IBminer: A Text Mining Tool for Constructing and Populating InfoBox Databases and Knowledge Bases

Hamid Mousavi
CSD, UCLA
Los Angeles, USA
hmousavi@cs.ucla.edu

Shi Gao
CSD, UCLA
Los Angeles, USA
gaoshi@cs.ucla.edu

Carlo Zaniolo
CSD, UCLA
Los Angeles, USA
zaniolo@cs.ucla.edu

ABSTRACT

Knowledge bases and structured summaries are playing a crucial role in many applications, such as text summarization, question answering, essay grading, and semantic search. Although, many systems (e.g., DBpedia and YaGo2) provide massive knowledge bases of such summaries, they all suffer from incompleteness, inconsistencies, and inaccuracies. These problems can be addressed and much improved by combining and integrating different knowledge bases, but their very large sizes and their reliance on different terminologies and ontologies make the task very difficult. In this demo, we will demonstrate a system that is achieving good success on this task by: i) employing available interlinks in the current knowledge bases (e.g. *externalLink* and *redirect* links in DBpedia) to combine information on individual entities, and ii) using widely available text corpora (e.g. Wikipedia) and our *IBminer* text-mining system, to generate and verify structured information, and reconcile terminologies across different knowledge bases. We will also demonstrate two tools designed to support the integration process in close collaboration with *IBminer*. The first is the *InfoBox Knowledge-Base Browser* (IBKB) which provides structured summaries and their provenance, and the second is the *InfoBox Editor* (IBE), which is designed to suggest relevant attributes for a user-specified subject, whereby the user can easily improve the knowledge base without requiring any knowledge about the internal terminology of individual systems.

1. INTRODUCTION

Knowledge bases are playing a role of increasing importance in many systems such as text summarization and classification, essay grading, semantic search, and question answering systems. In recent years, several efforts have been devoted to creating such knowledge bases; a prominent example is provided by the structured summaries in Wikipedia, called InfoBoxes, which list important attributes and their values for entities. Similar knowledge bases were created in recent years for specific domains or with a general scope Table 1). Unfortunately, since a manual process is often used to generate structured summaries, and standard ontologies are not always used, the resulting knowledge bases are prone to inconsistency, incorrectness and limited coverage.

Proceedings of the VLDB Endowment, Vol. 6, No. 12
Copyright 2013 VLDB Endowment 2150-8097/13/10... \$ 10.00.

To understand the first issue (low coverage), consider DBpedia [5], which mainly contains the structured part of Wikipedia and most importantly its *InfoBoxes*. An InfoBox can be seen as a set of triples in the form of $<s, \alpha, v>$ which indicates a value (v) for a specific attribute (α) of a given subject (s). Currently, around 43.9% of pages in DBpedia are missing their entire InfoBoxes. Many other pages are also missing part of their InfoBoxes. This mainly indicates DBpedia's coverage is quite incomplete, which directly impacts applications using DBpedia. Other knowledge bases suffer from the same problem as well. The second important challenge in using these knowledge bases is the inconsistency in their knowledge representation. For instance, a single attribute may have several synonyms or aliases (e.g., '*birth date*', '*date of birth*', and '*born*'). While this is a very common incident in DBpedia and many other manually created knowledge bases, automatically reconcile these differences is not a trivial task at all. The third issue most of the current knowledge bases are suffering from is inaccuracy. As our experiments show, more than 3.7% of the summaries provided in DBpedia are incorrect.

To address the aforementioned three issues, we pursue the two-prong approach of: i) integrating different publicly-available knowledge bases (Table 1) to create a more robust knowledge base, and ii) completing and improving this initial knowledge base by using our recently developed text mining tool, called *IBminer* [10]. The key idea for integrating current knowledge bases is to use the existing interlink information for the subjects and convert every piece of information to the triple format used by InfoBoxes. These interlinks are mainly provided by DBpedia through *alias*, *redirect*, *externalLink*, or *sameAs* links. Since many other resources, including YaGo2 and Freebase, link many of their subjects back to those in DBpedia, these link structures are invaluable in our endeavor. However, interlink information only covers a subset of the subjects in current knowledge bases, and in particular attribute mappings are completely missing. To find these missing interlinks or mappings as well as improving the coverage of the initial knowledge base, we use our newly developed *IBminer* system as described next.

