Exposing Digital Content as Linked Data, and Linking them using StoryBlink

Ben De Meester, Tom De Nies, Laurens De Vocht,
Ruben Verborgh, Erik Mannens, and Rik Van de Walle

Ghent University – iMinds – Multimedia Lab, Belgium
{firstname.lastname}@ugent.be

Abstract. Digital publications host a large amount of data that currently is not harvested, due to its unstructured nature. However, manually annotating these publications is tedious. Current tools that automatically analyze unstructured text are too fine-grained for larger amounts of text such as books. A workable machine-interpretable version of larger bodies of text is thus necessary. In this paper, we suggest a workflow to automatically create and publish a machine-interpretable version of digital publications as linked data via DBpedia Spotlight. Furthermore, we make use of the Everything is Connected Engine on top of this published linked data to link digital publications using a Web application dubbed "StoryBlink". StoryBlink shows the added value of publishing machine-interpretable content of unstructured digital publications by finding relevant books that are connected to selected classic works. Currently, the time to find a connecting path can be quite long, but this can be overcome by using caching mechanisms, and the relevancy of found paths can be improved by better denoising the DBpedia Spotlight results, or by using alternative disambiguation engines.

Keywords: EPUB 3, DBpedia Spotlight, Linked Data, NIF

Introduction

Digital publications such as digital books (e-books) and online articles house a vast amount of content. Meanwhile, the amount of published content is rising more than ever, due to advancements in communication technologies. This situation leads to a (harmful) information overload for end users [8].

Current systems for finding relevant publications are largely based on two systems [2]: social recommendation, and (content-based) recommendation based on metadata. On the one hand, social recommendation systems are usually biased towards popular publications, making the unpopular publications harder to find, and making these publications being read less (the so-called *long tail* effect [1]). Metadata-based recommendation systems on the other hand rely on the availability of high-quality tags, yet these tags currently need to be entered manually, making this process very tedious and costly.

The contribution of this paper is twofold. First, we describe a system for the automatic creation of relevant tags using Natural Language Processing (NLP) techniques. Second, we apply these tags to a proof-of-concept that detects publications linked between two publications that are relevant based on their content.

In Section 1, we will review the state of the art and introduce some important relevant technologies. Afterwards, we present the two contributions in Section 2 and Section 3 respectively. These contributions are evaluated in Section 4, after which we conclude in Section 5.

1 Background

In this section, we first have a look at systems that create machine-interpretable representations of natural language text, and link these representations to the Semantic Web (Subsection 1.1). Second, we review the used and relevant technologies for creating our proof-of-concept (Subsection 1.2).

1.1 Semantic Natural Language Processing

To link entities to Semantic data sources, these entities first need to be recognized in the text through Named Entity Recognition (NER). Then, a second analysis is needed to disambiguate these recognized entities into places, people, or other types (Named Entity Disambiguation or NED) [10]. NERD [13], AGDISTIS [15] and DBpedia Spotlight [7] are examples of recognition and disambiguation engines that also connect the detected concepts to their URI on `http://dbpedia.org`. SHELDON [12] is an information extraction framework – accessible via its web interface – that can represent outcomes from multiple NLP domains (e.g., Part-Of-Speech tagging, sentiment analysis, and relation extraction) as linked data. The core of SHELDON is FRED [11], which describes natural language sentences as linked data graphs. SHELDON can use different NLP mechanisms to extract extra information based on this graph. This linking from natural language to linked data sources is the key difference between conventional NLP tools and Semantic NLP tools, which enables publishing (parts of) natural language text as linked data. Also, the latter two systems are open-source, whereas the other engines are closed-source or commercial products.

GERBIL[16] is a general entity annotator benchmarking framework to compare the performance of different annotation engines. Although DBpedia Spotlight is not the most performant engine according to GERBIL, the fact that it can be installed locally makes this a very useful engine to annotate large corpusses of text. Also, DBpedia Spotlight can perform entity recognition, whereas AGDISTIS is solely a disambiguation engine.

