Attribute Extraction and Scoring: A Probabilistic Approach

Taesung Lee[†1,‡,*] Zhongyuan Wang[#,‡1] Haixun Wang [‡2] Seung-won Hwang[†2]

[†]*POSTECH*
Korea, Republic of

[#]*School of Information,*
Renmin University of China
Beijing, China

[‡]*Microsoft Research Asia*
Beijing, China

{[†1]elca4u, [†2]swhwang}@postech.edu

{[‡1]zhy.wang, [‡2]haixunw}@microsoft.com

***Abstract*—Knowledge bases, which consist of concepts, entities, attributes and relations, are increasingly important in a wide range of applications. We argue that knowledge about attributes (of concepts or entities) plays a critical role in inferencing. In this paper, we propose methods to derive attributes for millions of concepts and we quantify the *typicality* of the attributes with regard to their corresponding concepts. We employ multiple data sources such as web documents, search logs, and existing knowledge bases, and we derive typicality scores for attributes by aggregating different distributions derived from different sources using different methods. To the best of our knowledge, ours is the first approach to integrate concept- and instance-based patterns into probabilistic typicality scores that scale to broad concept space. We have conducted extensive experiments to show the effectiveness of our approach.**

I. Introduction

A fundamental goal in the creation of knowledge bases of concepts, entities, and attributes, is to enable machines to perform certain types of inferences as humans do. The input data is often sparse, noisy, and ambiguous. A human mind is able to make inferences beyond the information in the input data because humans have abstract background knowledge. A knowledge base is used to provide this background knowledge to machines, and thus the ability to acquire, represent, and reason over such knowledge has become the most critical step toward realizing artificial intelligence.

TABLE I
Concepts, attributes, and typicality scores.

Concept	Attribute	$P(c\|a)$	$P(a\|c)$
company	name	0.0401	0.0846
	operating profit	0.9658	0.0218
	...		
country	people	0.5760	0.0694
	population	0.2870	0.0436
	...		
...	...		

A knowledge base consists of a web of relationships among concepts, instances, and attributes. Among these relationships, the following three form the backbone of the knowledge base:

- *isA*: a relationship between a sub-concept and a concept (e.g., an IT company is a company);
- *isInstanceOf*[1]: a relationship between an entity and a concept (e.g., Microsoft is a company);
- *isPropertyOf*: a relationship between an attribute and a concept (e.g., color is a property/attribute of wine).

In this paper, we argue that attributes (i.e., the *isPropertyOf* relationship) play a critical role in knowledge inference. However, in order to perform inference, knowing that a concept can be described by a certain set of attributes is not enough. Instead, we must know how important, or how *typical* each attribute is for the concept. In this paper, we focus on automatically acquiring and scoring attributes. The output of our work is a big table. As shown in Table I, the table consists of (millions of) concepts, their attributes, and scores. The scores are important for inference, and we denote them as the typicality scores. Specifically:

- $P(c|a)$ denotes how typical concept c is, given attribute a.
- $P(a|c)$ denotes how typical attribute a is, given concept c.

In Table I, for example, *company* is not a typical concept for the attribute *name* because virtually every concept has *name*. On the other hand, *company* is a much more typical concept for the attribute *operating profit*. This is quantified by the scores in the table, as we can see

$$P(\textit{company}|\textit{operating profit}) > P(\textit{company}|\textit{name}) \quad (1)$$

On the other hand, when we are talking about a *company*, people are more likely to mention its *name* than its *operating profit*, so we have:

$$P(\textit{operating profit}|\textit{company}) < P(\textit{name}|\textit{company}) \quad (2)$$

But as the data in the table shows, the difference of the typicality scores in Eq 2, 0.06, is not as dramatic as in Eq 1, where the difference is larger than 0.9. This is consistent with how people feel about these attributes and concepts.

Inference

Now we explain why concepts, attributes, and the typicality scores are important in knowledge inference. Intuitively, given

*This work is done at Microsoft Research Asia.

[1]In the rest of this paper, we ignore the difference between *isA* and *isInstanceOf* and denote both of them as *isA*. For any concept, say *company*, we denote its sub-concepts and entities as its instances. Thus, both *IT company* and *IBM* are instances of *company*.

the short text, "capital city, population", people may infer that it relates to *country*. Given "color, body, smell," people may think of *wine*. However, in many cases, the association may not be straightforward, and may require some guesswork even for humans. Consider the following example. Assume that we encounter the table on the web as shown in Fig. 1(a). Can we guess what the table is about?

(a) A web table with a header

	website	president	city	motto	state	type	director

(b) # of possible concepts decreases as more attributes are known

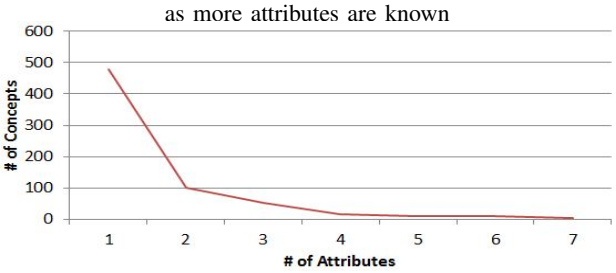

Fig. 1. How we guess what the table is about?

From a single attribute, say *website*, it is impossible to guess the exact content of the table, as *website* belongs to many concepts. However, as illustrated in Fig. 1, as the system we developed sees more and more attributes, it has fewer and fewer candidate concepts, and after 6 or 7 attributes, it finds the right concept with high confidence (the table is about university, institutes, etc.) As we can imagine, typicality scores $P(c|a)$ and $P(a|c)$ can play an important role in the process of deriving the right concept.

As another example, consider the following tweet message:

The Coolpix P7100 is announced. The powerful lens with 7.1x zoom offers high resolution (10MP) images.

Suppose we have no idea that Coolpix P7100 is a camera. Given the tweet, can we derive from its context that it is talking about a camera? As a human being, we probably can. How about machines with a knowledge base? Suppose that through natural language processing, we identify *lens, zoom, resolution* as attributes, and in the knowledge base, only *camera* and *smart phone* have attributes *lens, zoom, resolution*. Then, all we need to figure out is whether the probability $P(\text{camera}|\text{lens}, \text{zoom}, \text{resolution})$ is greater than $P(\text{smart phone}|\text{lens}, \text{zoom}, \text{resolution})$, or vice versa. In other words, we want to know whether *camera* is a more typical topic than *smart phone*, given the set of attributes.

With typicality scores, it is not difficult for a machine to perform the above inference. The goal is to find the most likely concept given a set of attributes. More specifically, we want to find concept $\hat{c}$ such that

$$\hat{c} = \underset{c}{\text{argmax}}\, P(c|A)$$

where $A = (\text{lens}, \text{zoom}, \text{resolution})$ is a sequence of attributes. We can estimate the probability of concepts using a naive Bayes model:

$$P(c|A) = \frac{P(A|c)P(c)}{P(A)} \propto P(c) \cdot \prod_{a \in A} P(a|c)$$

It is clear that this boils down to finding conditional probability $P(a|c)$, which is one of the typicality scores we have mentioned.

Our Contributions

To support the type of inference illustrated by the above examples, we focus on two tasks: *acquiring attributes* and *scoring attributes*. We leverage Probase [1], [2], a probabilistic knowledge base that contains a large number of concepts, entities, isA relationships, etc., in our work.

