Automatic Detection of Outdated Information in Wikipedia Infoboxes

Thong Tran[1] and Tru H. Cao[2]

[1] Da Lat University and John von Neumann Institute - VNUHCM
thongt@dlu.edu.vn

[2] Ho Chi Minh City University of Technology and
John von Neumann Institute - VNUHCM
tru@cse.hcmut.edu.vn

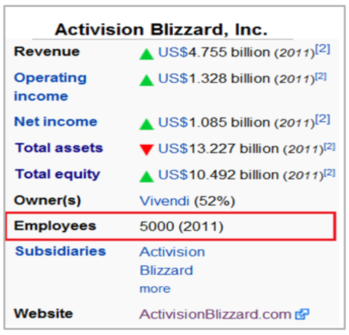
Abstract. An infobox of a Wikipedia article generally contains key facts in the article and is organized as attribute-value pairs. Infoboxes not only allow readers to rapidly gather the most important information about some aspects of the articles in which they appear, but also provide a source for many knowledge bases derived from Wikipedia. However, not all the values of infobox attributes are updated frequently and accurately. In this paper, we propose a method to automatically detect outdated attribute values in Wikipedia infoboxes by using facts extracted from the general Web. Our method uses the pattern-based fact extraction approach. The patterns for fact extraction are automatically learned using a number of available seeds in related Wikipedia infoboxes. We have tested and evaluated our system on a set of 100 well-established companies in the NASDAQ-100 index on their employee numbers, presented by the *num_employees* attribute value in their Wikipedia article infoboxes. The achieved accuracy is 77% and our test result also reveals that 82% of the companies do not have their latest numbers of employees in their Wikipedia article infoboxes.

Keywords: Information Extraction, Wikipedia Update, Pattern Learning.

1 Introduction

Currently, Wikipedia has grown into one of the central knowledge sources of mankind. Since its creation in 2001, Wikipedia has become one of the most popular websites in the world. As of August 2012, the English version of Wikipedia contains almost 4 million articles. The infobox of a Wikipedia article contains important facts about that article that are summarized in the tabulated form. Thanks to such structured information, readers can rapidly gather the most important information about some aspects of the article. Moreover, information in infoboxes can be automatically extracted and reorganized into ontologies, knowledge bases, or databases, such as DBPedia [1], YAGO [2], and Freebase [3].

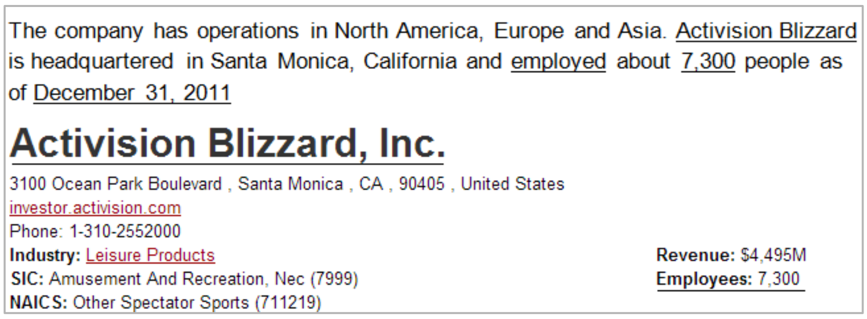
All the Wikipedia's content is manually updated or maintained by contributors. This leads to the fact that its information is not updated regularly and completely, while there is continuous change of related information on the Web. So, it is essential to have automatic methods to update Wikipedia's content to ensure that it contains the latest information. For instance, Fig. 1 is the infobox in the Wikipedia article of company *Activision Blizzard*. It shows that the number of employees of this company is *5,000* in *2011* and this value has not been updated until now[1]. However, the facts from three snippets of web pages[2,3] in Fig. 2 show that its current number of employees has changed to *7,300* since *December 31, 2011*. That means the current value of the number-of-employees attribute in the infobox of *Activision Blizzard* is outdated.



Activision Blizzard, Inc.

Revenue	▲ US$4.755 billion (2011)[2]
Operating income	▲ US$1.328 billion (2011)[2]
Net income	▲ US$1.085 billion (2011)[2]
Total assets	▼ US$13.227 billion (2011)[2]
Total equity	▲ US$10.492 billion (2011)[2]
Owner(s)	Vivendi (52%)
Employees	5000 (2011)
Subsidiaries	Activision Blizzard more
Website	ActivisionBlizzard.com

Fig. 1. The current infobox of company Activision Blizzard in Wikipedia



The company has operations in North America, Europe and Asia. Activision Blizzard is headquartered in Santa Monica, California and employed about 7,300 people as of December 31, 2011

Activision Blizzard, Inc.