1.2 Underlying Technologies

Digital books Digital books are usually distributed in a format so that they can be read offline. EPUB, version 3 [3] is a packaging format created by the *International Digital Publishing Forum* (IDPF). Under the hood, it is a zipped package of a folder with content files (i.e., HTML, but also images and videos for

```
file:///book.epub#epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/1:10)
```

Listing 1. Example identifier for an EPUB file to identify – from right to left – the tenth character (`:10`) of the text (`/1`) of the fifth paragraph (`/10[para05]`) of the body (`/4[body01]`) of the first chapter (`/6/4[chap01ref]`) of the file `book.epub`.

example), together with a package file that manages the links to these content files. What makes it stand out from other distribution formats, is that it is an open format, that makes use of the Open Web Platform technologies.

A notable part of the EPUB specification is the specification of the EPUB Canonical Fragment Identifier (CFI) [14]. This identifier allows the specification of any content within an EPUB package – albeit a range of text or a DOM element – and it is used to create a URI fragment that uniquely defines this piece of content. It uses the XML structure of the manifest file and the HTML-files to follow a slash delimited path. It uses even numbers (n) to go into the $\frac{n}{2}^{th}$ child element, and uneven numbers to go into the text node of that child element. To improve robustness, the ids of the DOM elements may be added between square brackets (`[id]`). Listing 1 shows an EPUB CFI that identifies the tenth character of the fifth paragraph of the first chapter of the file `book.epub`.

Project Gutenberg Project Gutenberg[1] is an effort into digitizing print books that are in the public domain. It currently houses over 49,000 free e-books in various formats, including plain text, HTML, and EPUB. The sample content used in this paper has been downloaded from the Project Gutenberg website.

The NIF and ITS Ontologies The NLP Interchange Format, version 2.0 (NIF) is an RDF format that provides interoperability between NLP tools, language resources and annotations [6]. Although it is a very extensive vocabulary, we will only need the ontology to link publications with their parts that have a recognized entity. NIF is being actively used in several European projects[2]. To connect the NLP results to links on DBpedia semantically, we used the Internationalization Tag Set, version 2.0 (ITS). ITS is a vocabulary[3] to foster the automated processing of Web content [5].

Triple Pattern Fragments Triple Pattern Fragments are a way of hosting and querying linked data in an affordable and reliable way [17]. By moving the complex query execution logic from the server to the client, and letting the server only answer in terms of simple patterns, the amount of needed server power is decreased greatly. This results in a much lower cost and an increased uptime

[1] `http://www.gutenberg.org/`
[2] e.g., LIDER and FREME (`http://www.lider-project.eu/` and `http://www.freme-project.eu/`, respectively).
[3] Also published as an ontology on `http://www.w3.org/2005/11/its/rdf#`.

when hosting such servers compared to standard SPARQL endpoints[4]. Because of the reliability of Triple Pattern Fragments servers, linked data applications can be built on top of live endpoints, whereas the uptime of other query endpoints is often not reliable enough to be used as back-end for linked data applications.

Everything is Connected The Everything is Connected engine (EiCE) is a path finding engine [4]. Given two semantic concepts, it returns paths between them using a linked data endpoint. Links between two nodes are found using configurable heuristics. In this paper, we chose for linking nodes when they share non-trivial relations. For example, a non-trivial relation is that both nodes have their birthplace in Paris, a trivial relation is that both nodes are of the type `Person`. EiCE looks for these links directly between the two given concepts, or indirectly using intermediate concepts. The found paths are weighted based on the amount of common relations per link, and length of the path.

2 From Books to Linked Data

Our overall goal is to provide a linked data endpoint that houses a relevant view of the content of a digital publication. To this end, we use DBpedia Spotlight [7] to extract and disambiguate the concepts in plain text (i.e., perform NER and NED), and link these concepts to their DBpedia URIs. However, two problems arise when using DBpedia Spotlight for an EPUB file, namely that (i) an EPUB file usually consists of multiple HTML files, whilst DBpedia Spotlight works with plain text, and (ii) DBpedia Spotlight has a practical limit of the maximal amount of characters that can be analyzed within one run[5].