- To the best of our knowledge, this is the first work that investigates typicality scores for attributes. We have shown that many applications could benefit from knowledge about concepts and attributes with typicality scores. In this work, we consider two views on typicality studied in psychology (*frequency* and *family resemblance*), and reflect them in probabilistic typicality scoring procedure.
- While most existing attribute extraction methods are instance-based, we also adopt a new concept-based approach, and we make in-depth comparisons between them. With concept-based attribute extraction, we directly obtain the attribute *population* for the concept *country*, from the text "the population of a country". Compared with instance-based extraction, concept-based approaches incur much less ambiguity and produce higher-quality attributes.
- We also deal with ambiguity, which is a big challenge, especially for instance-based attribute extraction, but not fully addressed in the previous approaches. For example, while finding attributes for the concept *wine*, from the text "the mayor of Bordeaux", we may mistake *mayor* for an attribute of *wine*. But in fact, 'Bordeaux' is ambiguous, as it is also the name of a city in southwestern France. In our work, we address the ambiguity issue in instance-based attribute extraction, and we also leverage the concept-based approach that is not vulnerable to instance-based ambiguity.
- We leverage attributes from different data sources, and introduce a novel learning to rank approach to combine them. Each data source and method has its own characteristics. For example, we may find that the attribute *name* is frequently observed in one method (i.e., concept-based approach), and not in another (instance-based approach), while the attribute *biography* is just the opposite. Putting them together gives us a more comprehensive and unbiased picture, which resolves issues such as ambiguity, noise, bias, and insufficient coverage, due to the eccentricity of a particular method/source. We will show the difference of the attributes extracted from different data sources, and present a simple yet effective method to aggregate the extraction results from the sources using a learning to

rank method. This is a totally novel approach. An existing work [3] uses a regression method to aggregate features to recognize good attributes. However, it requires human assessors to specify exact numeric values. In contrast, the learning to rank method has no such requirement.

- To have comprehensive and accurate coverage of attributes, we address the challenge of computational feasibility. We process very large text corpus including a web corpus, search logs, and existing knowledge bases such as DBpedia.

Paper Organization

The rest of the paper is organized as follows. Section II introduces our approach for finding attributes for millions of concepts from several sources. Section III shows how to give weights to the extracted attributes for each method, and how to aggregate the weights. We present experimental results in Section IV. Finally, we discuss related work of attribute mining in Section V, and conclude in Section VI.

II. Attribute Extraction

In this section, we introduce a knowledge empowered approach for attribute extraction. We extract (concept, attribute) pairs from multiple data sources. In Section III, we show how to assign a weight or a typicality score for each (concept, attribute) pair for each source, and then describe how to aggregate the weights from different sources into one consistent typicality score.

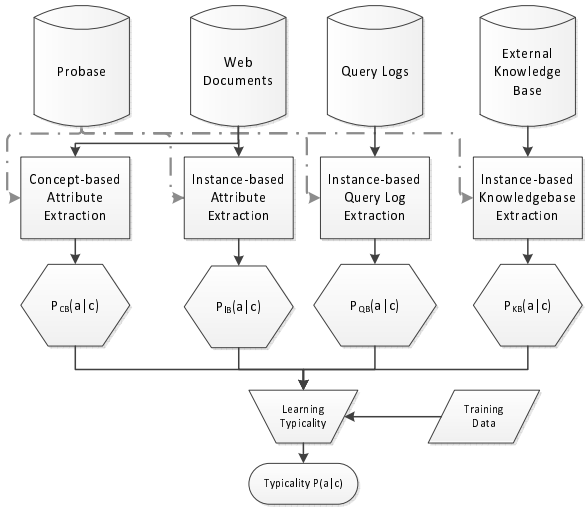

Fig. 2. The framework.

A. A Framework for Attribute Extraction

As we show in Fig. 2, our attribute extraction framework employs a probabilistic knowledge base (Probase) and performs extraction from the three types of data sources. The knowledge base, which we will describe in more detail in Section II-B, enables and guides attribute extraction from different data sources. The types of data we focus on include web data, search log data, and various structured data. We summarize the data in Table II. The web data contains 7.63 billion web documents with a total size of 240 TB. The search log contains all the web search queries with frequency greater than two collected over a six-month period. The structured data we use is DBpedia [4], which we regard simply as a set of (entity, attribute) pairs.

TABLE II
Summary of three types of data sources.

	Web documents	Bing query log	DBpedia
Size	huge (240 TB)	large (9 GB)	relatively small (64.3 MB)
Structure	unstructured, but syntactic	unstructured, not syntactic	structured
Method	concept-based, instance-based	instance-based	instance-based
Extraction Time	38.33 hours	115.33 minutes	-

The algorithms we propose for attribute extraction in this paper generally fall into two categories: the concept-based approach and the instance-based approach. The concept-based approach directly obtains attributes for a concept. The instance-based approach first obtains attributes for instances that belong to the same concept, then it aggregates them to derive attributes for the concept. We apply both methods to our web data. However, for the search log and structured data, only instance-based attributes are available. In Section II-C, we will describe these extraction methods in detail. We will show that attributes derived from different methods or sources have different properties and do not entirely overlap, thus the methods and the sources complement each other.

Each (attribute, concept) pair extracted from a data source is also associated with a weight that indicates how typical the attribute is given the concept. In the final phase, we merge the four probability distributions into one typicality score. The details are presented in Section III-D

B. A Probabilistic isA Network

In our work, we leverage a probabilistic knowledge base called Probase [1] to guide attribute extraction. The goal of Probase is to create a network of isA relationships for all the concepts (of worldly facts) in a human mind. IsA relationships can be obtained by information extraction using the Hearst linguistic patterns [5], which are also known as the SUCH AS patterns. For example, a sentence that contains "... artists such as Pablo Picasso ..." can be considered evidence for the claim that *artist* is a hypernym of *Pablo Picasso*.

Probase has two unique features. First, Probase has large coverage – it contains millions of concepts and entities (both are multi-word expressions) acquired through automatic information extraction from billions of web pages. For instance, it contains not only head concepts such as *country* and *city*, but also tail concepts such as *basic watercolor technique* and *famous wedding dress designer*. This enables Probase to better interpret human communications. Another important feature of Probase is that it is probabilistic. Probase maintains co-occurrence counts for every (*concept*, *sub-concept*), or (*concept*, *entity*) pair. The co-occurrence enables us to compute

typicality scores for isA relations. For example, we have

$$P(instance|concept) = \frac{n(instance, concept)}{\sum_{instance} n(instance, concept)} \quad (3)$$

where an instance denotes either a sub-concept or an entity that has an isA relationship to the concept, and $n(instance, concept)$ denotes the co-occurrence count of *concept* and *instance*. The typicality information is essential for understanding the intent behind a short text.

C. Concept- and Instance-based Extraction

In this section, we focus on attribute extraction using two methods: a concept-based method and an instance-based method. We perform extraction from the web corpus (Section II-C1), an existing knowledge base (Section II-C2), and query logs (Section II-C3). We also analyze the difference of the attributes obtained by the two methods, which demonstrates that both methods are important (Section II-C4). Finally, we introduce some filtering methods to improve the quality of extraction (Section II-C5).

1) Attribute extraction from the web corpus: Many information extraction approaches focus on finding more and more syntactic patterns in an iterative process for extracting a relation. However, this approach has one significant weakness. High quality syntactic patterns are very few. Most patterns produce a lot of noises. In our approach, we focus on the following high quality syntactic patterns for concept-based and instance-based attribute extraction on web corpus to 1)accurately process the documents (high precision), and 2)extract large amount of attributes from huge web corpus (high recall and feasibility).

- Syntactic pattern for **concept-based (CB)** extraction:

$$\text{the } \langle a \rangle \text{ of (the/a/an) } \langle c \rangle \text{ [is]} \quad (4)$$

- Syntactic pattern for **instance-based (IB)** extraction:

$$\text{the } \langle a \rangle \text{ of (the/a/an) } \langle i \rangle \text{ [is]} \quad (5)$$

Here, $\langle a \rangle$ is a target attribute that we want to obtain from texts that match the syntactic patterns, $\langle c \rangle$ is the given concept for which we obtain attributes, and $\langle i \rangle$ is an instance (sub-concept or entity) in concept $\langle c \rangle$. Both $\langle c \rangle$ and $\langle i \rangle$ are from the Probase semantic network. For example, let's assume that we want to find attributes for the concept $\langle c \rangle$ = *wine*. From the sentence, "... the acidity of a wine is an essential component of the wine ...", we know that $\langle a \rangle$ = *acidity* is a candidate attribute of wine. Furthermore, from the sentence "the taste of Bordeaux is ..." we know that $\langle a \rangle$ = *taste* is an attribute of 'Bordeaux'. From Probase, we know that 'Bordeaux' is an instance in the *wine* concept. Thus, $\langle a \rangle$ = *taste* is also a candidate attribute of *wine*.