IBminer infers attribute synonyms and generates more structured summaries by learning patterns from text (widely available at resources such as Wikipedia) and current structured summaries. To perform this non-trivial task, IBminer first converts free text into graph structures, called TextGraphs, using an NLP-based text mining framework, called *SemScape* [10]. SemScape uses morphological information in the text to capture the categorical, semantic, and grammatical relations between words and terms in the text. Then, *IBminer* uses i) the semantic links in the TextGraphs, ii) categorical information (e.g. provided by Wikipedia and WordNet), and iii) our initial knowledge base, to learn the patterns. In addition to being used to generate structured summaries, these patterns enable

IBminer to automatically build mappings between attribute names used in different knowledge bases. using text mining techniques. In this respect, *IBminer* extends Probase [13] which created a general taxonomy with more than 2.7 million concepts from the textual web documents; however, Probase contains only taxonomical information rather than general structured summaries. Our demonstration will focus on the following contributions:

- The *InfoBox Knowledge-Base Browser* (IBKB), which provides users with structured summaries and their originating sources (provenance) for any given subject in our currently integrated knowledge base. Using IBKB, users can easily provide feedback on significance, correctness, or relevance of each summary item in our knowledge base. Users' feedback can be used for ranking the summaries, and also for improving the performance of both *IBminer* and SemScape.
- The *InfoBox Editor* (IBE) tool, that enables users to create and edit structured summaries without requiring any knowledge about the underlying terminology of the summaries. By mining existing text in Wikipedia (or any user-supplied text) on a subject, IBE generates a structured summary about the subject and suggests possibly missing attributes for which the user may want to provide further information.

As previously mentioned, *IBminer* requires an initial knowledge base to operate. The more complete this initial knowledge base is, the better the quality of the learnt patterns will be. Thus, in Section 2, we explain how these knowledge bases are first integrated at a syntactic level. Then in Section 3, we provide more details on the *IBminer*'s approach to integrate them at the semantic level.Thus, in Section 4, we discuss the key features of IBKB and IBE tools, while their important applications are discussed in Section 5.

2. INTEGRATING KNOWLEDGE BASES

In this section, we explain the initial process of knowledge base collection and integration.

2.1 Data Collection

We will consider several publicly available knowledge bases (Table 1) and integrate them as explained later in this section. Among the introduced knowledge bases, there are some domain specific ones (e.g. MusicBrainz, Geonames, etc.), and some domain independent ones (e.g. DBpedia, YaGo2, etc.). Although pieces of knowledge in these sources are represented in various ways, we represent them in the form of triples <*Subject*, *Attribute*, *Value*> and store them in Apache Cassandra which is designed for handing very large amount of data. We recognize three main types of triples:

InfoBox triples: These triples provide information on a known subject in the subject/attribute/value format. For easing our discussion we refer to these as InfoBoxes (<*J.S. Bach*, *PlaceofBirth*, *Eisenach*>).

Subject/Category triples: They provide the categories that a subject belongs to in the form of *subject/link/category* where, *link* represents a taxonomical relation (<*J.S. Bach*, *isA*, *German Composers*>).

Category/Category triples: They represent taxonomical links between categories(<*German Composers*, *isA*, *German Musicians*>).

Currently, we have converted all the knowledge bases listed in Table 1 into the above triple formats.

2.2 Data Integration

Knowledge bases introduced in Table 1 contain a wealth of knowledge as the numbers indicate. However, lack of a standard ontology in these knowledge bases makes them very challenging to integrate. Our main goal is to tackle this challenge and discover the initial interlinks between subjects, attributes, and categories in order to eliminate duplication, align attributes, and improve consistency.

Name	Size (MB)	# of Entities (10^6)	# of Triples (10^6)
ConceptNet [12]	3075	0.30	1.6
DBpedia [5]	43895	3.77	400
FreeBase [6]	85035	$\approx$25	585
Geonames [1]	2270	8.3	90
MusicBrainz [2]	17665	18.3	$\approx$131
NELL [7]	1369	4.34	50
OpenCyc [3]	240	0.24	2.1
YaGo2 [9]	19859	2.64	124

Table 1: Public Knowledge Bases

Interlinking Subjects: Fortunately, the same names are often used to denote the same subjects in different databases. Moreover, DBPedia is interlinked with many existing knowledge bases, such as YaGo2 and FreeBase, which can serve as a source of subject interlinks. For the knowledge bases which do not provide such interlinks (e.g. NELL), in addition to exact matching, we use synonym matching. Synonyms can be obtained from links such as *redirect* and *sameAs*, WordNet, or using our recently proposed ontology generator OntoHarvester [11].