To solve these problems, we extract the text from the HTML files in the EPUB file, in reading order. Important to note is that correct handling of whitespace is important, as stripping the HTML tags could result in wrongly concatenated words, which in turn would result in wrong results from DBpedia Spotlight. Listing 2 shows how naively stripping the tags of a (valid) HTML document can introduce errors. Namely, the list $[f, i, n, l]$ is wrongly concatenated with the word *and*, which results in the word *finland*, changing the original text *You have following letters [f, i, n, l] and all are part of the latin alphabet* to *You have following letters finland all are part of the latin alphabet*. For the stripped sentence, DBpedia Spotlight will return "Finland" and "Latin alphabet" as disambiguated results, of which "Finland" is wrongly identified. Thus, depending on the context (i.e., whether it is phrasing content or only flow content), additional whitespace should be added to the original HTML or not[6].

After all text is extracted from the EPUB file, this text is split up into text parts manageable by DBpedia Spotlight. If we wouldn't split up the text, the

[4] 99.999% uptime between November 2014 and February 2015 [18]

[5] `https://github.com/dbpedia-spotlight/dbpedia-spotlight/issues/72`

[6] See `http://www.w3.org/TR/html5/dom.html#kinds-of-content` for the types of content in HTML5, and see `https://github.com/bjdmeest/node-html-to-text` for a possible HTML-to-text implementation.

```
<p>You have following letters
  <ul><li>f</li><li>i</li><li>n</li><li>l</li></ul>and
  all are part of the latin alphabet.
</p>
```

after stripping:

```
You have following letters
  finland
  all are part of the latin alphabet.
```

Listing 2. Whitespace should be taken into account when stripping the tags of an HTML document

DBpedia Spotlight instance would need to analyze the entire text of a publication in one run, which could result in either server response time-outs or out-of-memory errors. As DBpedia Spotlight only takes limited contextualization into account, this splitting up in sentences will have little to no effect on the results of the NER and NED done by DBpedia Spotlight. For this paper, we split the text minimally on sentence boundaries, with a maximal substring length of 1500[7].

The "annotate"-function of DBpedia Spotlight – which recognizes entities to annotate and returns a single identifier for each recognized entity – returns a list of identifiers, which are sorted in order of appearance. We use this order to reconnect these results with the original HTML files. By comparing the original text of the detected concept with the words in the text nodes in HTML, in order, we can reconstruct the range of the original detection. We then generate the EPUB CFI of this range and use this CFI in the semantic representation of the content of the publication.

By making use of the NIF and ITS ontologies, we use well-defined publicly available ontologies that are already being used in business environments. In our case, we used `nif:sourceURL` on the one hand to indicate the link between the detected range and the original book this range originated from, and `itsrdf:taIdentRef` on the other hand to indicate the DBpedia link that was detected from the text in this range (see Listing 3 for an abbreviated resulting semantic representation).

3 StoryBlink

Based on the previously described methodology, we have provided an automatic way of describing the content of (digital) publications as linked data. This linked data is not meant to be granular, i.e., describe every sentence in the publication,