The result of the extraction is a large set of (concept, attribute) pairs. Besides obtaining the pairs, we also want to know their weights; that is, how typical is an attribute for a concept? The result tuples (c, a) by CB, or (i, a) in the case of IB, of each extraction are grouped into $(c, a, n(c, a))$ tuples, or $(i, a, n(i, a))$ tuples, where $n(c, a)$ is the number of occurrences of $\langle c \rangle$ and $\langle a \rangle$, or $n(i, a)$ for $\langle i \rangle$ and $\langle a \rangle$. We will collect the list of $(c, a, n(c, a))$ tuples or $(i, a, n(i, a))$ tuples from which we will later compute probabilistic attribute scores in Section III.

There are also some implementation details that are worth mentioning. As we need to extract patterns from the web-scale corpus, existing pattern mining techniques requiring POS tagging cannot be used. Instead, we use light-weight extraction using the above patterns. The use of [is] is reported to lead to higher quality extraction in [6], and also facilitates the discovery of the noun phrase boundary of $\langle i \rangle$ and $\langle c \rangle$ without POS tagging which takes a huge amount of time. Another challenge is the use of articles [the/a/an] in the pattern. For the CB pattern, articles are always required in order to filter out attributes describing the concept term itself (e.g., the definition of wine, the plural form of country). For the IB pattern, the use of articles is selective and depends on whether instance $\langle i \rangle$ is a named entity (e.g., Microsoft) or not (e.g., a software company). We distinguish these two cases by considering a capitalized instance as an entity and requiring articles for the rest of non-entity instances.

2) Attribute extraction from an external knowledgebase: We also leverage an existing knowledge base containing entities and their attributes. In our work, we use DBpedia, which is usually based on structured information from Wikipedia. As DBpedia does not have possible concept-based attributes, we use **instance-based attributes from DBpedia entities (KB)**. In DBpedia, from each entity page described with its attributes, we obtain (i, a) tuples. Unfortunately, DBpedia does not have any information to derive how typical an attribute is for an instance, so we treat them equally; we use $n(i, a) = 1$ to produce $(i, a, n(i, a))$ tuples. Note that although we set 1 for all tuples, instances in the same concept have different attributes. Thus we obtain proper typicality after scoring as we do with IB tuple list (Section III-C) (i.e., atypical attributes are not common among instances in the concept).

3) Attribute extraction from the query logs: To extract **instance-based attributes from query logs (QB)**, we utilize the two-pass approach used in [7] to obtain as many tuples as possible. Note that we cannot expect concept-based attributes from query logs as people usually have interest in a more specific instance rather than a general concept (e.g., the query 'company operating profit' is rare compared to 'microsoft operating profit'). In the first pass, we extract, from the query logs, candidate instance-attribute pairs (i, a) from the pattern "the $\langle a \rangle$ of (the/a/an) $\langle i \rangle$", which we denote as A_{QB}. We then extend the candidate list A_{IU}, to include the attributes obtained from IB and KB (A_{IB} and A_{KB} respectively):

$$A_{IU} = A_{IB} \cup A_{KB} \cup A_{QB} \quad (6)$$

Next, in the second pass, we count the co-occurrence $n(i, a)$ of i and a in the query log for each $(i, a) \in A_{IU}$ to produce the $(i, a, n(i, a))$ list. We handle this list in the same way we do for other instance-based list (in Section III-C).

4) Attribute Distribution: Let us now look at the differences of attributes obtained from the CB and the IB lists. To

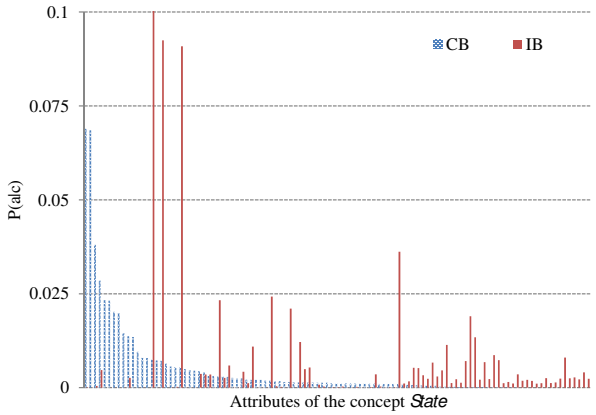

Fig. 3. The attribute distribution of CB and IB on the web corpus.

illustrate, Fig. 3 contrasts the differences of two distributions for *state* concept. For example, the attribute *name* is frequently observed from the CB pattern, such as "the name of a state is...", but people will not mention it with the specific state, as in "the name of Washington is...". From such differences, we observe the complementary strength of the two patterns as below.

The strength of CB attributes is that attributes can be mechanically bound to a concept. For example, from the CB pattern "the population of a state", a machine can naturally bind the attribute *population* to the concept *state*. In contrast, the IB pattern "the population of Washington" is hard to process for a machine, as 'Washington' belongs to both *state* and *president* concepts.

However, despite such difficulty, IB patterns may lead to the harvesting of higher-quality attributes. For instance, although the pattern "the population of a state" is rarely observed, when replacing *state* with its state instances, we can collect enough patterns, such as "the population of Washington", that is frequently observed. That is, IB patterns, by obtaining noisier yet statistically significant observations, complement CB patterns.

5) Pattern Extraction Filtering: Lastly, we discuss how we improve the quality of attributes, by filtering out non-attributes collected from our patterns. Toward this goal, we first categorize the collected results into the following three categories:

- **C1:** The CEO of Microsoft is ... – attributes
- **C2:** The rest of China has been ... – modifying expressions
- **C3:** The University of Chicago is ... – noun phrases containing 'of'

Out of these three categories, **C2** and **C3** would produce noisy results. We thus discuss our filtering rules for these two cases respectively.

C2 – Black List Filtering: To address **C2**, we generate a black list of attributes which can bind to virtually any concept, as follows:

- The lack of vitamin A is ...
- The rest of the country was in ...

TABLE III
TOP ENTRIES IN THE BLACK LIST.

Attribute	# concepts	Attribute	# concepts
meaning	35	best	27
definition	33	nature	26
importance	32	plural	26
rest	27	work	26

- The best of The Simpsons is ...

As the words lack, rest, and best are not specifically describing the concepts that follow, they should not be considered typical attributes. This kind of noisy attributes will have, if not filtered, high $P(a|c)$ which is undesirable.

To filter out such attributes, we identify attributes that are commonly observed in diverse and dissimilar concepts that are expected to share only few attributes. For this purpose, we select a set of 49 dissimilar concepts and rank each attribute by the number of the dissimilar concepts it belongs to, as shown in Table III. However, such a list may include false negatives, such as *name*, which is specific to the given concept yet appears in many concepts. To remove it from the black list, we consider the attribute scoring we will discuss in Section III and remove important attributes with high scores.

C3 – Named entities containing "of": To address **C3**, we filter out "of"-clause associated with named entities, such as "the University of Chicago", "the Bank of China", "the People's Republic of China". As filtering rules, we first consider capitalization, which is a strong feature to classify named entities. As the second rule, we refer to knowledge bases to rule out "of"-clause, which is an instance itself. For example, "the University of Chicago" is an instance in Probase and we thus do not consider *university* to be an attribute of the concept *city*. In this way, we can handle the caseless text such as tweets.