3100 Ocean Park Boulevard , Santa Monica , CA , 90405 , United States
investor.activision.com
Phone: 1-310-2552000
Industry: Leisure Products
SIC: Amusement And Recreation, Nec (7999)
NAICS: Other Spectator Sports (711219)

Revenue: $4,495M
Employees: 7,300

Fig. 2. The number of employees of company Activision Blizzard found on the Web

Wikipedia can be enriched using two main sources of information that are Wikipedia articles themselves or external sources such as the Web or some other knowledge bases. In [4], Lange et al. proposed a method for automatically populating infoboxes of Wikipedia articles by extracting unstructured information from the content of their articles. It used Conditional Random Field [5] for training to extract attribute values. Catriple [6] was a system that automatically extracted triples from Wikipedia articles and non-isa properties from Wikipedia categories. Those triples

[1] The time when this paper is completed.
[2] http://www.insideview.com/directory/activision-blizzard-inc
[3] http://www.alacrastore.com/storecontent/MarketLine_formerly_Datamonitor_Company_Profiles-Activision_Blizzard_Inc-2123-27054

could be used to add new attributes of an infobox. Meanwhile, Kylin [7] was a system that created new infoboxes or filled up existing infoboxes of Wikipedia articles by using articles with similar infoboxes to determine common attributes. The attribute values in those infoboxes were then extracted from their containing articles. Using an external source, Syed et al. [8] proposed a method that used information from Wikitolog [9], a hybrid knowledge base of structured and unstructured information, beside existing Wikipedia articles, for adding categories and inter-article links to new Wikipedia articles. Wu et al. [10] extended system Kylin [7] using out-of-Wikipedia information extracted by TextRunner [11]. However, those enrichment methods and systems only focused on adding new data or filling incomplete data to Wikipedia, but not detecting and updating its outdated information.

Besides, our proposed method involves information extraction on the Web, in particular relation extraction. As bootstrap-based systems, DIPRE [12] and Snowball [13] used a seed set of examples of the relation to be extracted for initialization, and then iteratively learned patterns and extracted instances of that relation. With the self-supervised approach, KnowItAll [11], TextRunner [14], and SRES [15] were systems that used only a few labeled examples to learn to extract relations from Web. Mintz et al. [16] investigated an alternative paradigm for automatic relation extraction without requiring labeled corpora, avoiding domain dependence. In contrast to those research works, relation extraction in our proposed method is driven by target outdated infoboxes in Wikipedia.

The remainder of this paper is structured as follows. Section 2 details our proposed method. Section 3 presents experiments and evaluation of the proposed method. Finally, Section 4 concludes the paper and suggests future work.

2 Proposed Method

2.1 Definitions

For clarity, we define the basic notions that are used to present the proposed method.

Definition 2.1: *Fact*
A fact f on a binary relation r, denoted by f_r, is defined to be of the form $<r(e_1, e_2), t>$ where e_1 and e_2 are entities of an instance of r, and t is the associated time of the fact.

We use $f_r.entity1$, $f_r.entity2$, and $f_r.time$ to denote the first entity, the second entity, and the time, respectively, of the fact f_r. For example, with the fact $f_{NumOfEmployees}$ = <*NumOfEmployees*(*Activision Blizzard, 5000*)*, 2011*> extracted from the infobox of company *Activision Blizzard* as shown in Fig. 1, $f_{NumOfEmployees}.entity1$ = *Activision Blizzard,* $f_{NumOfEmployees}.entity2$ = *5,*000, and $f_{NumOfEmployees}.time$ = *2011.* This fact says that the number of employees of company *Activision Blizzard* is *5,000* in *2011*.

Definition 2.2: *Outdated fact*
A fact f is defined to be outdated with respect to another fact g on the same relation r, if and only if:

1. $f_r.entity1 = g_r.entity1$, and
2. $f_r.entity2 \neq g_r.entity2$, and
3. $f_r.time < g_r.time$

Without loss of generality, in the above definition, we assume the outdated information is always at the second entity of a relation of discourse. Also, for an outdated fact in a Wikipedia infobox, the first entity is presumed to be the one represented by the Wikipedia article containing that infobox. For example, the fact $f_{NumOfEmployee}$ = *<NumOfEmployee(Activision Blizzard, 5000), 2011>* in the infobox in Fig. 1 is an outdated fact with respect to the fact $g_{NumOfEmployee}$ = *<NumOfEmployee(Activision Blizzard, 7300), December 31 2011>* in the news pages in Fig. 2 about the number of employees of the mentioned company.