[7] For each following chunk of text, we split off the next 1500 characters, and look for the last occurrence of a sentence boundary, which is the last dot (`.`), question mark (`?`) or exclamation mark (`!`). If no match is found, the full 1500 characters are used as next chunk of text.

```
@prefix schema: <http://schema.org/> .
@prefix nif: <http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-
    core#> .
@prefix itsrdf: <http://www.w3.org/2005/11/its/rdf#> .
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

<http://www.gutenberg.org/ebooks/84.epub#book> a schema:Book .

<http://www.gutenberg.org/ebooks/84.epub#epubcfi(/6/12!/4/2/4)>
    itsrdf:taIdentRef dbr:Chamois ;
    nif:sourceUrl <http://www.gutenberg.org/ebooks/84.epub> .
<http://www.gutenberg.org/ebooks/84.epub#epubcfi(/6/2!/4/46[chap01
    ]/16/42)>
    itsrdf:taIdentRef dbr:Chamois ;
    nif:sourceUrl <http://www.gutenberg.org/ebooks/84.epub> .

<http://www.gutenberg.org/ebooks/84.epub#epubcfi(/6/12!/4/2/6)>
    itsrdf:taIdentRef dbr:Desert ;
    nif:sourceUrl <http://www.gutenberg.org/ebooks/84.epub> .
...
```

Listing 3. Sample abbreviated output of the entity extraction method that links the Gutenberg books with their detected concepts using the ITS and NIF ontologies.

but it is meant to provide a high-level overview of the relevant concepts of a publication in an automatic and machine-interpretable way.

This extracted information is published using a Triple Pattern Fragments server[8], from hereon called the *books endpoint*. Using the HTML interface of the books endpoint, you can explore the mentioned concepts per book. We use this published linked data set in our proof-of-concept: StoryBlink. Through StoryBlink, we enable the discovery of stories by linking books based on their content.

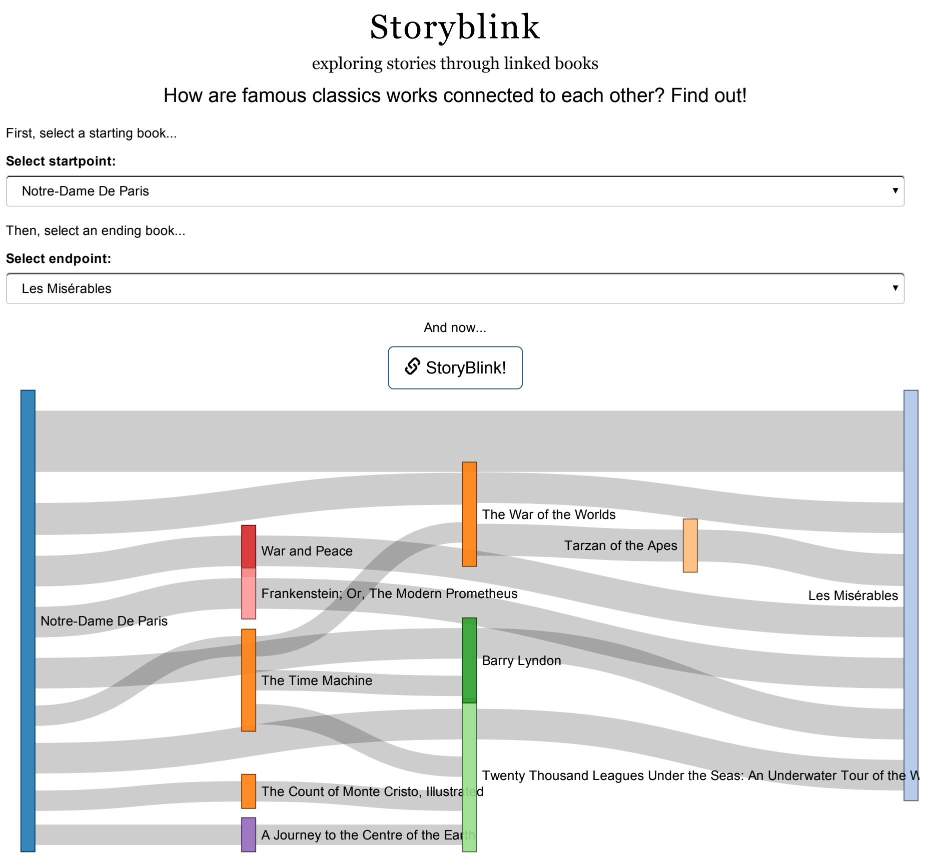
StoryBlink is a Web application that allows users to find relevant books based on two books selected by the user. EiCE finds the paths between these two books, where the nodes are related books, and the links are built using the common concepts as detected in the books.