Interlinking Attributes: Similar to the subject interlinking, we use exact matching and synonym matching for finding initial attribute interlinks. In Section 3.3, we explain how we can find synonym attributes and interlink different aliases for the attribute names used in different knowledge bases.

Interlinking Categories: In addition to exact matching, we compute the similarity of the categories in different knowledge bases based on their instances. Consider two categories c_1 and c_2, and Let $S(c)$ be the set of subjects in category c. The similarity function for categories interlink is defined as $Sim(c_1, c_2) = |S(c_1) \cap S(c_2)|/|S(c_1) \cup S(c_2)|$. If the $Sim(c_1, c_2)$ is greater than a certain threshold, we consider c_1 and c_2 as aliases of each other, which simply means that if the instances of two categories are highly overlapping, they can be assumed to represent the same category.

After retrieving these interlinks, we will merge similar triples based on the retrieved interlinks. Currently this task is finished for Geonames and MusicBrainz and partially done for DBpedia and YaGo2. We are quickly covering more of these two knowledge bases as well as the remaining ones in Table 1. All triples are also assigned with accuracy confidence and frequency values. As explained in [10], more evidence supporting the same piece of information will increase its confidence and frequency values. Observe that the sources from which the triples are generated are also stored for the provenance purposes.

Integrating current knowledge bases using the techniques explained above has several advantages discussed next. A) The integrated knowledge base covers more structured summaries, which results in richer and more confident patterns for *IBminer*. B) More attributes are encountered, since the focus on different sources are different. This also improves *IBminer*'s performance as explained in the next section. C) The use of multiple knowledge bases represents an effective way to validate preexisting structured summaries and those newly generated from text. Using these techniques, we can also evaluate the quality of the textual part of a page.

3. IMPROVING THE KNOWLEDGE BASE USING TEXT MINING

This section explains how *IBminer* uses the text describing a subject (mainly retrieved from Wikipedia) to improve the coverage, consistency, and accuracy of the integrated knowledge base for that subject. For more details, readers are referred to [10]. The key idea is to learn patterns from text that can generate new structured summaries (Subsection 3.1) and use the patterns to find more structured summaries (Subsection 3.2). We then use the same set of patterns to find synonyms for the attribute names in the existing knowledge bases (Subsection 3.3), and finally verify the correctness of some of the existing summaries (Subsection 3.4).

3.1 Learning Patterns

The key idea for learning patterns is to find a mapping between the current structured summaries for each subject in the knowledge base and links in the TextGraphs of its accompanying text (e.g. from Wikipedia page). For instance consider the two semantic triples (links) <*bach*, *was*, *composer*> and <*bach*, *was*, *German*> in the TextGraph generated from the text in the Wikipedia page for *bach*. These triples are generated by the SemScape framework using an NLP-based technique [10]. Obviously the link name *was* should be interpreted differently in these two cases, since the former one is connecting a '*person*' to an '*occupation*', while the latter is between a '*person*' and a '*nationality*'. Now, consider two existing InfoBox items <*bach*, *occupation*, *composer*> and <*bach*, *nationality*, *German*> which respectively match the mentioned triples from the text. These items clearly indicate that the link name *was* in our two triples should be interpreted respectively as *occupation* and *nationality*. Thus we can learn following two patterns from these examples:

- <*cat:Person*, *was*, *cat:Occupation_in_Music*>: *occupation*
- <*cat:Person*, *was*, *cat:German*>: *nationality*

Here the pattern $<c_1, l, c_2>:\alpha$ indicates that the link named l, connecting a subject in category c_1 to an entity or value in category c_2, can be interpreted as the attribute name α. Note that for each triple with a matching InfoBox item, we create several patterns since the subject and the values usually belong to more than one (direct or indirect) categories. Each pattern is also assigned a frequency value, so we can use the most frequent ones for generating new structured summaries and interlinking attributes as respectively explained in next two sections.

