Towards a Probabilistic Taxonomy of Many Concepts

Wentao Wu, Hongsong Li, Haixun Wang, Kenny Q. Zhu
Microsoft Research Asia

ABSTRACT

Knowledge is indispensable to understanding. The ongoing information explosion highlights the need to enable machines to better understand electronic text in natural human language. The challenge lies in how to transfer human knowledge to machines. Much work has been devoted to creating universal ontologies for this purpose. However, none of the existing ontologies has the necessary depth and breadth to offer "universal understanding." In this paper, we present a universal, probabilistic ontology that is more comprehensive than any of the existing ontologies. It contains 2.7 million concepts harnessed automatically from a corpus of 1.68 billion web pages and two years' worth of search log data. Unlike traditional knowledge bases that treat knowledge as black and white, it enables probabilistic interpretations of the information it contains. The probabilistic nature then enables it to incorporate heterogeneous information in a natural way. We present details of how the core ontology is constructed, and how it models knowledge's inherent uncertainty, ambiguity, and inconsistency. We also discuss potential applications, e.g., understanding user intent, that can benefit from the taxonomy.

1. INTRODUCTION

Knowledge is indispensable to understanding. Much effort has been devoted to developing ontologies and taxonomies that organize knowledge. Concepts are at the core of such ontologies and taxonomies. The definition of concept varies. In this paper, we use the following definition:

DEFINITION 1. *A concept is an abstract set of things or ideas. An instance is a singleton concept. A taxonomy is a hierachical structure of concepts organized in isA relations*[1].

Why concepts are important? Psychologist Gregory Murphy began his highly acclaimed book with the statement *"Concepts are the glue that holds our mental world together"* [14]. Still, Nature magazine book review calls it an understatement, because *"Without concepts, there would be no mental world in the first place"* [3]. Concepts are important because by being able to place something into its proper place in the taxonomy, one can learn a considerable amount about it [14]. Recently, there has been a surge of interest in *concept-centric* approaches (e.g., [8, 15]) to enhance web text understanding and hence web search. But existing ontologies and taxonomies are still insufficient for this task.

First, existing ontologies and taxonomies have *limited concept space.* Most taxonomies are constructed by a manual process carried out by experts in specific domains. The laborious, time consuming, and costly process limits the scope and the scale of the taxonomies thus built. For example, the Cyc project [13], after 25 years of continuing effort by many knowledge experts, contains about 120,000 concepts. To overcome this bottleneck, some open domain knowledge-bases, e.g., Freebase [4], rely on community efforts to increase the scale. However, while they have near-complete coverage of several specific concepts (e.g. books, music and movies), they lack general coverage of many other concepts. More recently, automatic taxonomy construction approaches, such as KnowItAll [10], TextRunner [2], YAGO [27], and NELL [6], have been in focus, but they still have a small scale and coverage in terms of concept space.

Second, in existing ontologies and taxonomies, knowledge is *black and white.* Most taxonomies believe that a knowledge base should provide standard, well-defined and consistent reusable information, and therefore these concepts and relations included in these taxonomies are kept "religiously" clean. In most of these systems, there is little or no ambiguity. We argue that human language is intrinsically diverse, ambiguous and sometimes inconsistent (e.g. big companies or best universities), and consequently cannot be effectively understood with only black-and-white knowledge.

Third, existing ontologies and taxonomies have *no scoring mechanism.* The ability to rank entities and concepts is essential in the context of web search (consider the query '*best universities around the world*'). However, none of the existing taxonomies come with scoring mechanisms to rank the knowledge they contain. For example, Freebase has complelte list of countries and companies, but it does not know which country is a *typical* developping country or which company is a *typical* IT company.

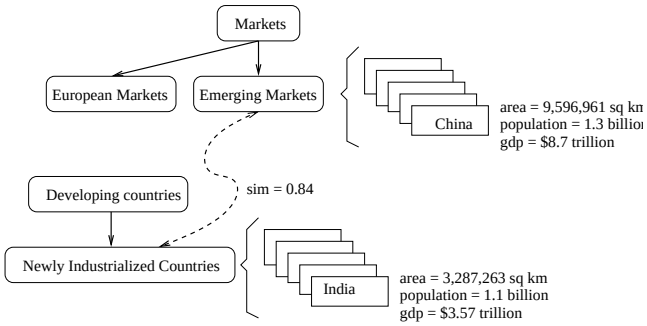


Figure 1: A Snippet of Probase's core taxonomy.

[1]We do not differentiate between the isA and the isInstanceOf relationship for presentation simplicity.

VLDB '11, August 29- September 3, 2011, Seattle, WA

In this paper, we introduce Probase, a universal, general-purpose, probibilistic taxonomy automatically constructed from a corpus of 1.6 billion web pages. Figure 1 is a snippet of Probase, which consists of concepts (e.g. *emerging markets*), instances (e.g., *China*), attributes and values (e.g., *China's population is 1.3 billion*), and relationships (e.g., *emerging markets*, as a concept, is closely related to *newly industrialized countries*), all of which are automatically derived in an unsupervised manner.

The Probase taxonomy is unique in three aspects:

1. It is the largest general-purpose taxonomy fully automatically constructed from HTML text on the web. It has a huge concept space with more than 2.7 million concepts, almost 20 times larger than that of YAGO, which makes Probase the largest taxonomy in terms of concept space (Table 2). Besides popular concepts such as "cities" and "musicians", which are included by almost every general purpose taxonomy, Probase also has millions of specific concepts such as "renewable energy technologies", "meteorological phenomena" and "common sleep disorders", which cannot be found in Freebase, Cyc, or any other taxonomies.

2. It is built by a novel iterative learning algorithm which extracts information that previously cannot be extracted using standard extraction techniques such as the Hearst pattern.

3. It uses probabilistic approaches to score and rank its knowledge. Knowledge is no longer black and white. Each fact or relation is associated with some metric to measure its plausibility or likelihood. Such a probabilistic treatment not only allows Probase to better capture the semantics of human languages, but also makes it easy to integrate with other existing taxonomies.

In this paper, we focus on the construction of Probase's backbone taxonomy which is made up of concepts in noun phrases (e.g., "basic watercolor techniques") and *isA* relationships among the concepts. The principles and techniques discussed here exemplify a *bootstrapping* or *iterative learning* mechanism which we use to obtain knowledge: In each step, we try to make sense of some information using Probase's knowledge, and the information, once understood, is added as new knowledge to Probase, which enables Probase to understand more information in the next step. This mechanism is used to obtain not only the backbone taxonomy, but also attributes and values for concepts and entities in Probase, as well as part-whole and other millions of relationships. More information about Probase, including Probase-enabled applications such as semantic web search, and a small excerpt of the Probase taxonomy, can be found at `http://research.microsoft.com/probase/`. We are currenlty working to make the Probase taxonomy available to the public in the near future.

The remainder of the paper is organized as follows: Section 2 describes an iterative algorithm that extracts *isA* relationship from the web corpus. Section 3 presents taxonomy construction from individual hypernum-hyponym pairs. Sections 4 and F (in Appendix) discuss how we score and enrich the taxonomy (by integrating data from other sources). We show experimental results in Section 5, related work in Section A, and conclude in Section 6.

2. ITERATIVE LEARNING

We harvest useful information from the web to build the taxonomy using an iterative learning approach: In each round of information extraction, we accumulate probabilistic knowledge. We then use this probabilistic knowledge in the next round to help us extract information we missed earlier. We perform this process iteratively until no more information can be extracted.

We use the example of building the backbone of the taxonomy to deomonstrate the power of iterative learning. The backbone of the taxonomy mainly consists of the *isA* relationships between the concepts. The reason we hone in on isA relationships is threefold. First, it is well recognized that isA relationships are the most critical relationships in all taxonomies and ontologies. In fact, all known approaches of automatic taxonomy construction [5, 25, 16, 18, 22] focus on using some linguistic patterns, such as the *Hearst* patterns [12], to extract isA (a.k.a., hypernym-hyponym) pairs from the text. Second, due to the complexities of natural languages, previous approaches based on naive linguistic pattern matching have low precision, which makes these approaches largely experimental. A new approach for large scale, accurate isA relationships extraction is needed. Third, we develop an iterative learning mechanism for isA relationship extraction. The mechanism exemplifies how a probabilistic knowledge base can be used in a bootstrapping manner for incremental knowledge discovery.

2.1 Problem and Challenge

We use Hearst patterns to demonstrate how we extract isA relationships through iterative learning. Table 3 (in Appendix C) lists six Hearst patterns that we are concerned with. Our goal is to extract isA pairs from any sentence that matches any of the six patterns. Specifically, given such a sentence, we want to obtain

$$s = \{(x, y_1), (x, y_2), ..., (x, y_m)\}$$

where x is the superordinate concept, and $\{y_1, \cdots, y_m\}$ are its subordinate concept. For example, from the sentence

"... in tropical countries **such as** *Singapore*, *Malaysia*, ..."[2]

we want to derive s ={(tropical countries, Singapore), (tropical countries, Malaysia)}.