III. ATTRIBUTE SCORING

In this section, we first discuss the intuition of our scoring in Section III-A. We then explain how we process CB and IB (or QB, KB) lists obtained from various sources for computing the typicality scores of attributes in Sections III-B and III-C respectively. Lastly, we discuss how we aggregate the scores from multiple sources in Section III-D.

A. Typicality Scoring

Our goal is to quantify $P(a|c)$ for attribute-concept pairs. This probabilistic score is not only useful for making inferences but it also has psychological significance.

Such a score, known as typicality, has been actively studied in cognitive science and psychology [8] to understand why some instances are typically mentioned by human beings for a concept. According to the literature, a dog is a typical instance of pet as 1) it is frequently mentioned as a pet, and 2) it shares some resemblance [9] to other pet instances. We can extend this intuition for understanding the typicality of attributes, as follows:

An attribute a is typical for a concept c if

- a is frequently mentioned as an attribute of c or its instances. (*frequency*)
- a is common among the instances of c. (*family resemblance*)

With this intuition, *population* is a typical attribute of a *country*, as such pairs are frequently observed in CB and IB lists. Further, *population* is a typical attribute of a *country*, as most country instances, such as China or Germany, share the same attribute *population*.

This intuition can again justify using both CB and IB for quantifying $P(a|c)$. We can observe frequency from both streams and using IB enables us to consider the resemblance across instances. In contrast, existing methods consider either one or none: [7], [10], [11], [12] consider only frequency, while [13] considers only resemblance. [14], [15] consider neither, and instead use contextual similarity of the extracted attributes with the seed attributes given a concept.

In the following sections, we see how we substantialize frequency view from a CB list, and frequency and resemblance view from IB lists.

B. Computing Typicality from a CB List

Recall that a CB list is in the format $(c, a, n(c, a))$. Grouping this list by c, we can obtain a list of attributes observed about c and their frequency distribution. Given this information, typicality score $P(a|c)$ can be straightforwardly obtained by normalization:

$$P(a|c) = \frac{n(c,a)}{\sum_{a^* \in c} n(c,a^*)} \tag{7}$$

C. Computing Typicality from an IB List

We now discuss how we can compute typicality from an IB list in the format $(i, a, n(c, a))$. Note that we obtain three IB lists from web documents, query log, and knowledge base, respectively. As we will discuss in Section III-D, the quality of the three lists varies, depending on concept c. We thus compute typicality from the three IB lists separately, then aggregate three scores with the score computed from the CB list.

To connect IB pattern with a concept, we first expand $P(a|c)$ as follows.

$$P(a|c) = \sum_{i \in c} P(a, i|c) = \sum_{i \in c} P(a|i,c)P(i|c) \tag{8}$$

With this expansion, our goal boils down to computing $P(a|i, c)$ and $P(i|c)$. To illustrate these goals, consider an IB pattern "the age of George Washington". This pattern can contribute to the typicality scoring of *age* for the concept *president*, knowing that 'George Washington' is an instance of the concept *president*. In the formula, $P(a|i, c)$ quantifies the attribute typicality of *age* for 'George Washington' when its underlying concept is *president*, while $P(i|c)$ represents how representative 'George Washington' is for the concept *president*.

Computing Naive $P(a|i, c)$ and $P(i|c)$ using Probase: For computing $P(a|i, c)$ and $P(i|c)$, we can leverage the isA relations of Probase, which stores, for example, how representative 'George Washington' is for the concept *president*. For the sake of presentation, we first compute Naive $P(a|i, c)$ and $P(i|c)$, under a simplifying assumption that one instance belongs to one concept, just to show biases that this computation incurs. We will later discuss how to unbias to relax this assumption.

First, for computing $P(a|i, c)$, under this simplifying assumption, $P(a|i, c) = P(a|i)$, such that

$$P(a|i,c) = P(a|i) = \frac{n(i,a)}{\sum_{a^* \in i} n(i,a^*)} \tag{9}$$

Second, for computing $P(i|c)$, we first reformulate the goal,

$$P(i|c) = \frac{P(c|i)P(i)}{\sum_{i^* \in c} P(c|i^*)P(i^*)} \tag{10}$$

such that we can reformulate our goal as obtaining $P(c|i)$ from Probase. Under our simplifying assumption, $P(c|i)$, representing how likely the concept is for the given instance i, would be $P(c|i) = 1$ if this concept-instance is observed in Probase and 0 otherwise.

However, in reality, the same instance name appears with many concepts due to the following cases:

- **[C1] ambiguous instance related to dissimilar concepts:** 'Washington' can refer to a president and a state, and the typical attributes for the two concepts are significantly dissimilar. Using naive computation leads to the identification of *population* as a typical attribute for the *president* concept.
- **[C2] unambiguous instance related to similar concepts:** Even an unambiguous instance can appear in different contexts. For example, 'George Washington' is a president, patriot, and historical figure. $P(i|c)$ for an ambiguous entity associated with the dissimilar concepts *state* and *president*, should be lower than an unambiguous one related to the multiple related concepts *president* and *patriot*, while naive computation does not consider the similarity of concepts.

Our goal is thus to approximate the unbiased value for $P(a|i, c)$ and $P(c|i)$, to consider both of the above cases.

Unbiasing $P(a|i, c)$ and $P(c|i)$: We now discuss how we unbias $P(a|i, c)$ and $P(c|i)$ to address cases C1 and C2 discussed above.

First, for computing $P(a|i, c)$, if instance i is ambiguous, a high $n(i, a)$ score observed from another concept should not be counted. For instance, even though *population* occurs frequently with 'Washington', in the context of state names, it should not be counted when considering attributes for the concept *president*. For this goal, we introduce join ratio (JR), representing how likely a is associated with c:

$$JR(a,c) = \frac{JC(a,c)}{\max_{a^* \in c} JC(a^*,c)} \tag{11}$$

where $JC(a, c)$ (join count) is defined as the number of instances in c that has attribute a, which quantifies the *family resemblance* of a. Observe that the JR score will be close to 0 for *population* and *president*, as most other president instances,

TABLE IV
$P(i|c)$ FROM EQ. 13 AND EQ. 14.

Instance Name	Eq. 13	Eq. 14
George Washington	0.0178	0.0060
Washington	0.2452	0.0059
Bush	0.0313	0.0230

TABLE V
ATTRIBUTE RANK IN PRESIDENT USING NAIVE AND UNBIASED $P(c|i)$.

Attribute	Naive	Unbiased
Mayor	53	137
Citizens	20	43
Population	9	15
County	17	39
Streets	24	105

such as 'George Bush' is not likely to appear in the pattern of "population of George Bush".

Using this notion, we unbias Eq. 9 into:

$$P(a|i,c) = \frac{n(i,a,c)}{\sum_{a^*} n(i,a^*,c)} \tag{12}$$

where $n(i,a,c) = n(i,a)JR(a,c)$.

Second, computing $P(c|i)$ can be relaxed to consider an entity belonging to multiple concepts, or **C2**, using Probase frequency counts $n_p(c,i)$:

$$P(c|i) = \frac{n_p(c,i)}{\sum_{c*} n_p(c*,i)} \tag{13}$$

However, using this equation would not distinguish **C2** from **C1**. We thus consider the similarity between the two concepts c and c^*, to discount frequency from dissimilar concepts. For computing this similarity, we use Probase, by comparing the instance sets of the two concepts using Jaccard Similarity. With this $sim(c,c')$ score in the range of [0:1], $P(c|i)$ can be computed as:

$$P(c|i) = \frac{n_u(c,i)}{\sum_{c^*} n_u(c^*,i)} \tag{14}$$

where $n_u(c,i) = \sum_{c'} n_p(c',i) sim(c,c')$. In this way, we not only deal with ambiguity problem, but also reflect the two views of typicality: *frequency* and *family resemblance*.

