Time-travel Translator: Automatically Contextualizing News Articles

Nam Khanh Tran, Andrea Ceroni, Nattiya Kanhabua, and Claudia Niederée

L3S Research Center / Leibniz Universität Hannover, Germany

{ntran, ceroni, kanhabua, niederee}@L3S.de

ABSTRACT

Fully understanding an older news article requires context knowledge from the time of article creation. Finding information about such context is a tedious and time-consuming task, which distracts the reader. Simple contextualization via Wikification is not sufficient here. The retrieved context information has to be time-aware, concise (not full Wiki pages) and focused on the coherence of the article topic. In this paper, we present Contextualizer, a web-based system that acquires additional information for supporting interpretations of a news article of interest that requires a mapping, in this case, a kind of time-travel translation between present context knowledge and context knowledge at time of text creation. For a given article, the system provides a GUI that allows users to highlight their interested keywords which are then used to construct appropriate queries for retrieving contextualization candidates. Contextualizer exploits different kinds of information such as temporal similarity and textual complementarity to re-rank the candidates and presents to users in a friendly and interactive web-based interface.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Time-aware contextualization; Complementarity; Temporal context

1. INTRODUCTION

Reading a current news article about the topic that you are familiar with is typically straightforward but things get worse if the article is from the past. In order to understand the article properly, acquiring context knowledge from the time of article creation is required. We call this process as *time-aware contextualization*. As an example, consider the advertisement poster from the 1950s in Figure 1. From today's perspective it is more than surprising that it would be actually doctors, who recommend smoking. It can, however, be understood from the context information at the right side of Figure 1, which has been extracted from the Wikipedia article on tobacco advertising.

Contextualization

In 1954, tobacco companies ran the ad "A Frank Statement." The ad was the first in a campaign to dispute reports that smoking cigarettes could cause lung cancer and had other dangerous health effects.

Prior to 1964, many of the cigarette companies advertised their brand by falsely claiming that their product did not have serious health risks. A couple of examples would be "Play safe with Philip Morris" and "More doctors smoke Camels". Such claims were made both to increase the sales of their product and to combat the increasing public knowledge of smoking's negative health effects.

Figure 1: Camel advertisement (left) and contextualization information taken from Wikipedia (right).

Basic forms of contextualization have already been suggested in early works such as [2, 5] by adding information, which is related to the entities and concepts mentioned in the text. In our example, these techniques can produce the Wikipedia article on the mention "Camels" (camels_cigarette), "Doctors" (medical_doctors) but obtaining the contextualization information as shown in Figure 1 is beyond their scope. In addition, the context information should be digestible in a short time with minimal disruption from the main reading. Thus, we aim for a contextualization unit granularity which is considerably smaller than a full Wikipedia page (e.g, paragraphs).

Therefore, time-aware contextualization, which aims to associate an information item d with time-aware, coherent context information c for easing its understanding, is a challenging task. We address several properties of the context c [7]: (1) c has to be relevant for d, (2) c has to complement the information already available in d, (3) c has to consider the time of creation (or reference) of d.

In this paper, we present a system, called Contextualizer, that provides a solution for the time-aware contextualization problem, implementing a general architecture that automatically retrieves context from user-highlighted keywords which we called *contextualization hooks* and presents them to the user in a meaningful way. To this end, we first construct

WWW '2015 Florence, Italy
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2742841.

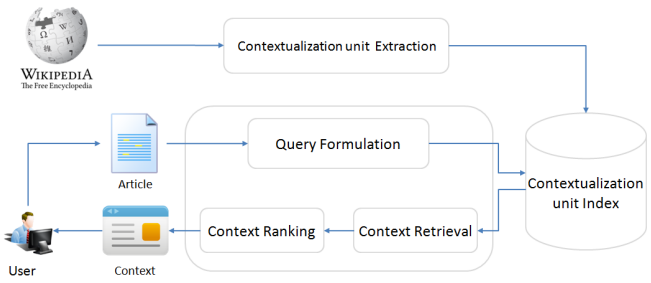

Figure 2: The Contextualizer Architecture.

appropriate queries from either the document itself or the contextualization hooks to retrieve contextualization candidates and in the next step re-rank the results by exploiting different types of information such as temporal similarity and textual complementarity. To the best of our knowledge, we are the first to present a tool that can automatically provide temporal context that is concise (not full Wiki pages) and consider the coherence of the article topic.

The rest of this paper is organized as follows. In Section 2, we will explain the methods underlying our system. Section 3 describes our implementation and the proposed demonstration plan. Finally, we present related work in Section 4 and conclude the paper in Section 5.

