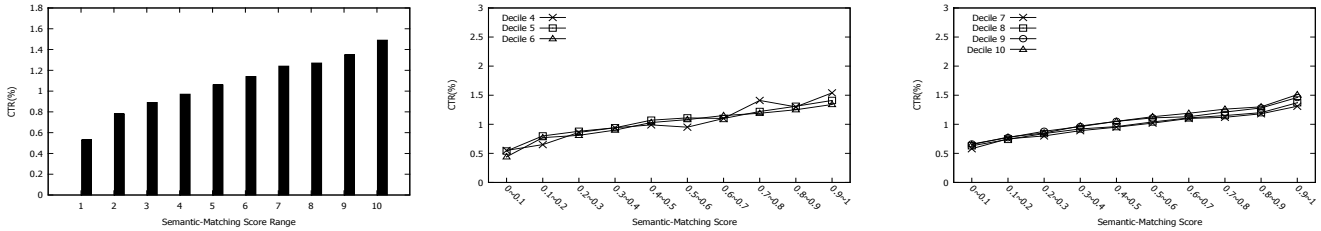

(a) Precision@1 (b) Precision@5 (c) Precision@10

(d) nDCG@1 (e) nDCG@5 (f) nDCG@10

**Figure 5: Precision and nDCG Comparison with Different Smoothing Values**

**Table 4: Examples of Basic-level Concepts by Different Scoring Approaches**

| **Instance** | **Rank by** $P(c\|e)$ | **Rank by** $P(e\|c)$ (same as $PMI$ in our scenario) | **Rank by** $Rep(e,c)$ (our approach) | **Rank by** $P(e\|c)$ **with smoothing** ($\varepsilon$ = *1e-4*) |
|---|---|---|---|---|
| Microsoft | 1. company<br>2. vendor<br>3. corporation | 1. standardized technology industry<br>2. global and leading organization<br>3. large software maker | 1. software company<br>2. technology company<br>3. software vendor | 1. software company<br>2. software vendor<br>3. technology company |
| Nokia | 1. company<br>2. brand<br>3. manufacturer | 1. leading global MNC handset brand<br>2. established technology vendor<br>3. brand-name cell phone product | 1. original mobile phone<br>2. mobile phone manufacturer<br>3. handset maker | 1. mobile phone manufacturer<br>2. handset maker<br>3. handset manufacturer |
| Java | 1. language<br>2. programming language<br>3. technology | 1. built-in language environment<br>2. controllable language<br>3. managed language | 1. object-oriented language<br>2. programming language<br>3. object-oriented programming language | 1. object-oriented language<br>2. programming language<br>3. object oriented programming system |
| Bluetooth | 1. feature<br>2. technology<br>3. wireless technology | 1. connectivity facility<br>2. disable networking capability<br>3. hands-free communication device | 1. wireless technology<br>2. wireless protocol<br>3. connectivity option | 1. wireless technology<br>2. connectivity option<br>3. wireless protocol |
| battery | 1. item<br>2. accessory<br>3. power source | 1. mobile power source<br>2. electrical energy storage system<br>3. auto supply | 1. power source<br>2. power supply<br>3. energy storage device | 1. power source<br>2. power supply<br>3. consumable part |




(a) Different SMS Range (b) Torso Queries (c) Tail Queries

**Figure 6: Correlation between Semantic Matching Score and CTR**

## 7. ACKNOWLEDGMENTS

This work was partially supported by the National Key Basic Research Program (973 Program) of China under grant No. 2014CB340403 & No.2015CB358800, the Natural Science Foundation of China(No.61472085), and the Fundamental Research Funds for the Central Universities & the Research Funds of Renmin University of China.

## 8. REFERENCES

[1] L. W. Barsalou. Ideals, central tendency, and frequency of instantiation as determinants of graded structure in categories. *JEP:LMC*, 11(4):629–654, 1985.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250. ACM, 2008.

[4] G. Bouma. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40, 2009.

[5] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, pages 1306–1313, 2010.

[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

[7] B. Daille. *Approche mixte pour l'extraction de terminologie: statistique lexicale et filtres linguistiques*. PhD thesis, 1994.

[8] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[9] O. Etzioni, M. Cafarella, and D. Downey. Webscale information extraction in knowitall (preliminary results). In *International World Wide Web Conference (WWW)*, 2004.

[10] C. Fellbaum, editor. *WordNet: an electronic lexical database*. MIT Press, 1998.

[11] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, volume 7, pages 1606–1611, 2007.

[12] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545, 1992.

[13] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[14] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou. Short text understanding through lexical-semantic analysis. In *International Conference on Data Engineering (ICDE)*, 2015.

[15] D. Kim, H. Wang, and A. Oh. Context-dependent conceptualization. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence (IJCAI)*, pages 2654–2661. AAAI Press, 2013.

[16] G. Lakoff. *Women, fire, and dangerous things: What categories reveal about the mind*. Cambridge Univ Press, 1990.

[17] T. Lee, Z. Wang, H. Wang, and S.-w. Hwang. Attribute extraction and scoring: A probabilistic approach. In *International Conference on Data Engineering (ICDE)*, 2013.

[18] D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley, 1989.

[19] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46, 1993.

[20] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[21] C. B. Mervis, J. Catlin, and E. Rosch. Relationships among goodness-of-example, category norms, and word frequency. *Bulletin of the Psychonomic Society*, pages 283–284, 1976.

[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*, 2013.

[23] G. L. Murphy. *The big book of concepts*. MIT Press, 2004.

[24] E. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem. Basic objects in natural categories. *Cognitive psychology*, 8(3):382–439, 1976.

[25] Y. Song, H. Wang, Z. Wang, H. Li, and W. Chen. Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence (IJCAI)*, pages 2330–2336. AAAI Press, 2011.

[26] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *International World Wide Web Conference (WWW)*, pages 697–706, 2007.

[27] F. Wang, Z. Wang, Z. Li, and J.-R. Wen. Concept-based short text classification and ranking. In *ACM International Conference on Information and Knowledge Management (CIKM)*, October 2014.

[28] J. Wang, H. Wang, Z. Wang, and K. Zhu. Understanding tables on the web. In *International Conference on Conceptual Modeling*, October 2012.

[29] Z. Wang, H. Wang, and Z. Hu. Head, modifier, and constraint detection in short texts. In *IEEE 30th International Conference on Data Engineering (ICDE)*, pages 280–291. IEEE, 2014.

[30] Z. Wang, K. Zhao, H. Wang, X. Meng, and J.-R. Wen. Query understanding through knowledge-based conceptualization. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[31] Wikipedia. Plagiarism — Wikipedia, the free encyclopedia, 2004. `http://en.wikipedia.org/w/index.php?title=Plagiarism&oldid=5139350`, [Online; accessed 22-July-2004].

[32] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492. ACM, 2012.