Table IV contrasts $P(i|c)$ from Eq. 13 and 14. Before unbiasing, $P(i|c)$ is overestimated for an ambiguous instance 'Washington', but its value drops significantly after unbiasing. In contrast, the instance 'Bush', even though it is also ambiguous by referring to two presidents, is associated with similar concepts to *president*, and thus should get significantly higher $P(i|c)$ for the *president* concept. Observe that our unbiasing follows this intuition, such that, after unbiasing, $P(i|c)$ of 'George Washington' and 'Bush' outscore that of 'Washington'.

Table V shows how such unbiasing can improve the typicality ranking of attributes for the *president* concept. Before unbiasing, $P(i|c)$ is overestimated for 'Washington', which is also highly likely to belong to the totally different concept *state*. As a side effect, attributes related to states, rank high for *president* concept as well. Unbiasing can significantly lower their ranks, as demonstrated in the table.

D. Typicality Score Aggregation

In the previous sections, we discussed how we compute $P(a|c)$ from one CB list from web documents and three IB lists from web documents, query log, and knowledge base. We notate these four scores as $P_{CB}(a|c)$, $P_{IB}(a|c)$, $P_{QB}(a|c)$, and $P_{KB}(a|c)$ respectively.

Aggregating these scores is non-trivial, as sources have complementary strength over different concepts. In one case, for concept where many instances are ambiguous, scores from IB lists are less reliable. For example, many instances in the *wine* concept, such as 'Bordeaux' and 'Champagne', double as city names and associate with city-specific attributes such as *mayor*. In this example, the reliability of scores from the IB lists is lower. However, in other cases, when a concept can be expanded to a large set of unambiguous instances, the scores from the IB lists are highly reliable. Recall that some attributes such as *population* are frequently discussed in the context of a specific state yet rarely discussed in the pattern of "population of a state".

This observation suggests that no single source can be most reliable for all concepts and thus motivates us to combine scores with complementary strength. Our goal is to automatically adapt the weights, according to concept characteristic, to behave like the most reliable source for all concepts. This unified approach will generalize for a large class of concepts, unlike existing approaches showing strength only for some specific concepts.

More formally, we formulate $P(a|c)$ as:

$$\begin{aligned} P(a|c) = {} & w_{CB}P_{CB}(a|c) + w_{IB}P_{IB}(a|c) \\ & + w_{QB}P_{QB}(a|c) + w_{KB}P_{KB}(a|c) \end{aligned} \tag{15}$$

Our goal is to learn weights for the given concept.

For this task, we take the Ranking SVM approach, with a linear kernel, which is a well-known pairwise learning to rank method. The advantage of using the pairwise ranking approach over a regression approach is that training data does not have to quantify absolute typicality score. While it is hard to give an absolute typicality score for the attribute *population*, one can easily state that *population* is more typical than *picture*. We collect such pairwise comparisons to learn the above weights with respect to the following features representing the characteristics of a concept.

More formally, the weight w_M of source M is a linear combination of f_M^i, where features represent the ambiguity of instances or the statistical significance of patterns.

- f_M^1: $avgModBridgingScore$: Bridging Score [16] measures the ambiguity of an instance i by quantifying whether it belongs to highly dissimilar concept pairs as follows:

$$BridgingScore(i) = \sum_{c_1} \sum_{c_2} P(c_1|i)P(c_2|i)sim(c_1,c_2) \tag{16}$$

Intuitively, this score is low for an ambiguous instance such as 'Washington' associated with dissimilar concepts. We use its variation ModBridgingScore, which we found to be more effective[2].

[2]While the original definition of Bridging Score uses the average of similarities between the concepts containing the instance, we compute mod by dividing the [0, 1] interval into 0.1 scale like [0, 0.1), [0.1, 0.2), ..., [0.9, 1], then find the mod interval, and within the mod interval, we calculate the weighted average.

- f^2_M: $avgP(c|i)$: $P(c|i)$ is low when an instance belongs to many dissimilar concepts and also indicates the ambiguity of an instance.
- f^3_M: $\sum(frequency_M(a))/\#Attribute_M$: When the frequency of attributes, per each attribute, is low, the statistical significance of our observation is low.
- f^4_M: $\sum(frequency_M(a))/\#Instance_P$: When frequency of attributes, per each instance, is low, the statistical significance of our observation is low.
- f^5_M: $Attribute\#_M/\#Instance_P$: When the average number of attributes for each instance in the concept is low, the effectiveness of the extraction strategy for the given concept is low.

Formally, w_M is represented as a linear combination of the five features: $w_M = w^0_M + \sum_k w^k_M f^k_M$. We can expand the original equation as follows.

$$\begin{aligned} P(a|c) =& w^0_{CB}P_{CB}(a|c) + ... + w^5_{CB}f^5_{CB}P_{CB}(a|c) \\ &+ w^0_{IB}P_{IB}(a|c) + ... + w^5_{IB}f^5_{IB}P_{IB}(a|c) \\ &+ w^0_{QB}P_{QB}(a|c) + ... + w^5_{QB}f^5_{QB}P_{QB}(a|c) \\ &+ w^0_{KB}P_{KB}(a|c) + ... + w^5_{KB}f^5_{KB}P_{KB}(a|c) \end{aligned} \tag{17}$$

It can be seen that this is a linear combination of the terms P_{CB}, ..., $f^5_{CB}P_{CB}$, P_{IB}, ..., $f^5_{IB}P_{IB}$, P_{QB}, ..., $f^5_{QB}P_{QB}$, P_{KB}, ..., $f^5_{KB}P_{KB}$ and hence we can learn the coefficients using Ranking SVM with a linear kernel.

Here we must stress that, unlike some existing approaches that require training data for every concept, our aggregation function can be trained by labels for only a few concepts.

E. Attribute Synonym Set

After quantifying the relation between concepts and attributes, we can get a list of attributes for a given concept. However, since we harvest attributes from the Web, people may use different terms to represent the same meaning, such as using *mission* or *goal* to represent *objective.*

Grouping these semantically identical attributes into a synonym set is an important task, without which the three terms would be considered as independent concepts and the typicality for this group of attributes would be diluted over these three terms.

To find potentially synonymous attributes, we can leverage the evidence from Wikipedia [17]. In our work, we consider the following ways to get attribute synonyms:

- Wikipedia Redirects: Some Wikipedia URLs do not have their own page. Accesses to such URLs are redirected to other articles describing the same subject. We use $x_i \rightsquigarrow y_i$ to denote the redirection.
- Wikipedia Internal Links: Links to internal pages are expressed in shorthand by `[[Title | Surface Name]]` in Wikipedia, where `Surface Name` is the anchor text, and the page it links to is titled `Title`. Again, we denote it as $x_i \rightsquigarrow y_i$, where x_i is the anchor text, and y_i is the title.

Using these evidence pairs, we connect synonymous attributes. Then, we take each connected component as a synonymous attribute cluster. Within each cluster, we set the most frequent attribute as a representative attribute of the synonymous attributes.

IV. Experiments

A. Experimental Settings

In this section, we evaluate the quality of attribute scores obtained from $P_{CB}(a|c)$, $P_{IB}(a|c)$, $P_{QB}(a|c)$ and $P_{KB}(a|c)$. The summary of the sources is presented in Table II. For the Web documents, we use a 240TB web snapshot (December 29th, 2011). For the query log data, we use 6 months of Bing query logs from December 28th, 2009 to June 28th, 2010. For the external knowledge base, we use DBpedia 3.4 which is generated from Wikipedia dumps of September 2009. For extraction, we use a distributed system with a cluster of 10,000 servers to extract attribute tuples from the Web documents. In this setting, extraction took approximately two hours for query log and 1.5 days for Web documents, as shown in the table. After extraction, raw data are reduced to the CB and the IB lists with their sizes shown in the table. These lists are then used to compute scores, which took us from 3 minutes to 6.5 hours, using a Windows Server 2003 (Enterprise x64 Edition) with a 2.53Ghz Intel Xeon CPU and 32 GB memory. Summing up, the entire process was done within a couple of days.