3.1 Web Application

The data to feed the Web application was extracted from twenty classic books as found on Project Gutenberg. The resulted linked data was published as the books endpoint, used as endpoint for the Everything is Connected engine.

When opening StoryBlink[9], the user needs to choose two books as endpoints. The Everything is Connected engine will then use the books endpoint to look for a path of books between the first book and the second book (Figure 1).

[8] `http://uvdt.test.iminds.be/storyblinkdata/books`
[9] `http://uvdt.test.iminds.be/storyblink/`



Storyblink

exploring stories through linked books

How are famous classics works connected to each other? Find out!

First, select a starting book...

Select startpoint:

Notre-Dame De Paris

Then, select an ending book...

Select endpoint:

Les Misérables

And now...

StoryBlink!
Notre-Dame De Paris
War and Peace
Frankenstein; Or, The Modern Prometheus
The Time Machine
The Count of Monte Cristo, Illustra d
A Journey to the Centre of the Ear
The War of the Worlds
Tarzan of the Apes
Barry Lyndon
Twenty Thousand Leagues Under the Seas: An Underwater Tour of the W
Les Misérables

Fig. 1. The StoryBlink Web application allows a user to find relevant links between two digital classic works.

The advantage of using the EiCE is that this engine aims at finding relevant yet surprising links between concepts. This way, StoryBlink returns content-based recommendations based on these books, without returning too obvious results. For example, recommending a publication because it is also of the type `Book` is not the desired result. The (semantic) commonalities between two linked books can easily be found using a SPARQL query as shown in Listing 4, and is also visualized in StoryBlink when clicking on a link between two books. These visualized commonalities allow the user to personally assess the relevancy between two books on a content level, e.g., one user can assess a book to be relevant because it mentions the same locations, whilst another user can assess another book to be relevant because it mentions the same religion.

```
SELECT DISTINCT * {
    </book1> ?predicate1 ?object .
    </book2> ?predicate2 ?object .
}
```

Listing 4. SPARQL query to find the commonalities between two books.

```
@prefix schema: <http://schema.org/> .
@prefix nif: <http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-
    core#> .
@prefix itsrdf: <http://www.w3.org/2005/11/its/rdf#> .
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix pg84: <http://www.gutenberg.org/ebooks/84.epub#> .

pg84:book a schema:Book .
pg84:book itsrdf:taIdentRef dbr:Chamois,
                            dbr:Desert,
                            ...
```

Listing 5. The extracted linked data can greatly be reduced.

3.2 Improving the Performance

The books endpoint as described above introduces performance issues, not only in reasoning time, but also in quality of the results: the EiCE returns a lot of paths, and takes too much time to do it. This has two causes: the fact that the data model uses a two-step link between a book and its concepts, and the fact that a lot of unimportant concepts are taken into account.

Two-step link The data model, as discussed in Section 2, invokes a two-step link between a book and its detected concepts. First, the book has a certain range, as defined by its CFI, and second, this CFI references to a certain concept in DBpedia. However, when we want to provide a high-level overview of a digital publication, we do not care where in the book the concept was detected, but just that it was detected. We can thus cut the data model short as shown Listing 5. This not only greatly reduces the amount of needed triples, it also enables a direct link between a book and its detected concepts.

Keeping all detected concepts There are two issues with keeping all detected concepts within a book in the data set, namely that (i) the data set grows larger, and (ii) concepts that are detected only a few times are irrelevant for the high-level overview of a book.

As the data set grows larger, the performance of the Everything is Connected engine becomes more troublesome. Indeed, when more data is available, the amount of potential paths found increases, and thus also the searching time of the path finding algorithm. And as this large data set contains a lot of irrelevant concepts, the path finding algorithm also returns a lot more irrelevant paths.

