Wikilink – Wikipedia Link Suggestion System, Its Problems and Possible Solutions

Vojtěch Mrkývka

Faculty of Arts, Masaryk University
Arne Nováka 1, 602 00 Brno, Czech Republic
421310@mail.muni.cz

Abstract. In my bachelor thesis I have created a tool, which was able to analyse paragraphs from the given Wikipedia article and suggested the editors internal links, that they could add into this article.
In this paper I return to my thesis and to the tool and evaluate its procedures. I offer ways to solve the problems associated with it, ways which would lead to overall improvement and acceleration of the tool.

Key words: lemmatization, Wikipedia, text mining

1 Introduction

Wikipedia is without doubts the largest encyclopedia in the world [1]. The difference between Wikipedia and its competitors is the flexibility or in the other words possibility to quickly react to current events and discoveries.

Wikipedia is an encyclopedia where anyone can add some information or edit the existing one. But in practice only a tiny percent of the users actively creates new content. Nevertheless there are active efforts to increase the number of editors with regards to conservation of the same or better quality of the content. When I was creating my tool I tried to follow the same principle. I hoped that it would ease the users' editing work and therefore it would increase quality of the target contribution.

2 About Wikipedia

Wikipedia is a free internet encyclopedia, which allow its users to read its content and participate on its development without need to pay any subscription fee or for any licence [2].

It was launched on 15 January 2001 by its founders Jimmy Wales and Larry Singer. Although Wikipedia was meant only as a side-project to encyclopedia Nupedia, which included only articles verified by renowned experts (Wikipedia should have acted like a source of topics and drafts for new articles), its popularity raised rapidly. Today Wikipedia contains more than 5.5 million of articles in its English mutation only. Although the number of visitors is astonishing, the problem is only a small percent of registered users is also helping as editors (by the recent data only about 0.4% edited at least once during the last month) [3].

Aleš Horák, Pavel Rychlý, Adam Rambousek (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2017, pp. 21–26, 2017.

3 Editing Wikipedia

One of the problems associated with the number of editors was the way of editing. Until 2013 the only possibility to create or change any article was to use wikitext, special markup language used for saving Wikipedia articles. This was inconvenient for less technically competent users. Because of it there was a great unbalance between different fields of study. To attract new editors is what Wikipedia Usability Initiative took as its aim. In its five-year-plan for years 2010–2015 proposed creation of new rich-text editor, which would, according to their idea, help to increase the number of Wikipedia contributors to 200,000 (from original number of about 80,000) [4]. In 2012 new editor called VisualEditor was presented in the Wikimedia project. One year later the gradual inclusion of this tool into different language mutations of Wikipedia started [5].

4 Linking Wikipedia

Because of the format of Wikipedia, the internet site, the individual links between Wikipedia articles are made as hypertext links. All of the Wikipedia's links can be separated into three groups. Internal links are those, which are linking pages within Wikipedia or its sister projects (Wiktionary, Wikidata, Wikibooks and others) [6], external links are targeting some external page [7]. Internal links can be further divided to truly-internal, which keeps link within single project and its single language mutation (for example English Wikipedia), and semi-internal, which do not meet the previous criteria.

Insertion of internal and external links is driven by different rules. External links usually connect the article with the source of extended information about the topic. For this reason they are usually situated in reference part of the article and not in the article's body [7]. The internal links on the other hand can help reader to understand the unknown parts of the text by exploring the related topics. They fulfil the role of classic *see something*, which is often present in paper encyclopedias [6]. Even their way of writing is different when the wikitext editation is used.

To keep the article readable, there is a rule on Wikipedia which states that only the first occurrence of some topic in the article's text (information boxes and similar do not count) should be linked also only the articles where is probability to help user with extending knowledge of this articles topic or can bring extending information about it should be linked [6].

5 Wikilink – the Link Suggestion Tool

In my bachelor thesis I designed and created a system, which suggested non-present truly-internal links to the editor of the article. In the beginning I wanted to create an automatic tool (bot), which would insert the links independent on any user. The problem was, create a tool, which would be precise enough to insert only correct links (within some small tolerance), would be task, which

was at the time far superior to my knowledge. Because of this I created the tool, which only suggested possible links to the user and let him to make the final decision.