3.2 Generating New Structured Summaries

To extract new structured summaries using the generated patterns, for any semantic TextGraph triple, say $<s, l, v>$, we find the matching patterns, such as $<c_s, l, c_v>:\alpha$, where $s \in c_s$ and $v \in c_v$. If one or more such matches exist, considering the most frequent pattern, we generate the new InfoBox triple $<s, \alpha, v>$. *IBminer* also uses a type checking technique to enhance the quality of the generated results [10]. As our preliminary experiments indicate, the accuracy of the generated results is more than 94%.

3.3 Generating Attribute Interlinks

Different knowledge bases use different terminologies for naming their attributes. Even in the same knowledge base, there may be many inconsistent attribute names. For instance in DBpedia, the attribute names '*birthdate*', '*data of birth*', '*born*' are all used to indicate the birthdate of a person, while YaGo2 uses '*wasBornOnDate*' sometimes. Moreover, the attribute name '*born*' is used both for birthdate and for birth place. As explained earlier, attribute interlinking is completely missing from current knowledge bases, where synonyms and anonyms can be the source of significant ambiguities and inconsistence. To address this problem, we use multiple matching patterns for the same TextGraph triples. Formally, if TextGraph triple $<s, l, v>$ matches with patterns $<c_s, l, c_v>:\alpha_1$ and $<c_s, l, c_v>:\alpha_2$ respectively with evidence frequency f_1 and f_2 ($f_1 \geqslant f_2$), we create following potential synonym pattern:

- $<c_s, \alpha_1, c_v>: \alpha_2$

The synonym pattern is also assigned a positive support s_p, a negative support s_n, and an evidence frequency f. Every time new evidence of the same synonym pattern is observed, s_p is increased by f_2, s_n is increased by $f_1 - f_2$, and f is incremented by 1. In simpler words, a synonym pattern indicates that attribute name α_1 from subject category c_s to value category c_v may be also called α_2 with positive support s_p, negative support s_n, and evidence frequency f. Again less frequent patterns and those with low supports are filtered and the rest is used to suggest attribute interlinks for the existing or generated InfoBoxes [10].

3.4 Verifying the Structured Summaries

More evidence for the same piece of information is a good indicator of its correctness. However, the knowledge from different resources is usually represented in different ways. Therefore, *IBminer* uses the synonym patterns introduced earlier to resolve some of these differences. Then, it assigns initial weights to different sources, combines triples from them, and calculates their significance and accuracy based on their initial weights, evidence frequency, the patterns correctness confidence, etc. If the same piece of information is generated from the text, *IBminer* accordingly updates its significance and accuracy.

We also try to find mismatches between items generated by *IBminer* and those were already part of the initial knowledge base. We say two triples $<s_1, l_1, v_1>$ and $<s_2, l_2, v_2>$ mismatch if $s_1{=}s_2$, $v_1{=}v_2$, $l_1{\neq}l_2$, and l_1 and l_2 are not synonyms. These mismatches are reported as incorrect summaries. Our experiments [10] indicate that *IBminer* reports 1.2% of the existing summary items as potential incorrect items, whereas only 57% of those are actually incorrect (a false negative rate that is under $1.2 \times 0.57 = 0.52\%$).