However, natural languages are rife with ambiguities, and as a result, syntactic patterns often deliver confusing results. Consider the following examples:

1) ... animals other than dogs **such as** *cats* ...

2) ... companies **such as** *IBM*, *Nokia*, *Proctor* **and** *Gamble* ...

3) Online presence helps companies **such as** *Amazon*, *Expedia* and *Microsoft notice you*.

4) ... representatives in North America, Europe, the Middle East, *Australia*, *Mexico*, *Brazil*, *Japan*, *China*, **and other** countries.

5) At Berklee, I was playing with cats **such as** *Jeff Berlin*, *Mike Stern*, *Bill Frisell*, **and** *Neil Stubenhaus*.

In order to extract isA pairs correctly from these examples, we must possess the following "knowledge": 1) *cats* are not *dogs*; 2) *Procter and Gamble* is the name of one company not two companies; 3) *Microsoft notice you* is not a company name; 4) *Middle East* is not a country but *Australia* is; 5) the word *cat* has more than one sense. Of course, none of this knowledge is conveyed by the syntax of the above sentences.

2.2 Iterative Extraction of isA Relationships

Given a web document, we break it down into a set of sentences using standard sentence boundary disambiguation methods, and find sentences that match any Hearst pattern. We also perform *deduplication* for long sentences. Our rationale is that each sentence serves as independent evidence for a claim. The probability that two long duplicated sentences are from two independent sources is low (they are usually the result of copy-and-paste).

[2]The underlined term is the superordinate concept, and the italicized terms are its subordinate concepts.

Specifically, we use a bootstrapping process to find isA pairs. Let Γ denote the knowledge we have, and in this case, it is the set of isA pairs that we have discovered. For each $(x, y) \in \Gamma$, we also keep a count $n(x, y)$, which indicates how many times (x, y) is discovered. Initially, Γ is empty. We search for isA pairs in the text, and we use Γ to help identify valid ones among them. Then, we enlarge Γ by adding the newly discovered pairs, which further enhances our power to identify more valid pairs. It is important to understand that having a pair (x, y) in Γ does not mean it is *true* that x and y have an isA relationship. All it means is that we have certain evidence for such a claim, although the evidence itself might be wrong or we might have interpreted the evidence incorrectly.

Algorithm 1 (in Appendix D) outlines our method. It repeatedly scans the set of sentences until no more pairs can be identified. Procedure *SyntacticExtraction* finds candidate superordinate concepts X_s and candidate subordinate concepts Y_s from a sentence s. If more than one candidate superordinate concepts exist, we call procedure *SuperOrdinateConceptDetection* to reduce X_s to a single element. Then, procedure *SubOrdinateConceptDetection* filters out unlikely subordinate concepts in Y_s. Finally, we add newly found isA pairs to the result. Due to the new results, we may be able to identify more pairs, so we scan the sentences again. We describe the three procedures in detail below.

SyntacticExtraction. Procedure *SyntacticExtraction* uses part-of-speech (POS) taggers and shallow/deep language parsers [7] to identify possible isA pairs in a sentence s.

Identifying possible superordinate concepts X_s from a sentence s can be done in a relatively straightforward manner based on the Hearst's pattern in use. For instance, given $s =$ "*... animals other than dogs such as cats ...*", we identify possible superordinate concepts as $X_s =$ {animals, dogs}. Note that we require that every element in X_s must be a noun phrase in plural form [22]. As a result, for sentence $s' =$ "*... countries other than Japan such as USA ...*", the set of possible superordinate concepts $X_{s'}$ contains "countries" but not "Japan."

It is more challenging to identify possible subordinate concepts Y_s from a sentence s. Based on the Hearst's pattern in use, we first extract a list of candidates by using ',' as the delimiter. For the last element, we also use "and" and "or" to break it down. Since words such as "and" and "or" may or may not be a delimiter, we keep all possible candidates in Y_s. For instance, given sentence 2), we have $Y_{2)} =$ {IBM, Nokia, Proctor, Gamble, Proctor and Gamble}.

The candidate super- and sub-ordinate concepts are further filtered by examining their presence in a search log, since intuitively, if an entity is prominent, web users will ask about it [18].

SuperOrdinateConceptDetection. In case $|X_s| > 1$, we must remove unlikely superordinate concepts from X_s until only one superordinate concept remains. For example, from sentence 1), we obtain $X_s = \{animals, dogs\}$ by syntactic extraction. Intuitively, since we might already have $(animals, cats) \in \Gamma$, we should be able to reject the possibility of $(dogs, cats)$ immediately. However, as we mentioned, the fact that a pair (x, y) is in Γ, does not necessarily mean it is true that x and y have an isA relationship. It only means we have encountered certain evidence for that claim. In other words, it is possible that we also have $(dogs, cats)$ in Γ. Thus, in order to remove unlikely superordinate concepts, we use a probabilistic approach for superordinate concept detection.

Let $X_s = \{x_1, \cdots, x_m\}$. We compute likelihood $p(x_k|Y_s)$ for $x_k \in X_s$. Without loss of generality, we assume x_1 and x_2 have the largest likelihoods[3]. We compute the ratio of likelihood $r(x_1, x_2)$ as follows and then we pick x_1 if the ratio is above a threshold:

$$r(x_1, x_2) = \frac{p(x_1|Y_s)}{p(x_2|Y_s)} = \frac{p(Y_s|x_1)p(x_1)}{p(Y_s|x_2)p(x_2)}$$

Assuming subordinate concepts in $Y_s = \{y_1, \cdots, y_n\}$ are independent given the superordinate concept, we have

$$r(x_1, x_2) = \frac{p(x_1)\prod_{i=1}^{n} p(y_i|x_1)}{p(x_2)\prod_{i=1}^{n} p(y_i|x_2)}$$

We compute the above ratio as follows: $p(x_i)$ is the percentage of pairs that have x_i as the superordinate concept in Γ, and $p(y_j|x_i)$ is the percentage of pairs in Γ that have y_j as the subordinate concept given x_i is the superordinate concept. Certainly, not every (x_i, y_j) appears in Γ, especially in the beginning when Γ is small. This leads to $p(y_j|x_i) = 0$, which makes it impossible for us to calculate the ratio. To avoid this situation, we let $p(y_j|x_i) = \epsilon$ where $\epsilon > 0$ is a small real value, when (x_i, y_j) is not in Γ.

Note that when we compute $r(x_1, x_2)$, Y_s may still contain a lot of noise. Practically, this is fine. It is possible that $r(x_1, x_2)$ may not be able to reach the required threshold at this point. Later, as Y_s is being cleaned up, we may be able to make a final decision.

SubOrdinateConceptDetection. Assume we have identified a single superordinate concept $X_s = \{x\}$ from a sentence. The next task is finding its subordinate concepts. It is a challenging task because the language is rife with ambiguities. For example:

- It is often difficult to detect where the list of subordinate concepts begins or ends in a sentence. For example, in sentence 3), the list of valid subordinate concepts ends at *Expedia*[4], and in sentence 4), the list begins at *Australia*.
- Delimiters such as ",", "*and*", and "*or*" may themselves appear in valid subordinate concepts (e.g., "Proctor *and* Gamble" is a single subordinate concept), which makes it difficult to identify subordinate concepts correctly.

Our subordinate concept detection method is based on two important observations. First, the closer a subordinate concept is to the *pattern keyword*, the more likely it is a valid subordinate concept. In fact, some extraction methods only take the closest one to improve precision. For example, in sentence 3), *Amazon* most likely is a valid subordinate concept because it is right after pattern keywords **such as**, and in sentence 4), *China* most likely is a valid subordinate concept as it is right before the pattern keywords **and other**. Second, if we are certain a subordinate concept at the k-th position from the pattern keyword is valid, then most likely subordinate concepts from position 1 to position $k-1$ are also valid.

Our strategy is to first find the largest scope wherein subordinate concepts are all valid, and then address the ambiguous issues inside the scope. Specifically, we find the largest k such that $p(y_k|x)$ is above a threshold, where y_k is the candidate subordinate concept at the k-th position from the pattern keywords. If, however, we cannot find any y_k that satisfies the condition, then we assume $k = 1$, provided that y_1 is well formed (e.g., it does not contain any delimiters such as *and* or *or*), because based on our observation, y_1 is most likely a valid subordinate concept.

Then, we study each candidate $y_1, \cdots, y_k$. For any y_i where $1 \le i \le k$, if y_i is not ambiguous, we add (x, y_i) to Γ if it is not already there, or otherwise we increase the count $n(x, y_i)$. If y_j is ambiguous, that is, we have multiple choices for position j, then we need to decide which one is valid. Assume we have identified $y_1, \cdots, y_{j-1}$ from position 1 to position $j-1$ as valid subordinate concepts, and assume we have two candidates at position j, that is,

[3] In practice, few X_s's have ≥ 3 candidate hypernyms.

[4] In fact, *Microsoft* should be the last valid subordinate concept in this case. However, *Microsoft notice you* may be incorrectly parsed.