2. TIME-AWARE CONTEXTUALIZATION

The architecture of the Contextualizer system is shown in Figure 2. Its three main components are *context source processing* which extracts and annotates contextualization units from the context source; *query formulation* which builds the appropriate queries with using different formulation methods; and *context ranking*, which retrieves and re-ranks the results in a time-aware fashion.

2.1 Context source processing

The first step is to consider which sources to be used for extracting contextualization candidates. In this demonstration, we employ Wikipedia because it is considered the largest and most up-to-date online encyclopedia covering a wide temporal range of general and specific knowledge; but the others such as News Archive can be also used. The system processes each contextualization candidate by extracting features and storing them to facilitate faster retrieval and re-ranking. We use Stanford CoreNLP[1] for tokenization, entity annotation and temporal expression extraction. In addition, *anchor* texts found in the hyperlinks are also extracted.

2.2 Query formulation

The goal of the query formulation component is to generate a set of queries for a given document to retrieve contextualization candidates as input for the next component, i.e. context ranking. For building appropriate queries, we explore two families of methods, one using the document itself as a "generator", and the other using contextualization hooks as generators.

For the former family of query formulation methods, we use three strategies which exploit the document content and structure including *title* which is indicative of main topic of the document; *lead* which is the lead paragraph of the document, representing a concise summary of the document and its main actors; and *title+lead* which is a combination of the previous methods.

Based on user-highlighted keywords, i.e. contextualization hooks[2] which can be not only entity mentions, concept mentions, but also general terms and even short phrases, we consider the hook-based query formulation method by including all the hooks in a single query, representing a tailored perspective of the user's combined information needs for the document. Because the hooks are considered in the context of the document, we enrich the hook-based queries by the title of document.

Before being performed, all the queries are pre-processed by tokenization, stop-word removal and stemming.

2.3 Context ranking

The queries determined given a document d in the previous step serves as a starting point for retrieving the ranked list of contextualization candidates. In order to obtain the candidates, we use query-likelihood language modeling [6] to estimate the similarity of a query q with the context c.

$$P(c|q) \propto P(c) \prod_{w \in q} P(w|c)^{n(w,q)} \quad (1)$$

where w is a query term in q, $n(w,q)$ is the term frequency of w in q, and $P(w|c)$ is the probability of w estimated using Dirichlet smoothing:

$$P(w|c) = \frac{n(w,c) + \mu P(w)}{\mu + \sum_{w'} n(w',c)} \quad (2)$$

where μ is the smoothing parameter, $P(w)$ is the probability of each term w in the collection.

Once we have obtained a ranked list of contextualization candidates for each document, we turn to context selection where we need to decide which of the context items are most viable. Our ranking algorithm needs to balance two goals, i.e., high topical and temporal relevance as well as complementarity for providing additional information. To this end, we exploit various complementarity features and use the trained model on a set of manual labeled samples (context to document mappings) [7] to re-rank the results.

Relevance and temporal features In order to retrieve high topical and temporal relevant contextualization candidates for the document, we first consider both relevance and temporal features. For the former one, we exploit the retrieval scores of context returned by our retrieval model. For the later one, we apply temporal similarity measurement, i.e., TSU computed as follows

$$TSU(t_1, t_2) = \alpha^{\lambda \frac{|t_1 - t_2|}{\mu}} \quad (3)$$

where α and λ are constants, $0 < \alpha < 1$ and $\lambda > 0$, and μ is a unit of time distance.

Complementarity features We make use of several types of complementarity features that are confirmed an important role in contextualization including *topic diversity, text different, anchor text distance* and *geometric distance* (see [7] for computation details).

[1] `http://nlp.stanford.edu/software/corenlp.shtml`

[2] We use user-highlighted keywords and contextualization hooks interchangeably

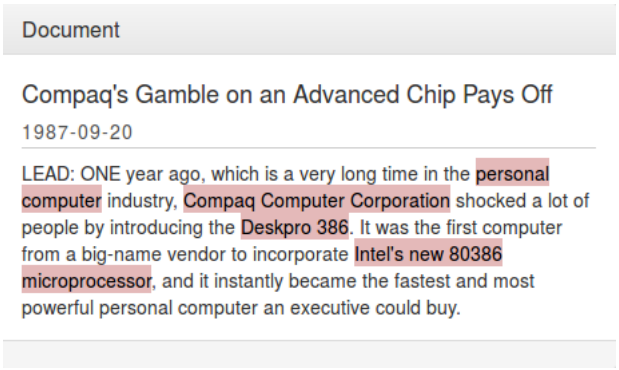

Figure 3: An example of user-highlighted keywords.