2.2 Processing Steps

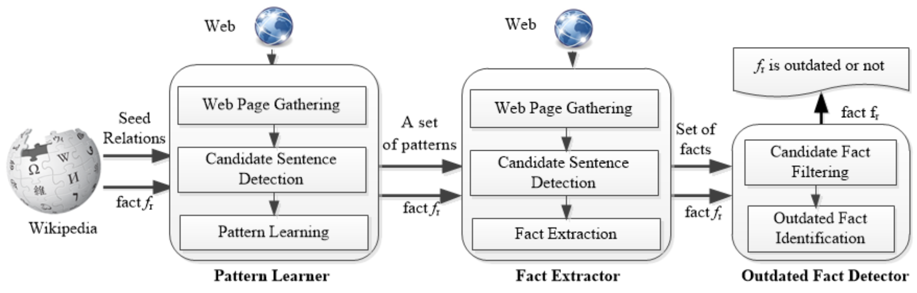


Fig. 3. The architecture of outdated Wikipedia infobox detection system

The architecture of our system to detect outdated information in Wikipedia infoboxes consists of three main components as shown in Fig. 3. The first component, *Pattern Learner,* uses the seed relations obtained from some Wikipedia infoboxes to search for a number of web pages that may contain candidate sentences related to a target relation, which is the relation of the possibly outdated fact to be detected. It then uses those sentences to learn patterns to extract instances of that relation. The second component, *Fact Extractor,* uses the learned patterns to extract those facts that are related to the target relation from the Web. The third component *Outdated Fact Detection* identifies if a fact is outdated or not by matching it with the candidate facts extracted from the Web.

2.3 Pattern Learner

The input of the pattern learning component is a small set of known instances of a target relation r that are automatically extracted from Wikipedia infoboxes. These relation instances are used as the seeds for the pattern learning process of the system. For example, the relation *NumberOfEmployee(Activision Blizzard, 5000),* extracted from the infobox of company *Activision Blizzard* on Wikipedia in Fig. 1, can be used as a seed for learning patterns related to the relation *NumberOfEmployee*. The pattern learning process has the three following steps.

Step 1: Web Page Gathering
From the information of seed instances of the target relation *r*, we use the Google search engine to gather web pages that may contain information of the seeds. The query keywords provided to the search engine are those words that represent the target relation, and the first and the second entities of the seed relation instances. Specifically, we manually build a gazetteer of keywords for each target relation. For each seed relation, the words representing the first entity are extracted from the URL of its corresponding Wikipedia article, while the words representing the second entity are those appear in the infobox containing that seed relation. Then, from the search results, only top-k web pages whose URLs or titles contain information about the first entity are collected.

For example, with the seed relation *NumberOfEmployee*(*Activision Blizzard, 5000*), the relation *NumberOfEmployee* is mapped to the keyword *"employ"*, using the constructed gazetteer. The URL of the Wikipedia article for company *Activision Blizzard* is *http://en.wikipedia.org/ wiki/Activision_Blizzard*. Therefore, the query keywords fed to the Google search engine are "*'Activision Blizzard' employ 5000*".

Step 2: Candidate Sentence Detection
From the search results in the above step, the text of each collected web page is extracted using library HtmlUnit[4]. Next, the text is segmented into a set of sentences using open natural language processing library OpenNLP[5]. Then, from this set, only candidate sentences that contain information about a seed relation are selected. Finally, each candidate sentence is classified into either the complete or incomplete-sentential form. A candidate sentence is considered to be a complete sentence if it has a verb phrase component. Otherwise, it is considered as an incomplete sentence. For example, from Figure 2, one has:

- s_1 = "*Employees: 7,300*" is an incomplete sentence, and
- s_2 = "*Activision Blizzard is headquartered in Santa Monica, California and employed about 7,300 people as of December 31, 2011*" is a complete one.

We also use library OpenNLP for POS tagging to classify sentences.

Step 3: Pattern Learning
Since there are two forms of candidate sentences, we need pattern structures suitable for each of the forms. Construction of patterns for the two forms is presented below.