The last row in Table VI shows how many (concept, attribute) pairs were found. KB, relying on manually created Wikipedia data, extracts the least amount of attributes, followed by CB. IB and QB extract the most, building on large-scale instances.[3]

B. Evaluation Measures and Baseline

We now discuss how we measure the precision of attribute scoring. As it is non-trivial to obtain ground-truth scoring for attributes, we use human annotators to group attributes into four clusters– very typical, typical, related, and unrelated, in decreasing order of scores. Examples of the human annotations are shown in Table VII. The number of labeled attributes for evaluation is 4846 attributes from 12 concepts.

As evaluation measures, we use the precision of Top-N attributes for the selected concepts:

$$\text{Precision@N} = \frac{\sum_{i=1}^{N} \text{rel}_i}{N} \tag{18}$$

where rel_i is the relevance score of the i-th attribute. For evaluation purposes, we assign relevance score to the four groups as 1, $2/3$, $1/3$, and 0, respectively.

For recall, although measuring absolute value requires ground truth scoring the universe of all possible attributes for the concept, we can approximate this value for relative comparison purposes. Specifically, we consider a set of attributes labeled as the universe and computed the recall with respect to this set. That is:

[3]QB harvests more records than IB from smaller raw data, as IB extracts only the attributes matching our textual patterns, unlike QB extracting all co-occurring pairs using A_{IU}. As query forms rarely follow textual patterns observed in full sentences or capitalized named entities, we apply lenient and case-insensitive extraction rules, which leads to a higher harvesting ratio.

TABLE VI
SUMMARY OF DATA SOURCES.

	$P_{CB}(a\|c)$	$P_{IB}(a\|c)$	$P_{QB}(a\|c)$	$P_{KB}(a\|c)$
Source	Web documents		6 month Bing query log	DBpedia
Raw data size	240TB		9GB	64.3MB
Extraction time	38.33 hours		115.33 minutes	-
Extracted list size	4.75GB		1.35GB	64.3MB
Scoring time	3 min	2 hour	6.5 hour	27 min
Distribution size	157MB	5.75GB	7.16GB	36.9MB
# of records	3.7 million	74 million	141 million	0.4 million

TABLE VII
THE EXAMPLES OF ANNOTATOR LABELS.

Label	Examples	
very typical	country: population president: election	company: products wine: taste
typical	country: history president: speech	company: vision wine: acidity
related	country: northern part company: sic code number president: 100th birthday wine: temperature	
unrelated	country: jews president: hotel	company: whole wine: mayor

$$\text{Recall@N} = \frac{\#\text{ retrieved very typical attributes in top N}}{\#\text{ very typical attributes}} \tag{19}$$

We then discuss a baseline approach to compare with. Among existing algorithms, we consider those applicable to our problem setting of supporting attribute scoring for broad concept space. For this purpose, existing methods requiring the identification of seed attributes for each concept cannot scale and thus cannot be used as a baseline. In addition, the baseline should be light-weighted in order to be applied to web-scale extraction.

The most suitable baseline satisfying all these constraints is Marius Pasca's two methods [13] using Web documents (MWD) and the query logs (MQL) respectively. We thus adopt these two as our baselines. They use a light-weighted pattern extraction strategy on the Web document corpus and the query logs and propose instance-based scoring, quantifying the *family resemblance* of instances.

We compare our proposed framework with these baselines: We train our framework to aggregate attribute scores, using labels from five concepts. After grouping attributes into four groups, we generate pairwise orderings for a pair of attributes in two different groups. For example, *population* $>$ *northern part* for *country*. We used 2059 labeled attributes for training.

C. Precision

Fig. 4 compares the precision of our proposed method, **unified typicality model (UTM)**, with two baselines– MWD using documents and MQL using query logs. To see how the component score from each source contributes, we also show the quality of the attributes obtained from CB, IB, QB, and KB sources. In each chart, the x-axis represents the number of top attributes we consider, and the y-axis is the precision measure introduced in Section IV-B.

We can see that the result of UTM shows consistently good result over all concepts. For example, in the *company* or *wine* concept, UTM gives the highest precision for almost all *N*. In other concepts, UTM closely emulates the winning component score, while each component shows highly inconsistent performances, performing well in one concept and poorly in another.

For example, in the *wine* concept, CB outperforms instance-based approaches, as many instances in *wine* are ambiguous, such as 'Bordeaux' belonging to both *wine* and *city* concepts. However, instance-based approaches excel in *country*, where well-defined unambiguous instances are frequently discussed in the corpus. Meanwhile, KB gives the best results for small N, containing manually provided common statistical information about a *lake*, such as *depth* and *length*, which does not scale for large N. This complementary nature justifies our unified approach.

Marius Pasca's methods show more or less similar trends to our instance-based methods. That is, they work well for popular concepts with unambiguous instances, such as *country*, but poorly for ambiguous ones, including *wine*. For the concept *lake*, both Marius Pasca's and QB performed poorly. As *lake* is not queried frequently, the query log does not contain enough evidence for such concept. For such a concept, leveraging large web corpus to collect more evidences is effective, which explains why our framework works better.

TABLE VIII
THE AVERAGE PRECISIONS AT N OVER THE 12 CONCEPTS

N	1	5	10	20	30	40	50
CB	0.78	0.75	0.74	0.67	0.67	0.65	0.64
IB	0.75	0.64	0.59	0.54	0.50	0.47	0.45
QB	0.50	0.43	0.37	0.32	0.31	0.30	0.31
KB	0.69	0.53	0.49	0.43	0.36	0.31	0.27
MWD	0.67	0.63	0.57	0.52	0.46	0.44	0.41
MQL	0.61	0.53	0.46	0.42	0.40	0.36	0.35
UTM	**0.89**	**0.86**	**0.82**	**0.73**	**0.70**	**0.68**	**0.66**

In summary, while component scores or existing approaches building on limited data sources give inconsistent performance across diverse concepts with different characteristics, the precision of our unified approach UTM closely approximates the best performing component for the given concept. In other words, we can see the average precision of UTM is the highest at all N in Table VIII. Such consistency over a broad concept space is an important property for our problem context.

D. Recall

Table IX and Table X show the recall of the compared methods for selected concepts with different N. We stress again that these values are relative recall for comparison purposes.

The relative strength of the approaches in terms of recall is similar to that observed for precision: UTM again shows consistently high recall over diverse concepts, while other techniques give inconsistent performances. For example, at N=10, CB performs well for the *wine* concept, but poorly