$y_j \in \{c_1, c_2\}$. We compute the likelihood ratio $r(c_1, c_2)$ as follows and then we pick c_1 over c_2 if the ratio is above a threshold:

$$r(c_1, c_2) = \frac{p(c_1|x, y_1, \cdots, y_{j-1})}{p(c_2|x, y_1, \cdots, y_{j-1})}$$

Assume $y_1, \cdots, y_{j-1}$ are independent given x and y_j, we have:

$$r(c_1, c_2) = \frac{p(c_1|x) \prod_{i=1}^{j-1} p(y_i|c_1, x)}{p(c_2|x) \prod_{i=1}^{j-1} p(y_i|c_2, x)}$$

As before, $p(c_1|x)$ is the percentage of pairs in Γ where c_1 is a subordinate concept given x is the superordinate concept. The challenge is how to calculate $p(y_i|c_1, x)$ and $p(y_i|c_2, x)$. If we assume y_i is independent of c_1 and c_2, then the ratio of $r(c_1, c_2)$ reduces to $\frac{p(c_1|x)}{p(c_2|x)}$. In some cases, it is powerful to exclude incorrect subordinate concepts. For example, in sentence 4), we may eliminate *the Middle East* if p(*the Middle East*|*countries*) is much smaller than p(*Australia*|*countries*). However, in other cases, using the reduced form may not be powerful enough to disambiguate between c_1 and c_2, especially when neither (x, c_1) nor (x, c_2) is in Γ yet.

We note that subordinate concepts in a list exhibit common patterns. For example, a list of persons names show the same pattern: Most names contain two capitalized words. In sentence 3) "*Online presence helps companies such as Amazon, Expedia, and Microsoft notice you.*", the phrase *Microsoft notice you* raises a flag because it does not conform with the pattern exhibited by *Amazon* and *Expedia*. Based on this, we estimate $p(y_i|c, x)$ by taking into consideration syntactic patterns such as: i) The POS tag of c and y_i; 2) the number of words in c and y_i; and 3) the number of capitalized words in the subordinate concept. As a result, in sentence 3), both $p(Amazon|c, company)$ and $p(Expedia|c, company)$ where $c =$ '*Microsoft notice you*' are small because the pattern is different.

3. TAXONOMY CONSTRUCTION

We have obtained a large set of isA pairs in the previous step. Each pair is considered as an edge in the taxonomy. Our goal is to construct a taxonomy from these individual edges.

3.1 Problem Statement

We model the taxonomy as a DAG (directed acyclic graph). A node in the taxonomy is either a *concept* node (e.g., *company*), or an *instance* node (e.g., *Microsoft*). A concept contains a set of instances and possibly a set of subordinate concepts. An edge (u, v) connecting two nodes u and v means that u is a superordinate concept of v. Differentiating concept nodes from instance nodes is natural in our taxonomy: *Nodes without out-edges are instances, while other nodes are concepts*.

The obvious task of creating a graph out of a set of edges is the following: *For any two edges each having a node with the same label, should we consider them as the same node and connect the two edges?* Consider the following two cases:

1. For two edges e_1 = (fruit, apple), e_2 = (companies, apple), should we connect e_1 and e_2 on node "apple"?

2. For two edges e_1 = (plants, tree), e_2 = (plants, steam turbine), should we connect e_1 and e_2 on node "plants"?

The answer to both of the questions is obviously *No*, but how do we decide on these questions?

Clearly, words such as "apple" and "plants" may have multiple meanings (senses). So the challenge of taxonomy construction is to differentiate between these senses, and connect edges on nodes that have the same sense. We further divide the problem into two subproblems: i) Group concepts by their senses, i.e., decide whether the two *plants* in the 2nd question above mean the same thing; and ii) group instances by their senses, i.e., decide whether the two *apples* in the 1st question mean the same thing. We argue that we only need to solve the first sub-problem, because once we correctly group all the concepts by different senses, we can determine the meaning of an instance by its position in the concept hierarchy, i.e., its meaning depends on all the superordinate concepts it has.

We attack the problem of taxonomy construction in two steps. First, we identify some properties of the isA pairs we have obtained. Second, based on the properties, we introduce two operators that merge nodes belonging to the same sense, and we build a taxonomy using the operators we defined.

3.2 Senses

Let x^i denote a node with label x and sense i. Two nodes x^i and x^j are equivalent iff $i = j$. For an edge (x, y), if (x^i, y^j) holds, then (x^i, y^j) is a possible *interpretation* of (x, y). We denote this as $(x, y) \models (x^i, y^j)$. Given an edge (x, y), there are 3 possible cases for interpreting (x, y):

1. There exists a unique i and a unique j such that $(x, y) \models (x^i, y^j)$. For example, (*planets*, *Earth*). This is the most common case.

2. There exists a unique i and multiple j's such that $(x, y) \models (x^i, y^j)$. For example, (*objects*, *plants*).

3. There exists multiple i's and multiple j's such that $(x, y) \models (x^i, y^j)$. This case is very rare in practice.

Finally, it is impossible that there exist multiple i's but a unique j such that $(x, y) \models (x^i, y^j)$.

3.3 Properties

We reveal some important properties for the isA pairs we obtain through Hearst's patterns. In our discussion, we use the following sentences as our running example.

EXAMPLE 1. *A running example.*

a) ... <u>*plants*</u> ***such as*** trees *and* grass ...

b) ... <u>*plants*</u> ***such as*** trees, grass *and* herbs ...

c) ... <u>*plants*</u> ***such as*** steam turbines, pumps, *and* boilers ...

d) ... <u>*organisms*</u> ***such as*** plants, trees, grass *and* animals ...

e) ... <u>*things*</u> ***such as*** plants, trees, grass, pumps, *and* boilers ...

PROPERTY 1. *Let $s = \{(x, y_1), \cdots, (x, y_n)\}$ be the isA pairs derived from a sentence. Then, all the x's in s have a unique sense, that is, there exists a unique i such that $(x, y_j) \models (x^i, y_j)$ holds for all $1 \leq j \leq n$.*

Intuitively, it means that sentences like "*... plants* **such as** trees **and** boilers ..." are extremely rare. In other words, the superordinate concept in all the isA pairs from a single sentence has the same sense. For example, in sentence a), the senses of the word *plants* in (*plants*, *trees*) and (*plants*, grass) are the same. Following this property, we denote isA pairs from a sentence as $\{(x^i, y_1), \cdots, (x^i, y_n)\}$ by emphasizing that all the x's have the same sense.

PROPERTY 2. *Let $\{(x^i, y_1), \cdots, (x^i, y_m)\}$ denote pairs from one sentence, and $\{(x^j, z_1), \cdots, (x^j, z_n)\}$ from another sentence. If $\{y_1, ..., y_m\}$ and $\{z_1, ..., z_n\}$ are similar, then it is highly likely that x^i and x^j are equivalent, that is, $i = j$.*

Consider sentences a) and b) in Example 1. The set of subordinate concepts have a large overlap, so we conclude that the senses of the word *plants* in the two sentences are the same. The same thing cannot be said for sentences b) and c) as no identical subordinate concepts are found.

PROPERTY 3. *Let* $\{(x^i, y), (x^i, u_1), \cdots, (x^i, u_m)\}$ *denote pairs obtained from one sentence, and* $\{(y^k, v_1), \cdots, (y^k, v_n)\}$ *from another sentence. If* $\{u_1, u_2, ..., u_m\}$ *and* $\{v_1, v_2, ..., v_n\}$ *are similar, then it is highly likely that* $(x^i, y) \models (x^i, y^k)$.

Intuitively, it means the word *plants* in sentence d) has the same sense as the word *plants* in sentence a), because their subordinate concepts have considerable overlap. This is not true for sentence d) and c). For the same reason, the word *plants* in sentence e) could be interpreted as the sense of *plants* in a) and c) at the same time.

3.4 Node Merging Operations

Based on the three properties, we can immediately develop some mechanisms to join edges that have endnodes of the same sense.

First, based on Property 1, we know that every superordinate concept in the isA pairs derived from a single sentence has the same sense. Thus, we join such isA pairs on the superordinate concept node. This is shown in Figure 2. We call the taxonomy obtained from a single sentence a *local taxonomy*. A local taxonomy with root x^i is denoted as T_x^i.

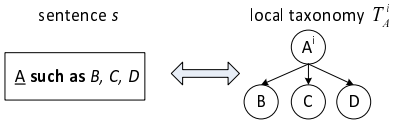


Figure 2: From a single sentence to a local taxonomy

Second, based on Property 2, given two local taxonomies rooted at x^i and x^j, if the child nodes of the two taxonomies demonstrate considerable similarity, we perform a *horizontal merge*, which is illustrated in Figure 3.