Patterns for incomplete sentences: Given a seed relation $r(e_1, e_2)$ and an incomplete sentence s = "$w_1\ w_2 ...\ w_n$" where each w_i is a token, the pattern p for s is defined to be

[4] http://htmlunit.sourceforge.net/
[5] http://opennlp.apache.org/

of the form "t_1 t_2...t_n" where each t_i is a slot representing a token or a label related to the seed relation instance. Pattern *p* is built as follows:

- t_i is [*r*] if the keywords for *r* in the gazetteer is a substring of w_i.
- t_i is [`entity2`] if w_i represents the second entity of *r*.
- Otherwise, t_i is w_i.

For example, with the seed relation *NumberOfEmployees*(*Activision Blizzard, 5000*) and the incomplete sentence s_1 above, the pattern for s_1 is p_1 = "[`Number-OfEmployees`]: [`entity2`]".

Algorithm 1: Learning patterns

```
Input: S is a set of seeds on a target relation r
Output: P is a set of patterns
begin
  P ← {}
  for each seed s in S do begin
    keywords ← words representing the first entity of s +
               words representing the relation of s +
               words representing the second entity of s
    webPages ← getTop-k-WebPages(keywords)
  end for
  CS ← {} //CS is the set of candidate sentences
  for each web page w in webPages do begin
    C ← set of candidate sentences in w
    CS ← CS ∪ C
  end for
  for each candidate sentence c in CS do begin
    if (isCompleteSentence(c)) then
      p ← pattern of complete sentence c
    else
      p ← pattern of incomplete sentence c
    end if
    if (p not in P) then
      P ← P ∪ {p}
    end if
  end for
  return P
end
```

Fig. 4. Pattern learning algorithm

Patterns for complete sentences: The pattern construction is similar to that for incomplete sentences, with only one addition that, if w_i represents the first entity of the seed relation, then t_i is [`entity1`]. For example, with the seed relation *NumberOfEm-*

ployees(*Activision Blizzard, 5000*) and the complete sentence s_2 above, the pattern for s_2 is:

p_2 = "`[entity1]` *is headquartered in Santa Monica, California and* `[Number-OfEmployees]` *about* `[entity2]` *people as of December 31, 2011*".

Figure 4 presents our pattern learning algorithm. We note that duplicated patterns are removed in the resulting set of patterns, as shown in code lines 20 to 21 in the algorithm.

2.4 Fact Extractor

Patterns learned in the previous stage are used to extract facts from the Web that are related to the possibly outdated target fact in a Wikipedia infobox. By Definition 2.1, a fact consists of a relation instance and a time. The two following steps are to extract instances of the relation of the target fact. Identification of the associated time of an extracted fact is presented later in Section 2.5.

Algorithm 2: Extracting facts

```
Input:  s is a relation instance of a target relation
        P is a set of patterns
Output: F = {<f_1, freq_1>,...,<f_n, freq_n>} is a set of facts
        with their occurrence frequencies
begin
  F ← {}
  keywords ←  words representing the first entity of s +
              words representing the relation of s
  webPages ←  getTop-k-WebPages(keywords)
  CS ← {} //CS is the set of candidate sentences
  for each web page w in webPages do begin
    C ← set of candidate sentences in w
    CS ← CS ∪ C
  end for
  for each candidate sentence c in CS do begin
    if (isMatchedWithPatterns(c, P)) then
      s ← extracted relation instance from c
      t ← extracted time of s
      freq ← extracted occurrence frequency of s
      F ← F ∪{(<s, t>, freq)}
    end if
  end for
  return F
end
```

Fig. 5. Fact extraction algorithm

Step 1: Web Page Gathering and Candidate Sentence Detection
This step is similar to the first two steps of the pattern learning stage. For the fact extraction stage, the difference is only that the target relation is used instead of a seed relation, and the second entity of the target relation is not used for searching related web pages. For candidate sentences, they need to contain only words representing the target relation and its first entity.

Step 2: Fact Extraction
Applying the learned patterns to the set of candidate sentences obtained from Step 1 above, a set of facts and their frequencies $\{(f_1, freq_1), ..., (f_n, freq_n)\}$ is extracted, where each f_i is a fact extracted from a candidate sentence that matches with some pattern, and $freq_i$ is the occurrence frequency of f_i in the set of web pages returned by the employed search engine. These frequencies are used to rank candidate facts for updating the target fact if it is outdated. For instance, pattern p_1 or pattern p_2 above may be applicable to extract the relation instance <*NumberOfEmployees*(*Activision Blizzard, 7300*)> from a certain related web page. The fact extraction algorithm is presented in Fig. 5.