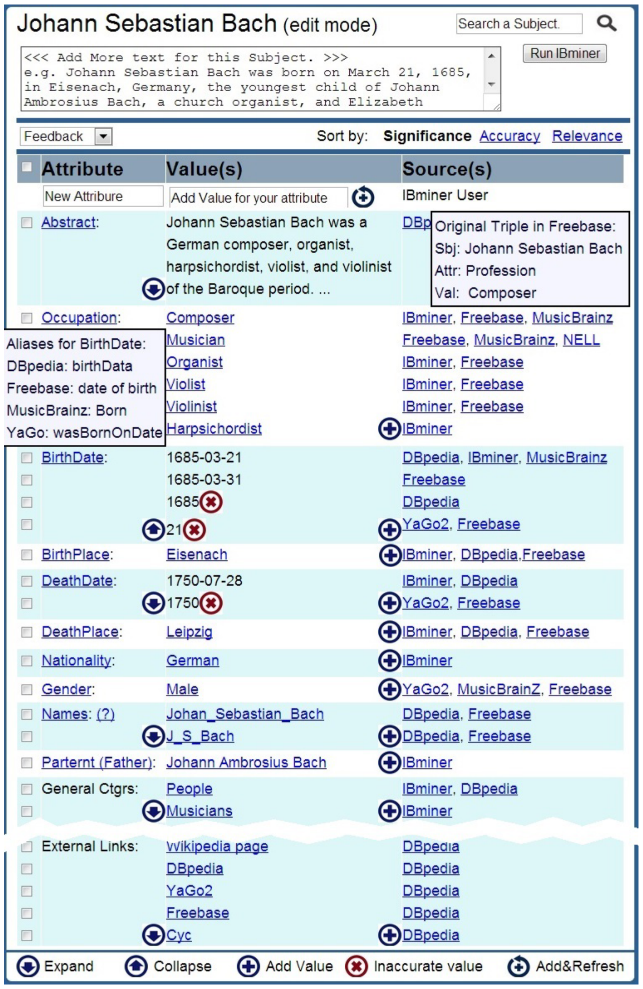
3.5 InfoBox Templates Suggestion

Finding a right InfoBox template (or simply a list of relevant attributes) for a subject can be a very time consuming task for users. To alleviate this issue, *IBminer* (through IBE as explained in the next section) suggests the most relevant attribute names for a given subject. To this end, *IBminer* first finds the most popular attributes currently used for the subjects in each category in our knowledge base. Then, for a subject of interest, all popular attributes from the categories listed for the subject are suggested as relevant attributes. For each attribute, say α_1, we also find the most popular attributes used with α_1 in the same InfoBox (i.e. for the same subject). Based on this co-occurrence data set, we suggest missing attributes which their counterpart is already in the current attributes for the subject. With this feature, users would use more standard attribute names and are more likely to enter structured information.

4. DEMONSTRATION

The following tools and applications were exhibited:

The InfoBox Knowledge-Base Browser (IBKB): IBKB is implemented to let users browse the current knowledge base. Its user interface is very similar to the one for IBE (Figure 1). For a given subject, the tool provides i) structured summary items in user specified order, ii) the synonyms found by *IBminer* for the attributes used in the summaries, and iii) wrong summary items recognized by *IBminer*. The tool can also determine the provenance of each piece of information and report the knowledge base from which it was originally taken, or that it was actually discovered by *IBminer*. By clicking on each source name, the user will be provided with the original form of the triple in that source. Each entity in the result pages is also connected to its own page to make the browsing easier for the users.


Johann Sebastian Bach (edit mode)

Search a Subject.

Run IBminer

```
<<< Add More text for this Subject. >>>
e.g. Johann Sebastian Bach was born on March 21, 1685,
in Eisenach, Germany, the youngest child of Johann
Ambrosius Bach, a church organist, and Elizabeth
```

Feedback — Sort by: **Significance** Accuracy Relevance

Attribute	Value(s)	Source(s)
New Attribure	Add Value for your attribute	IBminer User
Abstract:	Johann Sebastian Bach was a German composer, organist, harpsichordist, violist, and violinist of the Baroque period. ...	DBp
Occupation:	Composer	IBminer, Freebase, MusicBrainz
	Musician	Freebase, MusicBrainz, NELL
	Organist	IBminer, Freebase
	Violist	IBminer, Freebase
	Violinist	IBminer, Freebase
	Harpsichordist	IBminer
BirthDate:	1685-03-21	DBpedia, IBminer, MusicBrainz
	1685-03-31	Freebase
	1685	DBpedia
	21	YaGo2, Freebase
BirthPlace:	Eisenach	IBminer, DBpedia,Freebase
DeathDate:	1750-07-28	IBminer, DBpedia
	1750	YaGo2, Freebase
DeathPlace:	Leipzig	IBminer, DBpedia, Freebase
Nationality:	German	IBminer
Gender:	Male	YaGo2, MusicBrainZ, Freebase
Names: (?)	Johan_Sebastian_Bach	DBpedia, Freebase
	J_S_Bach	DBpedia, Freebase
Partemt (Father):	Johann Ambrosius Bach	IBminer
General Ctgrs:	People	IBminer, DBpedia
	Musicians	IBminer
External Links:	Wikipedia page	DBpedia
	DBpedia	DBpedia
	YaGo2	DBpedia
	Freebase	DBpedia
	Cyc	DBpedia

Original Triple in Freebase:
Sbj: Johann Sebastian Bach
Attr: Profession
Val: Composer

Aliases for BirthDate:
DBpedia: birthData
Freebase: date of birth
MusicBrainz: Born
YaGo: wasBornOnDate

Figure 1: A sample view of the InfoBox Editor (IBE) page. By hovering over the source names (as shown for *Freebase*) user can see the original triple in that source. Similarly, user can see the synonyms used for each attribute by hovering over that attribute name (as shown for *BirthDate*).