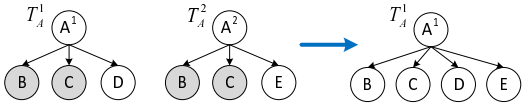


Figure 3: Horizontal merge

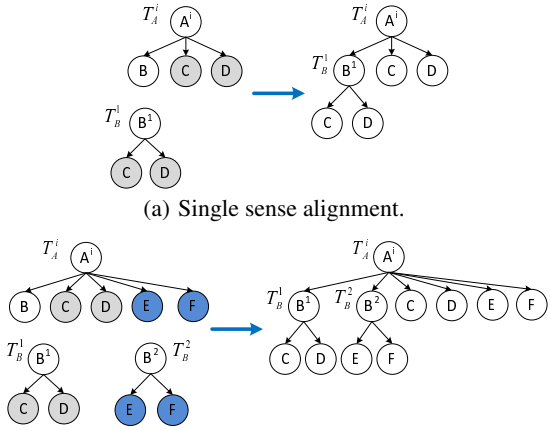
Third, based on Property 3, given two local taxonomies rooted at x^i and y^k, if x^i has a child node y, and the child nodes of the two taxonomies demonstrate considerable similarity, we merge the two taxonomies on the node y. We call this a *vertical merge*, and it is illustrated in Figure 4(a).

A special case for vertical merge is illustrated in Figure 4(b). Both T_B^1 and T_B^2 are vertically mergeable with T_A^i, as the child nodes of T_B^1 and T_B^2 having considerable overlap with the child nodes of T_A^i. However, the child nodes of T_B^1 and T_B^2 do not have considerable overlap, so it is still possible that they represent two senses. The result is two sub-taxonomies as shown in Figure 4(b).

3.5 Similarity Function

Suppose we use $Child(T)$ to denote the child nodes of a local taxonomy T. Given two local taxonomies T_1 and T_2, a central step lying in both the horizontal and vertical merge operations is to check the overlap of $Child(T_1)$ and $Child(T_2)$. In general, we can define a similarity function $f(A, B)$ such that two sets A and B are *similar* (denoted as $Sim(A, B)$) if $f(A, B) \geq \tau(A, B)$, where $\tau(A, B)$ is some prespecified threshold function.

We require our similarity function $f(A, B)$ to be defined in such a way that the following property could hold:



(a) Single sense alignment.

(b) Multiple sense alignment.

Figure 4: Vertical merge

PROPERTY 4. *If* A, A', B, *and* B' *are any sets s.t.* $A \subseteq A'$ *and* $B \subseteq B'$, *then* $Sim(A, B) \Rightarrow Sim(A', B')$.

This property is naturally implied by Properties 2 and 3. For instance, suppose T_1 and T_2 are two local taxonomies rooted at x^i and x^j, respectively. If $Sim(Child(T_1), Child(T_2))$, then according to Property 2, it's highly likely that $i = j$. Now given local taxonomies T_1' and T_2' also with roots x^i and x^j, such that $Child(T_1) \subseteq Child(T_1')$ and $Child(T_2) \subseteq Child(T_2')$. Since we have at least the same amount of overlapping evidence as in the case of T_1 and T_2, it is then rational to require $Sim(Child(T_1'), Child(T_2'))$ to hold so that we can also conclude that it is highly likely that $i = j$, based on T_1' and T_2'.

The simplest way then may be to define $f(A, B) = |A \cap B|$ and let $\tau(A, B)$ equal to some constant δ. Under this setting, we have $f(A', B') \geq f(A, B)$. Hence $f(A, B) \geq \delta \Rightarrow f(A', B') \geq \delta$, namely $Sim(A, B) \Rightarrow Sim(A', B')$.

3.6 The Algorithm

Algorithm 2 (in Appendix D) summarizes the framework of taxonomy construction. The whole procedure can be divided into three stages, namely, the *local taxonomy construction* stage (line 3-5), the *horizontal grouping* stage (line 6-10), and the *vertical grouping* stage (line 11-19). We first create a local taxonomy from each sentence $s \in S$. We then perform horizontal merges on local taxonomies whose root nodes have the same label. In this stage, small local taxonomies will be merged to form larger ones. Finally, we perform vertical merges on local taxonomies whose root nodes have different labels.

In Algorithm 2, we perform horizontal grouping before vertical grouping. It is not mandatory that we perform horizontal grouping first. As Theorem 1 shows, any sequence of grouping will eventually lead to the same taxonomy. However, as shown by Theorem 2, if we perform horizontal grouping then vertical grouping, the total number of merge operations is minimized, which is desirable in practice. The proofs of the two theorems can be found in [29].

THEOREM 1. *Let* $\mathcal{T}$ *be a set of local taxonomies. Let* $\mathbf{O}^\alpha$ *and* $\mathbf{O}^\beta$ *be any two sequences of vertical and horizontal operations on* $\mathcal{T}$. *Assume no further operations can be performed on* $\mathcal{T}$ *after* $\mathbf{O}^\alpha$ *or* $\mathbf{O}^\beta$. *Then, the final graph after performing* $\mathbf{O}^\alpha$ *and the final graph after performing* $\mathbf{O}^\beta$ *are the same.*

THEOREM 2. *Let* $\mathcal{O}$ *be the set of all possible sequences of operations, and let* $M = \min\{|\mathbf{O}| : \mathbf{O} \in \mathcal{O}\}$. *Suppose* $\mathbf{O}^\sigma$ *is the sequence that performs all possible horizontal merges first and all possible vertical merges next, then* $|\mathbf{O}^\sigma| = M$.

4. SCORING THE TAXONOMY

Knowledge is not black or white. Removing all the uncertainty is often impossible and harmful, as many applications rely on the uncertainty. Our philosophy is to learn to live with noisy data and make best use of it. For this purpose, we develop a set of measures to score the data. In this paper, we focus on scoring the isA relationships in Probase, although the same mechanism we develop works for other types of claims as well.

Consensus. The first measure is the *consensus* of a claim. We would like to know if a claim is true. Unfortunately, in many cases, truth does not exist, and in almost all cases, information from external data sources is unreliable, conflicting, or erroneous. However, consensus is measurable. We regard external information as evidence for a claim. We aggregate this evidence and derive weighted consensus for each claim. Based on the consensus scores, we implement a probabilistic evidential reasoning framework, which allows us to incorporate uncertain knowledge from other data sources.

Specifically, assume Probase contains a claim E which is derived from a set of sentences or evidence $\{s_1, \cdots, s_n\}$ on the web. Assume each piece of evidence s_i is associated with a weight p_i that reflects the strength of the evidence. We may treat p_i as a probability, that is, evidence s_i has probability p_i to be right about the claim E. We can simply assume E is false iff every piece of evidence in $s_1, \cdots, s_n$ is false. Assuming evidence is independent, we have

$$p(E) = 1 - p(\bar{E}) = 1 - p(\cap_{i=1}^{n} \bar{E}_i) = 1 - \prod_{i=1}^{n}(1 - p_i) \quad (1)$$

Such a model is certainly very naive. However, it serves to demonstrate the extensibility of the framework. By treating information from other sources as new evidence, we can easily update $p(E)$ to revise our belief about the claim.

We use more sophisticated models for evidential reasoning. First, we adopt a voting model. Each evidence casts a vote on the claim: It votes true with probability p_i, and false with probability $1 - p_i$. We consider the final claim true if we receive more than n/c true votes (c is a constant). The voting model extends the urns model [9] for managing redundant information in extraction. Second, we take into consideration *negative evidence*. For instance, a claim that x is an instance of Y may be used as a negative piece of evidence for another claim that x is an instance of Y', based on a function that measures the congruity between Y and Y'. In the simplest case, we know that a claim $X = 2$ can be considered as a negative evidence for claim $X = 3$. However, since knowledge is not black or white, for most claims it is difficult to find their exact negative form, which is why we develop a function to measure the congruity between two concepts for hypernym-hyponym claims. Due to space constraints, we refer readers to [31] for in-depth discussions on how Probase performs reasoning over evidence on the Web.

A remaining important issue is how do we derive p_i for evidence s_i? We consider two factors. First, p_i may depend on the type of the information source (e.g., we consider evidence coming from the New York Times to be more credible than those from a public forum). Second, p_i may depend on how confident the information extraction process is when it identifies evidence s_i in the text. In our work, we characterize each s_i by a set of features F_i. Then, we use a model (e.g., Naive Bayes) to derive p_i.

Typicality. A robin is more typical of the concept bird than is ostrich. The *Company* concept contains instances including *Microsoft*, *Xyz Inc.*, etc., and *Arnold Schwarzenegger* belongs to multiple concepts, including *Politician*, *Actor*, etc. We want to know, for the *Company* class, which company is more important or representative (typical), *Microsoft* or *Xyz Inc.*? And for *Arnold Schwarzenegger*, is he more known as a *politician* or as an *actor?*

Information about typicality is essential to many applications. For example, many information extraction algorithms start with a small set of "seeds" to bootstrap a learning process [16, 28]. If the algorithm wants to find information about companies, then it should prefer *Microsoft* over *Xyz Inc.* as a seed, because *Microsoft* is a better known company, and if it wants to find information about politicians, then it should avoid using *Arnold Schwarzenegger* as a seed, otherwise information about actors will turn up.