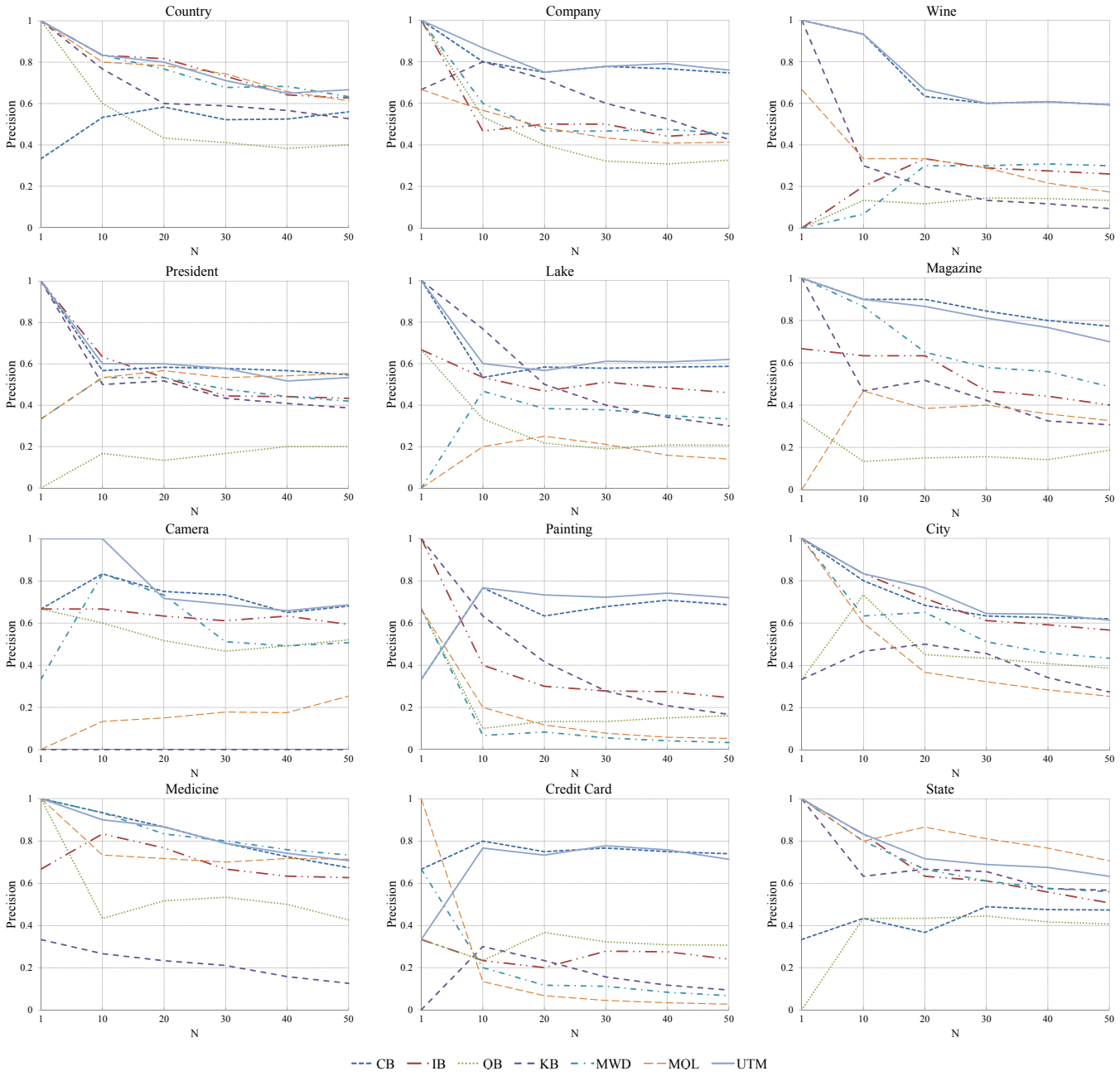

Fig. 4. Precision with respect to N for selected concepts.

for the *lake* concept, as similarly observed in our precision experiments. Other instance-based methods like IB, QB, KB and Marius Pasca's methods do not work relatively well for the *wine* and *company* concepts.

Another thing to observe is the recall over increasing N. For IB, recall increases dramatically for most concepts, from 0.0357 to 0.2857 for *lake*. Such an increase cannot be observed in many concepts for KB, QB, and MQL. This suggests that the use of large-scale web corpus for IB is effective in harvesting good attributes with high recall.

E. Typical Attributes and Concepts

We now qualitatively examine the quality of our results by presenting the most typical attributes identified for those selected in Table XI. In the table, we see that our method extracts high-quality attributes for diverse concepts. For example, for the *battery* concept, our method identifies *capacity* as its most typical attribute. We can observe the effect of unifying multiple sources from attributes of the *building* concept: *location* is rarely discussed for a general building, but frequently discussed for a specific building, while *exterior* is more frequently for a generic building. Our unified approach identifies both as

highly typical attributes for the *building* concept. In contrast, IB only identifies the former and CB only the latter.

We then reverse the direction of inference to infer related concepts from the given attributes. As our scoring function has a probabilistic meaning, we can easily calculate $P(c|a)$ from the result with the aid of Probase. Table XII shows the top typical concepts for the given attribute. For example, for the attribute *duration*, the typical concepts having that attribute are *activity* and *symptom* as well as *course* and *task*. Other interesting results include the concepts identified for *price*, such as *item*, *service*, *resource*, and concepts identified for *temperature*, such as *material* and *beverage*.

Lastly, we observe how our inference results evolve as more attributes are revealed, as illustrated in Fig. 1(b).

For example, when given the single attribute *resolution*, people will associate it with diverse concepts, such as *issues* and *problems*, as shown in Table XIV. However, when the next attribute revealed is *lens*, the associated concepts will converge to optical devices. As more attributes such as *zoom* are given, the top concept converges to cameras.

From the table, we can see the benefit of adopting Probase that contains a wide spectrum of concepts. For example, concepts that are related to *deadline, important dates* includes not only a popular head concept such as *conference*, but also tail concepts such as *stressful situation*.

F. Synonymous Attributes

We now present the quality of the synonymous attribute set we identified using Wikipedia. Table XIII shows the results of the synonymous attributes we identified. In the *country* concept, for example, we have a set of synonymous attributes {*inhabitants*, *population*, *demonym*} whose representative attribute is *population*. We can see that results identified include not only synonym pairs {*gender*, *sex*}, but also the same term in singular/plural forms such as (*flavor*, *flavors*) or spelled differently, such as (*flavor*, *flavour*). These results show the initial promise of identifying synonym sets using a simple technique and Wikipedia, and we leave more sophisticated techniques for future work.

TABLE IX
RECALL AT N=10

Recall@10	Wine	President	Company	Country	Lake
CB	**0.1081**	0.0513	0.0597	0.0333	0.0357
IB	0.0135	**0.0769**	0.0299	0.0556	0.0357
QB	0.0135	0.0256	0.0299	0.0333	0.0000
KB	0.0405	0.0256	0.0746	0.0444	**0.1786**
MWD	0.0000	**0.0769**	0.0299	**0.0778**	0.0357
MQL	0.0135	0.0513	0.0448	0.0667	0.0357
UTM	**0.1081**	**0.0769**	**0.0896**	0.0556	0.1071

TABLE X
RECALL AT N=50

Recall@50	Wine	President	Company	Country	Lake
CB	**0.2703**	0.2564	0.2687	0.1778	**0.5000**
IB	0.1081	0.2564	0.1493	0.2000	0.2857
QB	0.0405	0.0513	0.0746	0.0667	0.0714
KB	0.0541	0.0769	0.1642	0.1000	0.2143
MWD	0.1081	0.1538	0.1493	**0.2444**	0.1429
MQL	0.0541	**0.3077**	0.1045	0.2000	0.0357
UTM	**0.2703**	0.2051	**0.3134**	0.1778	**0.5000**

TABLE XI
EXAMPLES OF TOP TYPICAL ATTRIBUTES FOR A GIVEN CONCEPT.

Concept	Typical Attributes by UTM
battery	capacity, voltage, weight, life, support, lifetime, temperature
painting	size, subject, actual appearance, title, name, quality
country	people, population, capital, history, government, economy
company	name, operating profit, foundation, success, homepage, owner
camera	lens, performance, quality, angle, video, size, resolution, shutter
medicine	benefits, name, side effects, effects, prescription, price, efficacy
illness	cause, course, symptoms, nature, name, source, progression
building	exterior, location, outside, construction, architect, roof, inside
magazine	theme, cover, name, parent company, purpose, title, aim, editor
beverage	ethanol content, taste, temperature, quality, flavor
software	key feature, latest version, installation, name, cost
memory	contents, size, capacity, price, internal write time, output
wine	quality, taste, color, sweetness, name, aroma, texture, flavor

TABLE XII
EXAMPLES OF TOP TYPICAL CONCEPTS FOR A GIVEN ATTRIBUTE.