To improve on this, we only keep the top $X\%$ of mentioned concepts. As the initial analysis results kept all references of a detected concept, we can easily count the amount of mentions of a concept, and use those counts to only keep the most mentioned concepts. We keep the top 50% of mentioned concepts, as this is the ideal compromise between execution time and found paths (see Section 4).

Furthermore, we remove all Project Gutenberg specific mentions. As these books all originate from Project Gutenberg, they all have an identical disclaimer at the beginning. Thus, all books have the concept `Project_Gutenberg` detected. However, this link is irrelevant to the content of the book, which is why we remove all Project Gutenberg mentions out of the database.

The aformentioned two optimizations have been implemented in the final proof-of-concept, running at `http://uvdt.test.iminds.be/storyblink/`.

4 Evaluation

The proposed methodology allows for automatic analysis and extraction of relevant metadata, alleviating the need for manual annotations. The Everything is Connected engine uses these semantic annotations with no preference for popular or unpopular works, and thus avoids the long tail effect. However, the performance of StoryBlink is poor when taking into account all mentioned concepts, which is why we propose to only keep the top $X\%$ of all mentioned concepts. To find a good cutoff value, we evaluate the path finding algorithm in terms of time and amount of paths found, whilst varying this cutoff value (Figure 2)[10]. Given that the amount of triples per cutoff value rises exponentially, it is no surprise that the computation time also rises exponentially. In fact, the correlation between amount of triples taken into consideration and average path finding time is 99.45%. The number of paths found saturates around the 50% cutoff value.

When we would keep all detected concepts, we see how the path finding algorithm reaches the $60s$ timeout value. This results in a calculation time of $60s$ with 0 paths found, which is why we see a clear decrease in found paths when we keep all detected concepts in the books endpoint. In Figure 2, we can see how the graph has a noticeable breakpoint at the 50% mark. We also see that, after that mark, there is very little gain in the amount of found paths. If we compare the maximum of found paths with the amount of found paths at the 50% mark, we see how 94.06% of all potential paths can be found in about one eight of the time, i.e., $5.28s$.

Taking into account the large correlation between path finding time and amount of triples in the data set, we can calculate the linear regression between these variables, i.e., $executionTime(ms) = 2.2195 \cdot triples + 2194.7$. Given that in the test data set there were on average 54.55 triples per book, we can compute that we can take into account at most 64 books when trying to find a path with

[10] Windows 7 SP1 64bit, Intel i5-3340M CPU@2.70GHz, 8GB RAM, 256GB SSD.

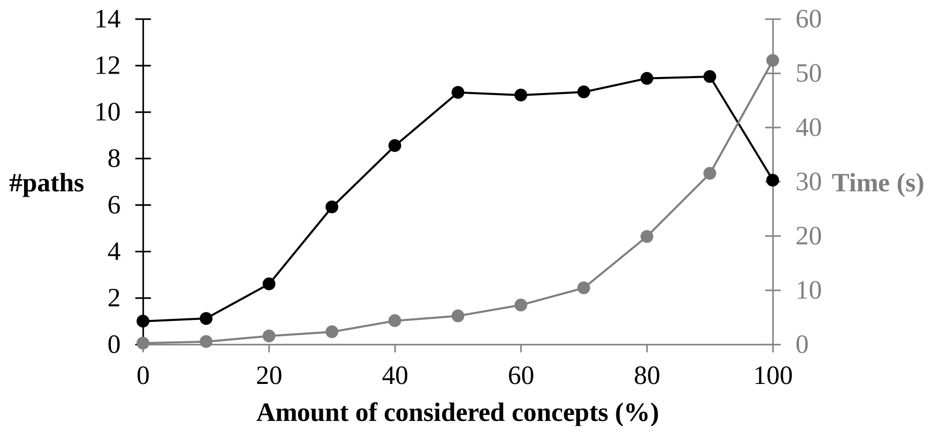


Fig. 2. When evaluating how much detected concepts to keep in the books endpoint, we see a saturation when we only take into account 50% of the mentioned concepts. The average path finding time for that cutoff value is $5.28s$, and 94.06% of the maximal amount of found paths is still found. We see a clear decrease in found paths for 100% of the considered concepts, as keeping all detected concepts introduces application timeouts.