2.5 Identification of Fact Time

Identification of the associated time of a fact depends on the source from which the fact is extracted. In this work, those sources are Wikipedia infoboxes and the Web. Time phrases in a text are recognized by pre-defined regular expressions.

Facts extracted from Wikipedia infoboxes
If a fact in a Wikipedia infobox contains a time as in Figure 1, then it is used as the associated time of the fact. Otherwise, the associated time of the fact is the time when the attribute value of the fact was added to the Wikipedia infobox. Such a time can be extracted from the updated history of a Wikipedia article. We use open library JWPL[6] (Java Wikipedia Library) to get this time information.

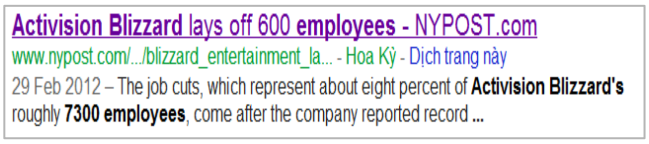

Activision Blizzard lays off 600 employees - NYPOST.com
www.nypost.com/.../blizzard_entertainment_la... - Hoa Kỳ - Dịch trang này
29 Feb 2012 – The job cuts, which represent about eight percent of **Activision Blizzard's** roughly **7300 employees**, come after the company reported record ...

Fig. 6. A Google snippet returned for the query "Activision Blizzard employ"

Facts extracted from the Web
For a fact extracted from a sentence in a web page, if a time phrase is included in the sentence, then it is the associated time for the fact. Otherwise, the publication date of the web page is used as the associated time of the fact.

The publication date of a web page is the time when it was published on the web. We use the snippet of Google search engine for the web page to identify its publica-

[6] http://www.ukp.tu-darmstadt.de/software/jwpl/

tion date, which is a time phrase appearing at the beginning of the snippet, because Google can automatically identify it when crawling websites. For example, given the Google snippet in Fig. 6, the publication date of the corresponding web page is *29 Feb, 2012*. If the snippet does not contain a time phrase, the publication date is identified by analyzing the HTTP response from the employed web server for the corresponding web page.

2.6 Outdated Fact Detection

Given a target fact f_r from a Wikipedia infobox and a set $\{(f_{r1}, freq_1), ..., (f_{rn}, freq_n)\}$ of extracted facts from the Web with occurrence frequencies, f_r is considered to be outdated if and only if it is outdated with respect to some fact f_{ri} in that set, by Definition 2.2. The later fact for f_r with the highest occurrence frequency can be used to update f_r. Our outdated fact detection algorithm is Algorithm 3 presented in Figure 7.

Algorithm 3: Dectecting outdated facts

```
Input:  $f_r$ is a target fact to be checked whether it is out-
        dated
        $F = \{<f_{r1}, freq_1>,...,<f_{rn}, freq_n>\}$ is a set of ex-
        tracted facts with occurrence frequencies
Output: a later fact to which $f_r$ is outdated, or
         NULL otherwise.
1:  begin
2:    $F^* \leftarrow \{\}$ // subset of $F$
3:    for $i$ from 1 to $n$ do begin
4:      if ($f_r$.entity1 = $f_{ri}$.entity1 and $f_r$.entity2 $\neq$
5:          $f_{ri}$.entity2 and $f_r$.time < $f_{ri}$.time) then
6:        $F^* \leftarrow F^* \cup \{<f_{ri}, freq_i>\}$
7:      end if
8:    end for
9:    if (isEmpty($F^*$)) then return NULL   end if
10:   $f_r^* = \text{argmax}_{<f,freq>\in F^*} freq$
11:   return $f_r^*$
12: end
```

Fig. 7. Outdated fact detection algorithm

3 Experiments and Evaluation

To evaluate our proposed method in detecting outdated information in Wikipedia infoboxes, we have selected business companies with their numbers of employees shown in their infoboxes, and the target relation as the number of employees of a company. Our constructed dataset includes 100 well-known companies in NASDAQ-100[7] index. For each company, we manually identified its latest number of employees from the company's official website, or from famous websites about jobs (e.g. glassdoor.com) or business (e.g. businessweek.com).