Using IBKB, users can select one or more summary items from the user interface, and provide their feedback on the correctness, relevance, and significance of the items. In addition to using such feedback to improve *IBminer*'s performance and to tune its patterns, users' feedback will be used to rank the structured summaries. The ranking mainly improves the user experience with IBKB, since many of the provided summaries are just common sense information for the users, and they usually do not want to see them on top of the list of summaries.

The InfoBox Editor (IBE): In addition to the browsing tool for the current knowledge base, we provide an easy-to-use tool, referred to as IBE, for enhancing the manual process of generating structured information by the users such as in Wikipedia. Figure 1 shows the IBE user interface for the subject *"J.S. Bach"*. For the existing subjects, IBE allows users to add more textual information and structured summaries. To create a new subject, users are asked to enter the name (a descriptive subject), one or more categories for the subject, and a descriptive text for it. They can optionally add as many structured summaries as they desire. The tool suggests a domain, an InfoBox template, and some structured summaries extracted from the entered text. In this way, users can easily edit the summaries and fill the missing spots in the suggested templates without worrying about the underlying attribute names. As a result, the manually generated summaries will follow a more standard terminology; this will improve the quality of the final knowledge base.

Notice that the main difference between IBE and Wikipedia is that IBE's focus is on generating structured summaries for machine use that requires a standard ontology, while Wikipedia is mainly designed for human readers. Moreover, IBE is able to automatically generate structured summaries and suggest InfoBox templates so users can provide structured summaries more efficiently.

5. APPLICATIONS

A very important class of applications that is expected to benefit from our system is semantic-web search systems such as *SWiPE* [4] and Faceted Wikipedia Search [8]. Indeed, the summaries currently provided by many InfoBoxes are such that many queries cannot return correct and complete answers due to the inconsistency and incompleteness of the knowledge base. For instance, the query "*find female musicians in 16th century*" does not provide any answer since the gender attributes are not usually reported in general knowledge bases. However, *IBminer* is able to extract such pieces of information from the text. This will enable applications such as *SWiPE* to provide better query results.

A second set of important applications for our tools is those dealing with textual documents. Document summarization and classification, review summarization, co-reference resolution, and essay/short answer grading systems are only a few examples of such applications. For instance in essay grading, one can use *IBminer* to extract structured summaries from the essay and map them to the ones extracted from the prompt essay. This will move us from current grammar checkers to semantic checkers.

Acknowledgments: This work was supported by the National Science Foundation under grant No. IIS 1118107.

6. REFERENCES

[1] Geonames. `http://www.geonames.org`.
[2] Musicbrainz. `http://musicbrainz.org`.
[3] Opencyc. `http://www.cyc.com/platform/opencyc`.
[4] M. Atzori and C. Zaniolo. Swipe: searching wikipedia by example. In *WWW (Companion Volume)*, pages 309–312, 2012.
[5] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. Dbpedia - a crystallization point for the web of data. *J. Web Sem.*, 7(3):154–165, 2009.
[6] K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.
[7] 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.
[8] R. Hahn, C. Bizer, C. Sahnwaldt, C. Herta, S. Robinson, M. Bürgle, H. Düwiger, and U. Scheel. Faceted wikipedia search. In *BIS*, pages 1–11, 2010.
[9] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artif. Intell.*, 194:28–61, 2013.
[10] H. Mousavi, D. Kerr, M. Iseli, and C. Zaniolo. Deducing infoboxes from unstructured text in wikipedia pages. In *CSD Technical Report #130001), UCLA*, 2013.
[11] H. Mousavi, D. Kerr, M. Iseli, and C. Zaniolo. Ontoharvester: An unsupervised ontology generator from free text. In *CSD Technical Report #130003), UCLA*, 2013.
[12] P. Singh, T. Lin, E. T. Mueller, G. Lim, T. Perkins, and W. L. Zhu. Open mind common sense: Knowledge acquisition from the general public. In *Confederated International Conferences DOA, CoopIS and ODBASE*, London, UK, 2002.
[13] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: a probabilistic taxonomy for text understanding. In *SIGMOD Conference*, pages 481–492, 2012.