a response time lower than $10s$. This can be done by, e.g., picking 62 books at random out of a catalog besides the two already selected books. However, these calculations do not need to be done for every request, as the data set is a static data source. Therefore, caching (part of) the responses could influence the response times greatly, thus allowing StoryBlink to take more books into account.

5 Conclusions and Future Work

By using engines such as DBpedia Spotlight, it is possible to extract detected concepts from a digital publication, and as such, provide a high-level overview from the content of this publication. In this paper, we presented our methodology to achieve an automatic extraction and publication of detected concepts of a (digital) publication.

However, taking into account every detected concept harms the resulting data set, as unrelevant concepts are also taken into account. By extracting a simplified subset of all detection results, we were able to fuel a content-based recommendation system based on the Everything is Connected engine. This engine can connect digital publications with each other by trying to find relevant yet surprising paths between two concepts. The result is a novel way of content recommendation, no longer tied to social recommendation systems, thus avoiding the long tail effect.

The path finding algorithm takes on average 5.28s to find all relevant books for a data set of 20 books, which will make our Web application usable for book catalogs of maximally 64 books. It is clear that this is not ready for real-life catalogs. However, this problem can easily be resolved by caching the results. As the data set is fixed and not prone to a lot of change, this is a possible solution.

This work could be further improved as following.

- The choice of keeping the top 50% of mentioned concepts biases the books endpoint towards common concepts. Future work is to compare this method with similar metrics that do not bias towards common concepts, such as term frequency-inverse document frequency (tf-idf).
- Instead of selecting two books to find a paths between them, StoryBlink could be adapted to start from only one book . This way, StoryBlink can become a recommendation system, suggesting linked books based on a starting book. This implies making changes to the Everything is Connected engine.
- DBpedia Spotlight could be replaced by other NER and NED engines, to see how this influences the perceived quality of StoryBlink. We can evaluate how using, e.g., Babelfy [9], affects the results of StoryBlink.
- As the Triple Pattern Fragments clients allows for federated querying, we can expand the usage of StoryBlink by finding links across data sets, e.g., books and movies, or books and music.

Acknowledgements The research activities described in this paper were funded by Ghent University, iMinds, the IWT Flanders, the FWO-Flanders, and the European Union, in the context of the project "Uitgeverij van de Toekomst" (Publisher of the Future).

References

1. Anderson, C.: The Long Tail: Why the Future of Business is Selling Less of More. Hyperion (July 2006)
2. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowledge-Based Systems 46, 109–132 (Jul 2013), `http://www.sciencedirect.com/science/article/pii/S0950705113001044`
3. Conboy, G., Garrish, M., Gylling, M., McCoy, W., Makoto, M., Weck, D.: EPUB 3 Overview. Tech. rep., International Digital Publishing Forum (IDPF) (June 2014), `http://www.idpf.org/epub/301/spec/epub-overview.html`, accessed January 22nd, 2015
4. De Vocht, L., Coppens, S., Verborgh, R., Vander Sande, M., Mannens, E., Van de Walle, R.: Discovering meaningful connections between resources in the Web of Data. In: Bizer, C., Heath, T., Berners-Lee, T., Hausenblas, M., Auer, S. (eds.) Linked Data on the Web (LDOW). pp. 1–8. CEUR, Rio De Janeiro, Brazil (May 2013), `http://ceur-ws.org/Vol-996/papers/ldow2013-paper-04.pdf`
5. Filip, D., McCance, S., Lewis, D., Lieske, C., Lommel, A., Kosek, J., Sasaki, F., Savourel, Y.: Internationalization Tag Set (ITS) version 2.0. Tech. rep., W3C (October 2013), `http://www.w3.org/TR/its20/`, accessed June 16th, 2015