Our importance measure is a generalization of the TF-IDF score. Let $r(x, i)$ denote the typicality of instance i to concept x.

$$r(x, i) = \mathtt{TF}(x, i) \cdot \mathtt{IDF}(x, i) \quad (2)$$

We define the TF score as:

$$\mathtt{TF}(x, i) = \frac{N(x, i)}{\sum_{i' \in I_x} N(x, i')}. \quad (3)$$

where $N(x, i)$ is i's frequency in x. However, x is not a standalone document, instead, it is a concept in a well-connected taxonomy. To take the taxonomy into consideration, we define $N(x, i) = n(x, i) + \sum_{v \in V(x)} n(v, i)$, where $n(x, i)$ is i's frequency in x alone, and $V(x)$ contains concepts that are x's subconcepts or equivalent concepts. Furthermore, instead of using $n(x, i)$ directly, we also use a weighted version, where the weight is the consensus score. Next, we define the IDF score as:

$$\mathtt{IDF}(x, i) = \log \frac{|S|}{|S(i)| - |S_x(i)| + 1}. \quad (4)$$

where S is the entire set of concepts, $S(i)$ is the set of concepts that contain instance i, and $S_x(i)$ is the set of x's sub concepts and equivalent concepts that contain instance i. We add 1 in the denominator to avoid the divide-by-zero error. In the case i is contained only by x, we have $|S_x(i)| = 1$, $N(x, i) = n(x, i)$, and the generalized TF-IDF score degenerates to its original form.

Similarity. We want to know, for instance, how similar or how related are the *company* concept and the *corporation* concept, or the *celebrity* concept and the *model* concept? A few approaches (e.g., [21]) have been proposed to measure the similarity of two concepts in a taxonomy. Our unique advantage is the richness of Probase. We measure the similarity between two concepts by their *content*. Intuitively, if two concepts are similar, we expect they have common instances, similar sub concepts (*child classes*), and similar super concepts (*parent classes*).

Given two concepts x_1 and x_2, we measure the similarity of their instances, subordinate concepts, and superordinate concepts independently, and then we aggregate them as the final similarity between x_1 and x_2:

$$s(x_1, x_2) = \alpha \cdot s(I_{x_1}, I_{x_2}) + \beta \cdot s(C_{x_1}, C_{x_2}) + \gamma \cdot s(P_{x_1}, P_{x_2})$$

where we require $\alpha + \beta + \gamma = 1$ so that $s(x_1, x_2)$ is between 0 and 1. For definitions of $s(.,.)$, see Appendix E.

5. EXPERIMENTAL RESULTS

The proposed taxonomy inference framework was implemented on a cluster of servers using the *Map-Reduce* model. For our latest version, we used 7 hours and 10 machines to find all the isA pairs, and we then used 4 hours and 30 machines to construct the taxonomy. Due to space constraints, only highlights of the results are provided. Readers are referred to `http://research.microsoft.com/probase/experiments.htm` for complete experimental results.

We extract 326,110,911 sentences from a corpus containing 1,679,189,480 web pages, after sentence deduplication. To the best of our knowledge, the scale of our corpus is one order of magnitude larger than the previously known largest corpus [22]. We then

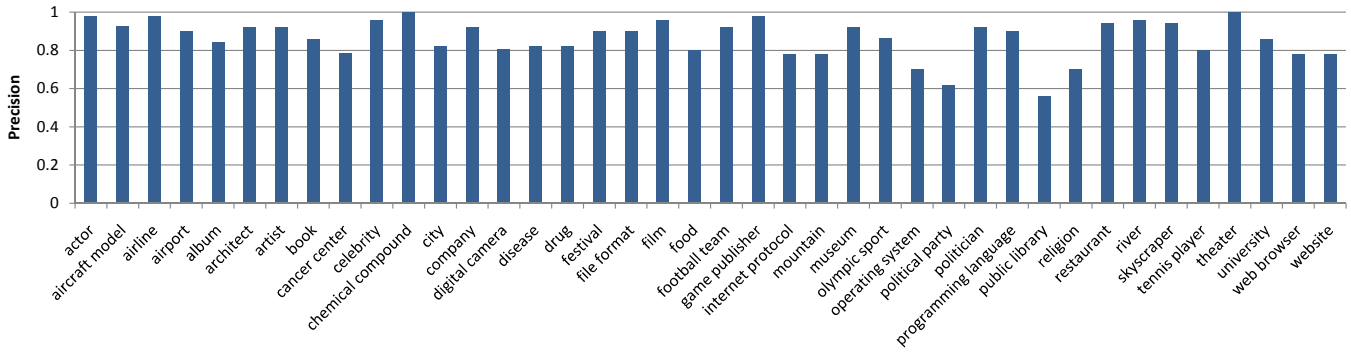


Figure 5: Precision of extracted pairs

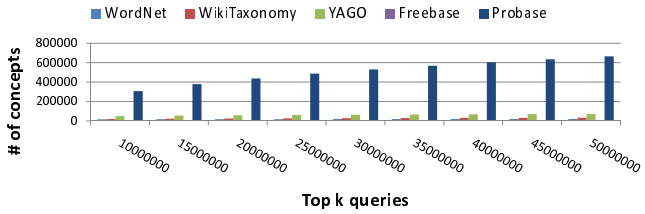


Figure 6: Number of relevant concepts in taxonomies.

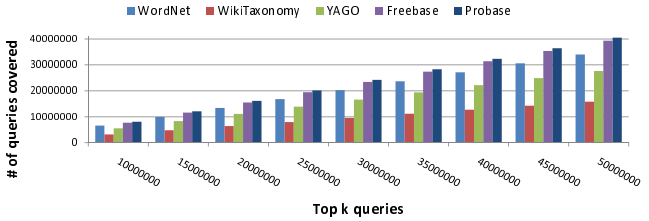


Figure 7: Taxonomy coverage of the top 50 million queries.

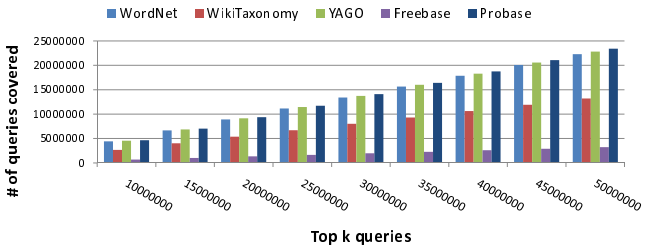


Figure 8: Concept coverage of the top 50 million queries.

extract 143,328,997 isA pairs from the sentences, with 9,171,015 distinct hypernym labels and 11,256,733 distinct hyponym labels.

The inferred taxonomy contains 2,653,872 distinct concepts (down from 9.17 million after the extraction phase), 16,218,369 distinct concept-instance pairs, and 4,539,176 distinct concept-subconcept pairs (20,757,545 pairs in total). The number of concept labels decreases since we have changed all labels to lowercases and flatten the concepts with only one instance (and refer to them as instances). Below we analyze the characteristics of the concept space and isA relationship space of Probase.

5.1 Concept Space

Given that Probase has many more concepts than other taxonomies. A reasonable question to ask is how many of these concepts are *relevant*. This question is akin to the *precision* measure in information retrieval (IR). Here for the purpose of comparison, we define a concept to be relevant [5], if it appeared at least once in web queries. We analyzed Bing's query log from a two-year period, sorted the queries in decreasing order of their frequency (i.e., the number of times they are issued through Bing), and computed the number of useful concepts in each of the 5 taxonomies with respect to the top 50 million queries. Figure 6 shows the result.

In total, 664,775 concepts are considered relevant in Probase, compared to 70,656 in YAGO. This reflects the well-known *long-tail* phenomena of user queries. While a small number of basic concepts (e.g., *company*, *city*, *country*) representing common sense knowledge appear very frequently in user queries, Web users do mention other less well-known concepts. Probase does a better job at capturing these concepts in the long tail (see Figure 10) and hence has a better chance of understanding these user queries.

We next measure the *taxonomy coverage* of queries by Probase, which is akin to the *recall* measure in IR. A query is said to be *covered* by a taxonomy if the query contains at least one *concept or instance* within the taxonomy.[6] Figure 7 compares the coverage of queries by Probase taxonomy against four other general-purposed open-domain taxonomies: *WordNet*, *WikiTaxonomy*, *YAGO*, and *Freebase*. Probase outperforms all the other taxonomies on the coverage of top 10 million to top 50 million queries. In all, Probase covers 40,517,506 (or, 81.04%) of the top 50 million queries.

We also measure *concept coverage*, which is the number of queries containing at least one *concept* in the taxonomy. Figure 8 compares the concept coverage by Probase against the other four taxonomies. Again, Probase outperforms all the others. Note that, although Freebase presents comparable taxonomy coverage with Probase in Figure 7, its concept coverage is much smaller.

5.2 isA Relationship Space

There are two kinds of isA relationships in Probase: the concept-subconcept relationship which are the edges connecting internal nodes in the hierarchy, and the concept-instance relationship which are the edges connecting a leaf node.

Table 1 compares the concept-subconcept relationship space of Probase with the other taxonomies. The *level* of a concept is defined to be one plus the length of the longest path from it to a *leaf concept* (i.e., concept without any subconcepts/children). All leaf concepts thus receive a level of 1. Table 1 shows that even with an order of magnitude larger number of concepts, Probase still has a similar hierachical complexity to the other taxonomies. The exception is Freebase which exhibits trivial values on these measured metrics because it has no isA relationship among its concepts at all.