Attribute	Typical Concepts by UTM
duration	activity, service, symptom, support service, course, task
temperature	material, part, product, person, surface, element, beverage
flavor	vegetable, dish, herb, beverage, ingredient, product
price	item, factor, service, issue, stock, amenity, material, resource
size	person, factor, item, issue, detail, area
developer	application, tool, technology, program, service
location	company, industry, organization, person, facility, resource
population	area, country, city, place, region, community, district

V. RELATED WORK

Although attribute extraction for concepts has been widely studied, existing work do not focus on computing probabilistic scores or scaling over large concept space. Our work is the first to extract the attributes for large concept space with rigorous analysis of the typicality of attributes by leveraging several sources together.

Many works require seed attributes [14], [15], [18] to identify extraction patterns to obtain more attributes, by exploring attributes of the member instances on plain Web documents [18], query logs [14], or structured data [15] including Web tables, lists, and HTML tag hierarchy. However, in our problem scenario, it is infeasible to expect seed attributes to be manually identified for the millions of concepts available on the Web.

TABLE XIII
EXAMPLES OF SYNONYMOUS ATTRIBUTES.

Concept	Representative	Attributes
country	population	inhabitants, population, demonym
person	sex	gender, sex
wine	flavor	flavour, flavor, flavors
wine	color	colour, color
wine	provenance	authenticity, provenance
company	objective	objective, goal, mission, vision, object
company	president	ceo, president, chairman, head
company	operating profit	operating income, operating profit
camera	size	size, dimensions
camera	power supply	power source, power supply
medicine	expiration date	shelf life, expiration date, expiry date

TABLE XIV
EXAMPLES OF TOP TYPICAL CONCEPTS FOR GIVEN A SET OF ATTRIBUTES.

Attributes	Typical Concepts by UTM
resolution	issue, problem, instrument, technical problem
resolution, lens	projector, camera, instrument, device
resolution, lens, zoom	camera, expensive model, consumer application, telescope
president	company, country, organization, institution
president, population	country, company, area, asian country
title	topic, person, country, issue
title, issue	topic, country, magazine, periodical
title, issue, cover	magazine, periodical, topic, document
time	person, event, incident, activity
time, witness	person, crime, event, incident
time, witness, victim	crime, murder, incident, offense
story	person, company, country, outdoor activity
story, cast	TV serial, outdoor activity, cult movie, person
location	company, industry, organization, person
location, production	manufacturer, facility, company, resource
location, production, consumption	resource, service, product, drug
deadline	task, thing, event, credit card
deadline, important dates	conference, educational program, stressful situation, international event
deadline, important dates, tracks	conference, educational program, international event, industry event

Those not requiring seed attributes perform extraction using simple IB patterns, collected from query logs or Web document [13]. However, as we empirically compared in Section IV, work relying on a single data source, performs poorly in some concepts such as *wine* and *credit card*.

More recently, approaches that combine multiple sources have been proposed [10], [11]. Pasca et al [11] use both query logs and query sessions, and [10] combines several structured data sources, including Web tables, search hit counts, DBpedia and Wikipedia. However, they do not discuss how to compute probabilistic scores or systematically aggregate scores from multiple sources.

Attribute extraction work without scoring includes [19], which uses POS tagging, [20], which uses random walk-based attribute label propagation to address sparsity, and [21], which leverages instance-based extraction on web tables. In clear contrast, our proposed framework replaces POS tagging with lighter weight pattern extraction to achieve scalability and addresses sparsity using the knowledge base Probase. Another key distinction of our work is robust quantification of attribute typicality scores using multiple data sources.

Web table-based method [3] quantifies the joint probability of attributes and return, for a given set of attributes, other related attributes. Our work distinguishes itself by addressing the ambiguity and quantifying robust attribute typicality scores. Another distinction is that, unlike their approach using a regression method to quantify a score from attribute features that requires human assessors to give exact score for training, ours do not requires them by using a learning-to-rank method.

Methods heavily based on web tables [3], [10] can extract attributes having simple values – numerical or short text values. However, another category of attributes, such as *history* of a country, may not be described briefly, hence excluded often. Therefore, these approaches are not suitable for our goal of finding wide range of typical attributes in human mind.

VI. CONCLUSION

In this paper, we proposed a framework for extracting attributes from multiple data sources and computing probabilistic scores. Unlike previous instance-based work, we address instance ambiguity and aggregate with concept-based patterns. To the best of our knowledge, this is the first work that unifies instance- and concept-based patterns from multiple sources aggregated into a probabilistic score using a pairwise learning to rank method. In summary, our framework can give both practical and rigorous mathematical attribute typicality.

ACKNOWLEDGMENT

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea and Microsoft Research, under IT/SW Creative research program supervised by the NIPA (National IT Industry Promotion Agency). (NIPA-2012-H0503-12-1036).

REFERENCES

[1] W. Wu, H. Li, H. Wang, and K. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *SIGMOD*, 2012.
[2] Y. Song, H. Wang, Z. Wang, and H. Li, "Short text conceptualization using a probabilistic knowledgebase," in *IJCAI*, 2011.
[3] M. J. Cafarella, E. Wu, A. Halevy, Y. Zhang, and D. Z. Wang, "Webtables: Exploring the power of tables on the web," in *VLDB*, 2008.
[4] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. G. Ives, "Dbpedia: A nucleus for a web of open data," in *ISWC/ASWC*, 2007, pp. 722–735.
[5] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *COLING*, 1992, pp. 539–545.
[6] A. Almuhareb and M. Poesio, "Attribute-based and value-based clustering: An evaluation," in *EMNLP*, 2004, pp. 158–165.
[7] M. Paşca and B. V. Durme, "What you seek is what you get: Extraction of class attributes from query logs," in *IJCAI*, 2007, p. 2832–2837.
[8] G. L. Murphy, *The Big Book of Concepts*. The MIT Press, 2002.
[9] L. Wittgenstein, *Philosophical Investigations*. Prentice Hall, 1999.
[10] A. Kopliku, M. Boughanem, and K. Pinel-Sauvagnat, "Towards a framework for attribute retrieval," in *CIKM*, 2011, pp. 515–524.
[11] M. Pasca, E. Alfonseca, E. Robledo-Arnuncio, R. Martin-Brualla, and K. Hall, "The role of query sessions in extracting instance attributes from web search queries," in *ECIR*, 2010, pp. 62–74.
[12] K. Tokunaga and K. Torisawa, "Automatic discovery of attribute words from web documents," in *IJCNLP*, 2005, pp. 106–118.
[13] M. Paşca, V. D. Benjamin, and N. Garera, "The role of documents vs. queries in extracting class attributes from text," in *CIKM*, 2007, pp. 485–494.
[14] M. Paşca, "Organizing and searching the world wide web of facts–step two," in *WWW*, 2007, p. 101–110.
[15] S. Ravi and M. Paşca, "Using structured text for large-scale attribute extraction," in *CIKM*, 2008, pp. 1183–1192.
[16] K. Onuma, H. Tong, and C. Faloutsos, "Tangent: a novel, 'surprise me', recommendation algorithm," in *SIGKDD*, 2009, pp. 657–666.
[17] T. Lee, Z. Wang, H. Wang, and S. won Hwang, "Web scale taxonomy cleansing," in *VLDB*, 2011.
[18] K. Bellare, P. P. Talukdar, G. Kumaran, F. Pereira, M. Liberman, A. McCallum, and M. Dredze, "Lightly-Supervised Attribute Extraction," in *NIPS*, 2007.
[19] B. Van Durme, T. Qian, and L. Schubert, "Class-driven attribute extraction," in *Coling*, 2008, pp. 921–928.
[20] E. Alfonseca, M. Paşca, and E. Robledo-Arnuncio, "Acquisition of instance attributes via labeled and related instances," in *SIGIR*, 2010, p. 58–65.
[21] M. Yakout, K. Ganjam, K. Chakrabarti, and S. Chaudhuri, "Infogather: entity augmentation and attribute discovery by holistic matching with web tables," in *SIGMOD*, 2012, pp. 97–108.