6. Hellman, S.: Nif 2.0 core ontology. Tech. rep., AKSW, University Leipzig (2015), http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core/nif-core.html, accessed June 16th, 2015
7. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: DBpedia Spotlight: Shedding light on the Web of documents. In: Proceedings of the 7th International Conference on Semantic Systems. pp. 1–8. ACM (2011)
8. Misra, S., Stokols, D.: Psychological and health outcomes of perceived information overload. Environment and Behavior 44(6), 737–759 (Nov 2012), http://eab.sagepub.com/content/44/6/737
9. Moro, A., Raganato, A., Navigli, R.: Entity Linking meets Word Sense Disambiguation: a Unified Approach. Transactions of the Association for Computational Linguistics (TACL) 2, 231–244 (2014), http://wwwusers.di.uniroma1.it/~navigli/pubs/TACL_2014_Babelfy.pdf
10. Nadeau, D., Sekine, S.: A survey of Named Entity Recognition and Classification. Lingvisticae Investigationes 30(1), 3–26 (2007), http://dx.doi.org/10.1075/li.30.1.03nad
11. Presutti, V., Draicchio, F., Gangemi, A.: Knowledge extraction based on discourse representation theory and linguistic frames. In: Knowledge Engineering and Knowledge Management. pp. 114–129. Springer (2012)
12. Reforgiato Recupero, D., Nuzzolese, A.G., Consoli, S., Presutti, V., Mongiovì, M., Peroni, S.: Extracting knowledge from text using SHELDON, a Semantic Holistic framEwork for LinkeD ONtology data. In: Proceedings of the 24th International Conference on World Wide Web Companion. pp. 235–238. International World Wide Web Conferences Steering Committee (2015)
13. Rizzo, G., Troncy, R.: NERD: Evaluating Named Entity Recognition tools in the Web of Data. In: Workshop on Web Scale Knowledge Extraction, ISWC2011. Bonn, Germany (October 2011)
14. Sorotokin, P., Conboy, G., Duga, B., Rivlin, J., Beaver, D., Ballard, K., Fettes, A., Weck, D.: EPUB Canonical Fragment Identifier (epubcfi) Specification. Tech. rep., International Digital Publishing Forum (IDPF) (June 2014), http://www.idpf.org/epub/linking/cfi/epub-cfi.html, accessed June 16th, 2015
15. Usbeck, R., Ngonga Ngomo, A.C., Auer, S., Gerber, D., Both, A.: AGDISTIS - graph-based disambiguation of Named Entities using Linked Data. In: International Semantic Web Conference. Springer (2014), http://svn.aksw.org/papers/2014/ISWC_AGDISTIS/public.pdf
16. Usbeck, R., Röder, M., Ngonga Ngomo, A.C., Baron, C., Both, A., Brümmer, M., Ceccarelli, D., Cornolti, M., Cherix, D., Eickmann, B., et al.: GERBIL: General entity annotator benchmarking framework. In: Proceedings of the 24th International Conference on World Wide Web. pp. 1133–1143. International World Wide Web Conferences Steering Committee (2015), http://svn.aksw.org/papers/2015/WWW_GERBIL/public.pdf
17. Verborgh, R., Hartig, O., De Meester, B., Haesendonck, G., De Vocht, L., Vander Sande, M., Cyganiak, R., Colpaert, P., Mannens, E., Van de Walle, R.: Querying datasets on the Web with high availability. In: International Semantic Web Conference 2014. pp. 180–196. Springer (2014)
18. Verborgh, R., Mannens, E., Van de Walle, R.: Initial usage analysis of DBpedia's Triple Pattern Fragments. In: Proceedings of the 5th USEWOD Workshop on Usage Analysis and the Web of Data (Jun 2015), http://linkeddatafragments.org/publications/usewod2015.pdf