We also compare the concept-instance relationships between Probase and Freebase. We choose Freebase since it is the only existing taxonomy with comparable scale on instance space (24,483,434

[5]This definition makes sense especially in the context of web search.

[6]This definition is reasonable because the taxonomy *contributes* to the understanding of the query if it can understand at least one term in the query.

	# of isA pairs	Avg # of children	Avg # of parents	Avg level
WordNet	283,070	11.0	2.4	1.265
WikiTaxonomy	90,739	3.7	1.4	1.483
YAGO	366,450	23.8	1.04	1.063
Freebase	0	0	0	1
Probase	4,539,176	7.53	2.33	1.086

Table 1: The concept-subconcept relationship space.

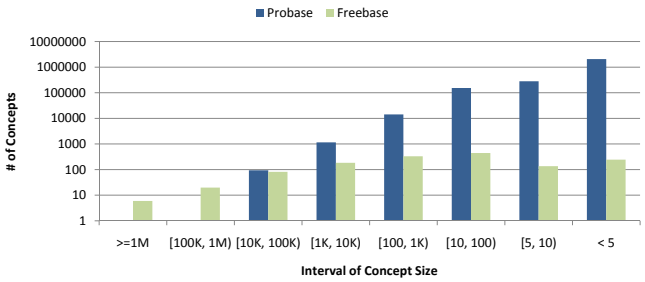


Figure 9: Concept size distributions in Probase and Freebase.

concept-instance pairs). We define *concept size* to be the number of instances directly under a concept node. Figure 9 [7] compares distributions of concept sizes in Probase and Freebase. While Freebase focuses on a few very popular concepts like "*track*" and "*book*" which include over two million instances, Probase has many more medium to small size concepts. In fact, the top 10 concepts in Freebase contain 17,174,891 concept-instance pairs, or 70% of all the pairs it has. In contrast, the top 10 concepts in Probase only contains 727,136 pairs, or 4.5% of its total. Therefore, Probase provides a much broader coverage on diverse topics, while Freebase is more informative on specific topics. On the other hand, the instances of large concepts in Freebase like *book* are mostly from specific websites like Amazon, which could be easily merged into Probase using the the integration framework proposed in Section F.

To estimate the correctness of the extracted isA pairs in Probase, we create a benchmark dataset (Table 4 in Appendix G) containing 40 concepts in various domains. The concept size varies from 21 instances (for *aircraft model*) to 85,391 (for *company*), with a median of 917. Benchmarks with similar number of concepts and domain coverage have also been reported in information extraction research [17]. For each concept, we randomly pick up to 50 instances/subconcepts and ask human judge to evaluate their correctness and hence also the precision of the extraction algorithm. Figure 5 shows the result. The average precision of all pairs in benchmark is 86%, which outperforms precision reported from other previous notable information extraction frameworks like KnowItAll [10] (64% on average), NELL(74%) and TextRunner(80% on average). It is not fair to directly compare our results with Wikipedia-based frameworks like WikiTaxonomy (86% precision) and YAGO (97% precision), whose data sources are much cleaner. Nevertheless, only YAGO has a better overall precision than Probase.

6. CONCLUSION

In this paper, we presented a probabilistic framework which automatically inferences a universal taxonomy from the entire web. This taxonomy, to the best of our knowledge, is the largest and the most comprehensive of its kind to date. The framework allows the integration of both precise and ambiguous knowledge and even tolerates inconsistencies and errors which are common on the Web. The taxonomy built by this framework can be an important source of knowledge for concept related search, which is currently not well supported by traditional keyword-based search engines, and other knowledge-based applications.

[7] Logarithmic scale on the Y-axis.

7. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *ISWC/ASWC*, pages 722–735, 2007.

[2] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the web. In *IJCAI*, pages 2670–2676, 2007.

[3] P. Bloom. Glue for the mental world. *Nature*, (421):212–213, Jan 2003.

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

[5] S. A. Caraballo. Automatic construction of a hypernym-labeled noun hierarchy from text. In *ACL*, 1999.

[6] 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*, 2010.

[7] B. Crysmann, A. Frank, B. Kiefer, S. Mueller, G. Neumann, J. Piskorski, U. Schäfer, M. Siegel, H. Uszkoreit, F. Xu, M. Becker, and H.-U. Krieger. An integrated archictecture for shallow and deep processing. In *ACL*, pages 441–448, 2002.

[8] N. N. Dalvi, R. Kumar, B. Pang, R. Ramakrishnan, A. Tomkins, P. Bohannon, S. Keerthi, and S. Merugu. A web of concepts. In *PODS*, pages 1–12, 2009.

[9] D. Downey, O. Etzioni, and S. Soderland. A probabilistic model of redundancy in information extraction. In *IJCAI*.

[10] O. Etzioni, M. J. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in knowitall. In *WWW*, pages 100–110, 2004.

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

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

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

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

[15] A. G. Parameswaran, H. Garcia-Molina, and A. Rajaraman. Towards the web of concepts: Extracting concepts from large datasets. *PVLDB*, 3(1):566–577, 2010.

[16] M. Pasca. Acquisition of categorized named entities for web search. In *CIKM*, pages 137–145, 2004.

[17] M. Pasca. Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In *WWW*, 2007.

[18] M. Pasca. Turning web text and search queries into factual knowledge: Hierarchical class attribute extraction. In *AAAI*, 2008.

[19] S. P. Ponzetto and R. Navigli. Large-scale taxonomy mapping for restructuring and integrating wikipedia. In *IJCAI*, 2009.

[20] S. P. Ponzetto and M. Strube. Deriving a large-scale taxonomy from wikipedia. In *AAAI*, 2007.

[21] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.

[22] A. Ritter, S. Soderland, and O. Etzioni. What is this, anyway: Automatic hypernym discovery. In *AAAI Spring Symposium on Learning by Reading and Learning to Read*, 2009.

[23] E. Segal, D. Koller, and D. Ormoneit. Probabilistic abstraction hierarchies. In *NIPS*, pages 913–920, 2001.

[24] P. Singh, T. Lin, E. Mueller, G. Lim, T. Perkins, and W. Li Zhu. Open Mind Common Sense: Knowledge acquisition from the general public. *On the Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE*, pages 1223–1237, 2002.

[25] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *NIPS*, 2005.

[26] R. Snow, D. Jurafsky, and A. Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *ACL*, 2006.

[27] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.

[28] R. C. Wang and W. W. Cohen. Language-independent set expansion of named entities using the web. In *ICDM*, pages 342–350, 2007.

[29] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Towards a probabilistic taxonomy of many concepts. Technical report, Microsoft Research, 2010.

[30] D. Zhang and W. S. Lee. Web taxonomy integration using support vector machines. In *WWW*, pages 472–481, 2004.

[31] Y. Zhou, B. Shao, H. Wang, and R. Jin. Evidential reasoning in probase. 2010. Under submission.

APPENDIX

A. RELATED WORK

Extracting hypernym-hyponym relationships from the whole web is much more challenging than doing that in specific domains. When extracting such relationships from a single website like Wikipedia, certain structural information can be utilized. For instance, the "is-a" facts in DBpedia [1] are extracted from the data contained in the so-called *infobox* template of Wikipedia. However, such structural information is not generally available on the Web. Instead, approaches based on hand-written patterns became prevalent and dominant when dealing with Web-scale data. The problem of extracting hypernym-hyponym pairs from text with hand-written linguistic patterns was first studied by Hearst [12]. A later study by Snow *et al.* [25] showed that Hearst's patterns were among the highest precision ones from nearly 70,000 patterns they induced for hypernym-hyponym identification. Due to its simplicity and effectiveness, methods based on Hearst's patterns have been applied to many Web-scale information extraction systems such as KnowItAll [10], TextRunner [2], and Pasca's recent work [18]. However, there was only limited discussion [10, 22] on the extraction issues we encountered in practice (specified at the beginning of Section 2), and the techniques introduced there cannot resolve all those issues. For example, Etzioni *et al.* [10] only targeted the problem of extracting hyponyms that are *named entities* (i.e., *proper nouns*), with the restriction that the words within proper nouns need to be capitalized. This obviously does not work on hyponyms that are also *concepts*, which are essential to our taxonomy inferencing task. On the other hand, Ritter *et al.* [22] only focused on the extraction of hypernyms located at the exact positions specified by Hearst's patterns, which, as we have shown, may miss many *correct* hypernyms. The deficiency of the existing state-of-the-art techniques motivates us to propose the extraction framework in Section 2.2. Our iterative extraction algorithm shares a similar spirit to NELL [6] despite that it took the latter a much longer time to extract a much smaller ontology.

Besides manually constructed taxonomies through expert and community efforts such as WordNet, Cyc and Freebase, automatic taxonomy construction has been extensively studied in the literature as well [5, 23, 26, 27, 20]. The most notable towards this effort may be WikiTaxonomy [20] and YAGO [27]. Both of them attempt to derive a taxonomy[8] from Wikipedia *categories*, which could be viewed as thematic topics used to classify Wikipedia articles. However, the taxonomies thus inferred (not surprisingly) contain only concepts and entities from the Wikipedia site, which is very limited compared to either Freebase or Probase. For instance, the "is-a" semantic links within WikiTaxonomy and YAGO, are at least an order of magnitude less than what is available in Probase. This also explains the high extraction precision (97.7%) reported by YAGO, since Wikipedia data is much cleaner than Web data. However, the coverage is an inherent issue in such Wikipedia-based approaches, because the growth of Wikipedia is much slower than the rest of the Web. Other older works usually adopted more complex approaches on even smaller data sets, which were hard to scale to the Web.