For pattern learning, seed relations are randomly chosen from the infoboxes of five companies in the dataset. For each seed relation, the top-100 related web pages from the results of the Google engine search are selected, from which patterns are extracted. After eliminating duplicated patterns, there are 7 patterns for extracting incomplete sentences and 15 patterns for extracting complete sentences. Then, for fact extraction for each company in the dataset using the learned patterns, the top-100 related web pages returned by the Google engine search are employed.

We use the accuracy measure based on the numbers of true positive (*TP*), false positive (*FP*), true negative (*TN*) and false negative (*FN*) cases, and calculated by the following formula:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

In our experiments, the meanings of the detected cases are as follows:

- *TP*: the number of employees of a company actually changed and the detection system says it changed.
- *FP*: the number of employees of a company actually changed but the system says it did not change.
- *TN*: the number of employees of a company actually did not change and the system says it did not change.
- *FN*: the number of employees of a company actually did not change but the system says it changed.

Table 1. The evaluation results on 100 companies

Cases	Result
TP	65
FP	17
FN	6
TN	12

Table 1 presents the obtained results with the number of each case. The accuracy of the proposed method is thus 77% (*TP* + *TN*). Besides, it shows that 82 (*TP* + *FP*) out of 100 companies did have their numbers of employees changed but there are no latest numbers in their Wikipedia infoboxes.

[7] http://en.wikipedia.org/wiki/NASDAQ-100

4 Conclusion and Future Work

In this paper we introduce the problem of automatically detecting outdated information in Wikipedia infoboxes. That is challenging and important because Wikipedia has grown fast and become a major information resource, but is still edited manually while the world is changing rapidly.

We have proposed an automatic method for detecting outdated facts in Wikipedia infoboxes with respect to the facts extracted on the Web. It is based on patterns automatically learned from the initiative information in Wikipedia infoboxes of the facts to be detected if being outdated or not. The method also suggests newer facts for updates based on their occurrence frequencies. The experimental results on one particular type of information, which is the number of employees of a company, show a good performance of the proposed method and reveal the striking truth about the outdated status of Wikipedia. The method is however general for arbitrary relations.

There are a number of possible ways to improve the proposed method. First, some machine learning techniques could be employed to find new attribute values in outdated Wikipedia infoboxes. Second, the credibility of an information resource could be taken into account, besides occurrence frequency, to rank and recommend correct and most up-to-date facts for revising Wikipedia. These are among the topics that we are currently working on in this area of research.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: Proceedings of International Semantic Web Conference, Korea. (2007)
2. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A large ontology from wikipedia and wordnet. Elsevier Journal of Web Semantics (2008)
3. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of The 2008 ACM SIGMOD International Conference on Management of Data, New York (2008)
4. Lange, D., Böhm, C., Naumann, F.: Extracting structured information from wikipedia articles to populate infoboxes. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, New York, ACM (2010)
5. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, San Francisco (2001)
6. Liu, Q., Xu, K., Zhang, L., Wang, H., Yu, Y., Pan, Y.: Catriple: Extracting triples from wikipedia categories. In: Proceedings of The 3rd Asian Semantic Web Conference on The Semantic Web, Berlin (2008)
7. Wu, F., Weld, D.S.: Autonomously semantifying wikipedia. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge management, New York, ACM (2007)
8. Syed, Z., Saba, Finin, T.: Approaches for automatically enriching wikipedia. In: Collaboratively-Built Knowledge Sources and AI. Volume WS-10-02. (2010)

9. Syed, Z., Joshi, A.: Wikitology: Using wikipedia as an ontology. Technical report (2008)
10. Wu, F., Hoffmann, R., Weld, D.: Information extraction from wikipedia: Moving down the long tail. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York (2008)
11. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligent (2005)
12. Brin, S.: Extracting patterns and relations from the world wide web. In: International Workshop on The World Wide Web and Databases, London (1999)
13. Agichtein, E., Gravano, L.: Snowball: Extracting relations from large plain-text collections. In: Proceedings of the Fifth ACM Conference on Digital Libraries, New York (2000)
14. Etzioni, O., Banko, M., Soderland, S., Weld, D.S.: Open information extraction from the web. Communication ACM (2008)
15. Rozenfeld, B., Feldman, R.: Self-supervised relation extraction from the web. Knowledge Information System (2008)
16. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of International Joint Conference on Natural Language Processing of the AFNLP, Stroudsburg (2009)