Wikilink, as I named the tool, consisted of two parts – the client part and the server part. The client part was written in Javascript with jQuery. In the beginning the purpose of the client part was to send data to the server (processing) part and display the results back on the Wikipedia website. Due to the same-origin policy, which prevents AJAX response from another domain, the final version of the client part only send data without receiving any [8]. I used VisualEditor as the source of the data due to the assumption that new users would use it rather than wikitext editing. I avoided editing the result after saving because it would create more unnecessary versions of the article.

Data gained from the client part were processed by the server part. The server part, written in Python[1], removed undesirable elements like information boxes or references. The following processes split individual paragraphs into tokens, lemmatized them and searched for potential links in the reference file.

The reference file was made from Wikipedia backup database dump [9], more specifically from the list of all pages in the main namespace[2] which is generated multiple times every month. This file had to be lemmatized and sorted by lemma for faster functioning.

Output of the tool consisted of the text of every paragraph followed by the table. This table consisted of alphabetically sorted pairs of the link text and the article name to which it should be connected. Because of the first occurrence rule I removed those lines, where the suggestion was done in any of the previous paragraphs.

6 Shortcomings of the First Version

Although the Wikilink was principally working, meaning it suggested new links, which were not present in the article, it contained number of shortcomings, which limited its release and possible wide spreading.

First problem was solely based on user-friendliness. When I was writing my bachelor thesis I wasn't able to integrate the button, which started the whole analysis, into the VisualEditor interface. More specifically I wasn't able to force any Javascript code to start only after the VisualEditor was loaded. Due to this the button to start the Wikilink's process had to be situated outside the editor interface. Because VisualEditor is sometimes loaded without page reload (for example from article itself), the button had to be displayed also on these pages. In this case, the button only displayed special alert message.

Other shortcomings occured in the server part during analysis. Analytic process took lemmatized tokens as an input and tried to find them in the reference list. Output contained two lists, where lemmatized token:

[1] version 2.6

[2] The main namespace contains articles, other namespaces contain portals, help pages, categories and others [10].

- Fully corresponded to lemmatized article name in the reference list (full match)
- Partially corresponded to lemmatized article match in the reference list (partial match)

Union of these lists was the input for the second run of the analytic process. In the second run, the bi-grams were analysed and so on. When the run returned empty result in both of the lists, the cycle of analytic processing stopped. Because of size of the reference list, even the bisection search[3] returned results very slowly (see Table 1). Furthermore I kept links from full match list even if there was found longer string on the same place. I believed, that there are cases, where the link on shorter term can be in the context more specific than on the longer term. In practice, however, I wasn't able to prove this claim, so I think that if there is special case where this concept is right, it occurs so rarely, that there is no need to take it into account.

The last shortcoming was the output format. As I stated before, initially I wanted to display the results back on Wikipedia, ideally right inside the editing window, but due to the same-origin policy, which I wasn't able to overcome at the time, I created special output page where the analysis results were displayed. Thinking of clarity improvement, I ordered the list of results under every paragraph alphabetically, but due to high number of the false results, the solution was far from perfect.

7 Possible Solutions

Shortcomings presented in the previous chapter can be sorted into three overlapping categories – accuracy, overall speed and user-friendliness.

From the perspective of accuracy it is necessary to filter links to the articles which are invalid (or common) by its nature. Example of the very common suggestion can be article *Comma*, to which redirects article name ,. It is very unlikely that any of the Wikipedia articles wouldn't have any punctuation.

[3] using *bisect* package [11]

Table 1. Statistical representation of Wikilink results on random articles. Relevant links column is purely opinion-based, but it can outline the precision of the tool (however links already present in the article are skipped, so the true number would be probably higher).