In addition, while most of the works mentioned above treat the problem of taxonomy inferencing in a deterministic way, a few exceptions [23, 26] have attempted to model the construction procedure into a probabilistic framework. However, the inference models usually include complex learning and optimization phases, which are difficult to implement and not scalable on Web-sized data. Moreover, even though the inference process is probabilistic, the resulting taxonomy remains deterministic due to the optimization phase. To the best of our knowledge, we are among the first to try to build a *probabilistic* taxonomy that captures the inherent uncertainty within the structure of human knowledge.

Data integration has been extensively studied by the database community for a long time. However, only limited research efforts were made in *taxonomy* integration. Zhang *et al.* [30] studied the problem of classifying objects from one data source into categories within another data source, which was hard to directly apply to integrating general taxonomies with hierarchical structures. Recently, Ponzetto *et al.* [19] proposed an approach to restructuring the WikiTaxonomy by aligning it with WordNet. However, the general problem of how to integrate taxonomy categories and structures from multiple sources remains open. As we have pointed out, handling uncertainty is crucial in such integration tasks, and the framework proposed in Section F is the first effort we made in this direction.

[8]YAGO actually tries to derive a more general ontology with taxonomic information included.

B. SCALE OF OPEN DOMAIN TAXONOMIES

	# of concepts
Freebase [4]	1,450
WordNet [11]	25,229
WikiTaxonomy [20]	$<$ 127,325
YAGO [27]	149,162
DBPedia [1]	259
ResearchCyc [13]	$\approx$ 120,000
KnowItAll [10]	N/A
TextRunner [2]	N/A
OMCS [24]	N/A
NELL [6]	123
Probase	**2,653,872**

Table 2: Scale of open-domain taxonomies

Figure 10 shows the frequency distribution of the concepts in Probase. The Y axis is the frequency (popularity) of the concepts in a logarithmic scale, and the X axis is dozens of concepts ordered by their popularity (these concepts are sampled from the 2.7 million concepts in Probase).

C. HEARST PATTERNS

ID	Pattern
1	*NP* such as {*NP*,}*{(or | and)} *NP*
2	such *NP* as {*NP*,}*{(or | and)} *NP*
3	*NP*{,} including {*NP*,}*{(or | and)} *NP*
4	*NP*{,*NP*}*{,} and other *NP*
5	*NP*{,*NP*}*{,} or other *NP*
6	*NP*{,} especially {*NP*,}*{(or | and)} *NP*

Table 3: Heast patterns (*NP* stands for *noun phrase*)

D. ALGORITHMS

Algorithm 1 and 2 describe our frameworks of data extraction and taxonomy construction, respectively.

E. SIMILARITY IN SCORING

We tested several similarity metrics and find the following weighted version of Jaccard similarity coefficient works the best:

$$Sim(I_{x_1}, I_{x_2}) = \frac{\sum_{i \in I_{x_1} \cap I_{x_2}} \min\{r(x_1, i), r(x_2, i)\}}{\sum_{i \in I_{x_1} \cup I_{x_2}} \max\{r(x_1, i), r(x_2, i)\}}, \quad (5)$$

$$Sim(C_{x_1}, C_{x_2}) = \frac{\sum_{c \in C_{x_1} \cap C_{x_2}} \min\{r(x_1, c), r(x_2, c)\}}{\sum_{c \in C_{x_1} \cup C_{x_2}} \max\{r(x_1, c), r(x_2, c)\}}, \quad (6)$$

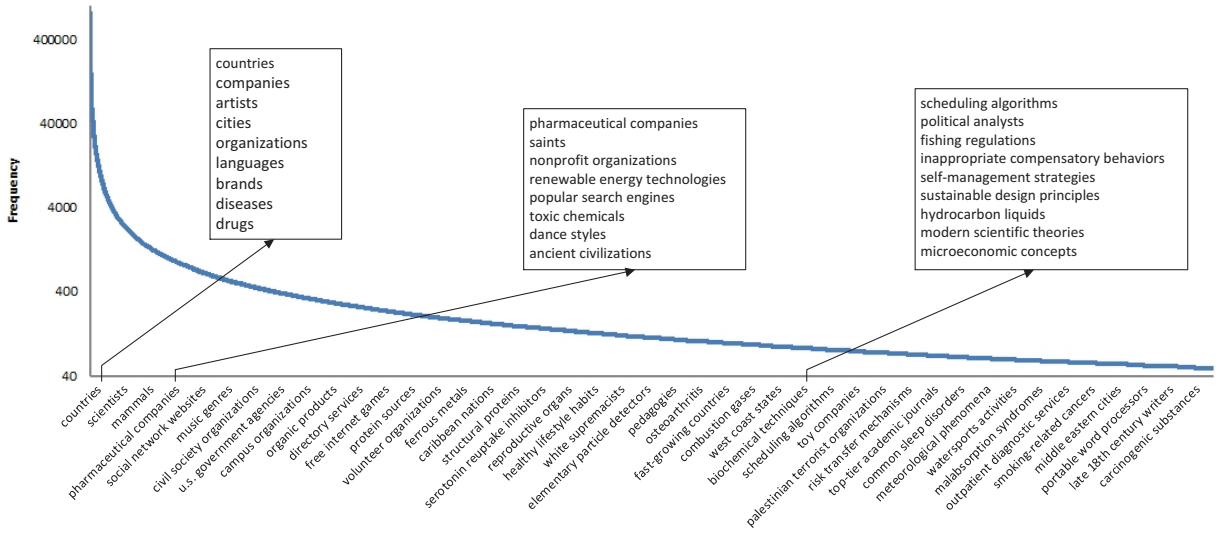


Figure 10: Frequency distribution of the 2.7 million concepts

Algorithm 1: isA extraction

Input: S, sentences from web corpus that match Hearst's patterns
Output: Γ, set of isA pairs

$\Gamma \leftarrow \emptyset$;
repeat
  foreach $s \in S$ **do**
    $X_s, Y_s \leftarrow SyntacticExtraction(s)$;
    if $|X_s| > 1$ **then**
      $X_s \leftarrow SuperOrdindateConceptDetection(X_s, Y_s, \Gamma)$;
    end
    if $|X_s| = 1$ **then**
      $Y_s \leftarrow SubOrdinateConceptDetection(X_s, Y_s, \Gamma)$;
      add valid isA pairs to Γ;
    end
  end
until *no new pairs added to* Γ;
return Γ;

$$Sim(P_{x_1}, P_{x_2}) = \frac{\sum_{p \in P_{x_1} \cap P_{x_2}} \min\{r(p, x_1), r(p, x_2)\}}{\sum_{p \in P_{x_1} \cup P_{x_2}} \max\{r(p, x_1), r(p, x_2)\}}. \quad (7)$$

We obtain the similarity between x_1 and x_2 using a simple linear interpolation scheme

$$Sim(x_1, x_2) = \alpha \cdot Sim(I_{x_1}, I_{x_2}) + \beta \cdot Sim(C_{x_1}, C_{x_2}) + \gamma \cdot Sim(P_{x_1}, P_{x_2}), \quad (8)$$

where we require $\alpha + \beta + \gamma = 1$ so that the resulted $Sim(x_1, x_2)$ is between 0 and 1.

F. ENRICH THE TAXONOMY

In addition to hypernym-hyponym pairs extracted by applying Hearst's patterns on web pages, there are other data sources on the Web which may provide complementary knowledge. Such data sources include manually constructed taxonomies like WordNet, or semi-automatically constructed taxonomies like the ones from Freebase and Wikipedia. Thanks to our probabilistic taxonomy construction framework, we could integrate other taxonomy data seamlessly into Probase. In this section, we describe how to extend the probabilistic framework to tackle the problem of taxonomy integration.

Algorithm 2: Taxonomy construction

Input: S: the set of sentences each containing a number of isA pairs.
Output: T: the taxonomy graph.

Let $\mathcal{T}$ be the set of local taxonomies;
$\mathcal{T} \leftarrow \emptyset$;
foreach $s = \{(x^i, y_1), ..., (x^i, y_n)\} \in S$ **do**
  Add a local taxonomy T_x^i into $\mathcal{T}$;
end
foreach $T_x^i \in \mathcal{T}, T_x^j \in \mathcal{T}$ **do**
  if $Sim(Child(T_x^i), Child(T_x^j))$ **then**
    $HorizontalMerge(T_x^i, T_x^j)$;
  end
end
foreach $T_x^i \in \mathcal{T}$ **do**
  foreach $y \in Child(T_x^i)$ **do**
    foreach $T_y^m \in \mathcal{T}$ **do**
      if $Sim(Child(T_x^i), Child(T_y^m))$ **then**
        $VerticalMerge(T_x^i, T_y^m)$;
      end
    end
  end
end
Let the graph so connected be T;
return T;

Let $\mathcal{T}$ be the set of taxonomies that we want to integrate. We will choose our probabilistic taxonomy T as the *target*, and treat other taxonomies in $\mathcal{S} = \mathcal{T} - \{T\}$ as *source*s.