Article	Word count	Avg. run time	Suggested	Relevant
Bergelmir	89 words	5.45 s	34 links	2 links
Pavel Suchý	420 words	10.44 s	95 links	11 links
SARS	749 words	19.53 s	183 links	59 links
Spolek přátel Rumburku	1,171 words	47.09 s	319 links	34 links
Brno	13,842 words	393.00 s	1,978 links	329 links

Concurrently can be assumed, that only a small number of articles should truly link to the *Comma*. These cases could be removed by application of the simple stop-list, or regular expression. The more complicated are cases where the lemmatization fails and for the specific string is suggested for example some abbreviation. Stop-list could be used to remove the most common mistakes, but not all of them. Because the most of these mistakes link to particular parts of speech and common words the filter could follow them. More specifically the system should filter links made only by preposition, conjunction or one of the verbs *to be* or *to have*.

Considering my previous statement, that shorter strings are usually worse than longer on the same place, the analytic phase of the application can be changed from the multiple runs to the single one. This could improve not only accuracy, but also the overall speed, the second condition for better usability. Speed can be further improved by focusing more on preprocessing of the reference list, which would lower the user requirements. For example the reference list can be rebuild into form of the tree, which can be saved in the JSON format. Crawling the tree would find longer links significantly faster than using the old method.

As for user-friendliness, the previous form of the input and the output was influenced by two mentioned problems – VisualEditor implementation and especially the same-origin policy, which made AJAX requests across different domains impossible. Since my bachelor thesis I found the ways, how to overcome both of these inconveniences. On the website of MediaWiki, whose editing team created the VisualEditor, could be found parts of code and tutorials to operate with editor's interface [12]. The same-origin policy could be evaded by the HTTP header *Access-Control-Allow-Origin* [13]. The results then could be displayed straight inside the VisualEditor interface.

8 Conclusion

Wikilink as a tool did in principle what it was designed for. Although it was far from practical tool due to its speed and accuracy, it outlined some trends which can be followed in further development. In this article I tried to present the problems of the tool with possible solutions, but they are only some of the improvements by which can be the tool enhanced. There are other possibilities which can be explored, for example separate AJAX request, which would suggest possible links continuously. But the aspects of further improvements must be evaluated before any judgements.

Acknowledgments. This work was supported by the project of specific research *Čeština v jednotě synchronie a diachronie* (Czech language in unity of synchrony and diachrony; project no. MUNI/A/0915/2016).

References

1. Wikipedia: Wikipedia:size comparisons — wikipedia, the free encyclopedia (2017) [Online; accessed 29-October-2017].
2. Wikipedia: Wikipedia:about — wikipedia, the free encyclopedia (2017) [Online; accessed 29-October-2017].
3. Wikipedia: Special:statistics — wikipedia, the free encyclopedia (2017) [Online; accessed 29-October-2017].
4. Strategic Planning: Product whitepaper — strategic planning, (2011) [Online; accessed 29-October-2017].
5. MediaWiki: Visualeditor — mediawiki, the free wiki engine (2017) [Online; accessed 29-October-2017].
6. Wikipedia: Wikipedia:manual of style/linking — wikipedia, the free encyclopedia (2017) [Online; accessed 27-October-2017].
7. Wikipedia: Wikipedia:external links — wikipedia, the free encyclopedia (2017) [Online; accessed 27-October-2017].
8. Mozilla Foundation: Same-origin policy - web security | mdn (2017) [Online; accessed 20-October-2017].
9. Wikimedia Foundation: Wikimedia downloads (2017) [Online; accessed 20-October-2017].
10. Wikipedia: Wikipedia:namespace — wikipedia, the free encyclopedia (2017) [Online; accessed 20-October-2017].
11. Python Software Foundation: 8.5. bisect — array bisection algorithm — python 2.7.14 documentation (2017) [Online; accessed 20-October-2017].
12. MediaWiki: Visualeditor/gadgets — mediawiki, the free wiki engine (2017) [Online; accessed 20-October-2017].
13. Mozilla Foundation: Cross-origin resource sharing (cors) - http | mdn (2017) [Online; accessed 20-October-2017].
14. Mrkývka, V.: Návrh chybějících interních odkazů v české wikipedii (2016) [Online; accessed 31-October-2017].