We assume each $T_S \in \mathcal{S}$ has the general representation introduced in Section 3.2: Each node in T_S with label v and sense i is represented as v^i. The integration process is then very similar to our taxonomy construction framework, as shown in Algorithm 3.

We first horizontally group the local taxonomies contained in T_S and T. If a local taxonomy in T_S cannot be merged with any existing local taxonomies within T, it will be treated as a new local taxonomy and inserted into T. For each local taxonomy now in T after horizontal grouping, we collect (symbolic) subordinate con-

Algorithm 3: Taxonomy enrichment

Input: T_S and T: the source and target taxonomy, respectively.
Output: T': the target taxonomy after integrated with T_S

Let $\mathcal{T}_S$ be the set of local taxonomies contained T_S, and $\mathcal{T}_T$ be the set of local taxonomies contained in T;
foreach $S_x^i \in \mathcal{T}_S$ **do**
  foreach $T_x^j \in \mathcal{T}_T$ **do**
    if $Sim(Child(S_x^i), Child(T_x^j))$ **then**
      $HorizontalMerge(S_x^i, T_x^j)$;
    end
  end
end
foreach $T_x^i \in \mathcal{T}_T$ **do**
  foreach $y \in Child(T_x^i)$ and $y \in T_S$ **do**
    foreach $y^m \in \mathcal{T}_T$ **do**
      if $Sim(Child(T_x^i), Child(T_y^m))$ **then**
        $VerticalMerge(T_x^i, T_y^m)$;
      end
    end
  end
end
Recompute scores;
Let T' be the target taxonomy after integration with T_S;
return T';

cept nodes that also appear in T_S. For each y of these nodes, we try to connect it to the local taxonomies with label y as the root in T after integration, by applying the vertical merge operations. Finally, we will update the scores, as discussed next.

In section 4, we have discussed a simple framework to integrate multiple evidence of a given pair (x, y). There, each sentence s_i $(1 \leq i \leq n)$ with $(x, y) \in s$ is treated as a piece of evidence with p_i, and the final consensus of (x, y) is modeled as $c(x, y) = 1 - \prod_{i=1}^{n}(1 - p_i)$. This framework can be directly extended to do the recomputation of the consensus scores, where we now treat each data source as an evidence. We assume each data source has its own way to give a consensus score to the pairs[9]. For the pair (x, y)[10], suppose we have evidence from k data sources with consensus scores q_j $(1 \leq j \leq k)$, then $c(x, y) = 1 - \prod_{j=1}^{k}(1 - q_j)$. The importance and similarity scores can also be adjusted if other information like frequency of the pair (x, y) in T_S is available.

G. BENCHMARK

Table 4 shows the benchmark we used in our experimental evaluation.

H. EXPERIMENTAL RESULTS FOR TAXONOMY SCORING

We conducted some experiments to evaluate the effectiveness of the scoring in Probase.

H.1 Consensus Score

We experimented with the simple model in Equation (1) for computing consensus of a claim using the benchmark concepts. We expect that the consensus scores to be approximately equal to the actual percentage of true claims as the number of evidences grows. This is verified in Figure 11. The average consensus scores matches the actual percentage of true claims (checked by human judges) quite well, except when there is only one evidence. Figure 11 has an uneven scale on the x-axis because the frequency distribution of claims in Probase has a long tail (in Figure 10).

[9]This is a reasonable assumption in general, and there are various ways to give consensus scores. The way we compute consensus scores for our taxonomy T serves as an example when the data is inherently noisy. For data sources coming from manual effort, we might simply assign a unique (and high) probability for each pair.

[10]We assume that adding the pair (x, y) will not lead to a cycle in T, otherwise, we simply discard (x, y).

Concept (# of Instances)	Representative Instances
actor (3466)	Tom Hanks, Marlon Brando, George Clooney
aircraft model (21)	Airbus A320-200, Piper PA-32, Beech-18
airline (1221)	British Airways, Deltae
airport (790)	Heathrow, Gatwick, Stansted
album (1938)	Thriller, Big Calm, Dirty Mind
architect (882)	Frank Gehry, Le Corbusier, Zaha Hadid
artist (57482)	Picasso, Bob Dylan, Madonna
book (8628)	Bible, Harry Potter, Treasure Island
cancer center (55)	Fox Chase, Care Alliance, Dana-Farber
celebrity (8381)	Madonna, Paris Hilton, Angelina Jolie
chemical compound (308)	carbon dioxide, phenanthrene, carbon monoxide
city (9632)	New York, Chicago, Los Angeles
company (85391)	IBM, Microsoft, Google
digital camera (64)	Canon, Nikon, Olympus
disease (8784)	AIDS, Alzheimer, chlamydia
drug (5417)	tobacco, heroin, alcohol
festival (3039)	Sundance, Christmas, Diwali
file format (698)	PDF, JPEG, TIFF
film (13402)	Blade Runner, Star Wars, Clueless
food (4875)	beef, dairy, French fries
football team (59)	Real Madrid, AC Milan, Manchester United
game publisher (99)	Electronic Arts, Ubisoft, Eidos
internet protocol (168)	HTTP, FTP, SMTP
mountain (832)	Everest, the Alps, the Himalayas
museum (2441)	the Louvre, Smithsonian, the Guggenheim
olympic sport (62)	gymnastics, athletics, cycling
operating system (86)	Linux, Solaris, Microsoft Windows
political party (254)	NLD, ANC, Awami League
politician (953)	Barack Obama, Bush, Tony Blair
programming language (520)	Java, Perl, PHP
public library (39)	Haringey, Calcutta, Norwich
religion (1115)	Christianity, Islam, Buddhism
restaurant (4546)	Burger King, Red Lobster, McDonalds
river (3050)	Mississippi,the Nile, Ganges
skyscraper (121)	the Empire State Building, the Sears Tower, Burj Dubai
tennis player (46)	Maria Sharapova, Andre Agassi, Roger Federer
theater (632)	Metro, Pacific Place, Criterion
university (2048)	Harvard, Stanford, Yale
web browser (232)	Internet Explorer, Firefox, Safari
website (3487)	YouTube, Facebook, MySpace

Table 4: Benchmark concepts and their representative instances

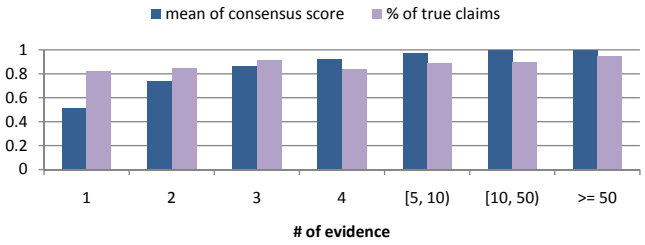


Figure 11: Consensus score vs. the actual percentage of true claims

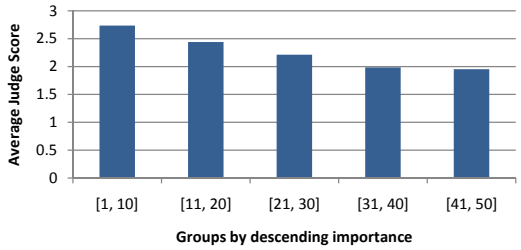


Figure 12: Importance evaluation results

H.2 Importance Score

Table 4 lists some typical instances according to our importance ranking framework in Section 4. We further conduct a user study for this relatively subjective measure. First, we pick 10 concepts, and the top 50 instances for each concept, according to their importance scores. Then, we invite 4 users to manually score the importance of the instances (with order shuffled) in their respective concepts, as 3 (very important), 2 (correct but not very important), 1 (unknown), and 0 (incorrect).

We divide the 50 instances of each concept into 5 groups by their importance score ranks (*i.e.* top 10 instances from each concept go to Group 1, second 10 instances from each concept go to Group 2, and so on), and then compute the average judge scores assigned to instances within each group. Figure 12 shows that the importance of the instances in their classes, as perceived by human judges, decreases with computed importance scores, which means our definition of the importance score is sensible.

H.3 Similarity Score

For each concept (which we call host concept) in the benchmark, we pick the five most similar concepts and rank them according to the similarity scores. We thereby form 5 groups of similar concepts in decreasing order of the scores. We then ask the judges to manually rate their proximity to the host concepts on a normalized scale from 0 (least similar) to 1 (most similar). We average these judge scores in each of the 5 groups and present the results in Figure 13, which indicates that the similarity scores match the judges' perceptions.

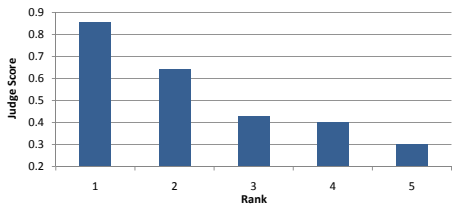


Figure 13: Similarity evaluation results