3. DEMONSTRATION OF THE SYSTEM

3.1 Implementation

Contextualizer is implemented in Java and uses Lucene for indexing and retrieving contextualization candidates. After computing all complementarity measures between each contextualization candidate and the document, we make use of the trained model on manual labeled examples using Random Forests, a learning-to-rank algorithm, to re-rank the results [7]. We implemented a web interface using the Angular[3], Bootstrap[4] frameworks and some parts from Elianto[5], all the actions are performed calling the REST api provided by the server. In order to acquire user feedback information for improving our model, we allow users to rate the candidates according to how much they are contextually related to the document in four different ratings:

Top relevant (3 stars) if the candidate are topically and temporally related to the document, and complements to the main topics of the document.

Highly relevant (2 stars) candidates are descriptions of the essential entities (e.g., what are entities) discussed in the document.

Partially relevant (1 star) candidates that provide background information about minor aspects mentioned in the document

Not relevant candidates that are about different topics or repeat the same thing in the document.

In this demo, we use 51 articles that spanned a wide range of topics and publication dates from New York Times Corpus[6]. The Wikipedia dump of February 4, 2013 is used as a context source and paragraphs are considered as contextualization units. The online system are published at `http://forgetit.l3s.uni-hannover.de:9090/wikinews-webapp`.

3.2 Scenarios

In this demonstration, we will show how to retrieve additional information to the document using our Contextualizer system. As one of examples for an article of interest, suppose that a user is reading the article published in 1987 about Deskpro 386 and highlights several keywords that he/she wants more information at that time such as *"Compaq Computer Corporation"*, *"Desktop 386"*, *"Intel's new 80386 microprocessor"* as shown in Figure 3. In order to support the user, our system automatically suggests some candidates by using the spotting module of Dexter [1], a framework that provides all the tools needed to develop any Entity Linking technique. The user can either use these keywords or highlight his/her own ones. The Contextualizer system will use these highlighted keywords to build appropriate queries (see Section 2.2) such as a query *"Compaq Computer Corporation and Desktop 386 and Intel's new 80386 microprocessor"* and issue it to our retrieval component (see Section 2.3). First, a list of contextualization candidates are retrieved based on language modeling, and the system will decide which candidates are to be shown as context based on a variety of features and a learning to rank algorithm, i.e., Random Forests.

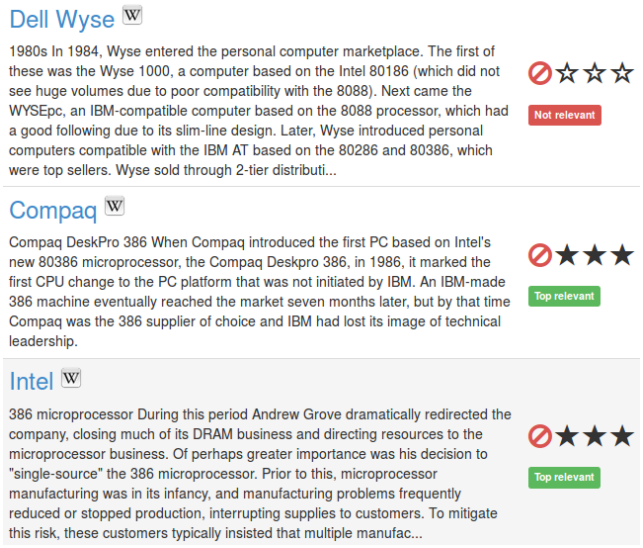

Figure 4: Contextualization candidates retrieved by the Contextualizer system.

As shown in Figure 4, three contextualization candidates are returned with its Wikipedia page title and followed by the actual content. In our example, we observe that the two candidates *"When Compaq introduced the first PC based on Intel's new 80386 microprocessor, the Compaq Deskpro 386, in 1986, it marked the first CPU change to the PC platform that was not initiated by IBM..."* and *"During this period Andrew Grove dramatically redirected the company, closing much of its DRAM business and directing resources to the microprocessor business. Of perhaps greater importance was his decision to "single-source" the 386 microprocessor..."* contextually relates to the document, whereas the last one does not.

As discussed in [7], contextualization hooks can be not only entity mentions or concept mentions but also general terms and even short phrases, as shown in Figure 5. In this example, a user highlights two phrases *"experimental student loan program"* and *"shift the costs"*. These phrases explicitly represent the information needs of the user or, more precisely, what requires contextualization to be understood and interpreted. Here the entity linking approaches can not dealt with these contextualization hooks since there is no explicit Wikipedia page for such phrases. In contrast to these approaches, Contextualier system can be able to retrieve contextualization candidates based on the user-

[3] `https://angularjs.org/`
[4] `http://getbootstrap.com/`
[5] `https://github.com/dexter-entity-linking/elianto`
[6] `http://catalog.ldc.upenn.edu/LDC2008T19`

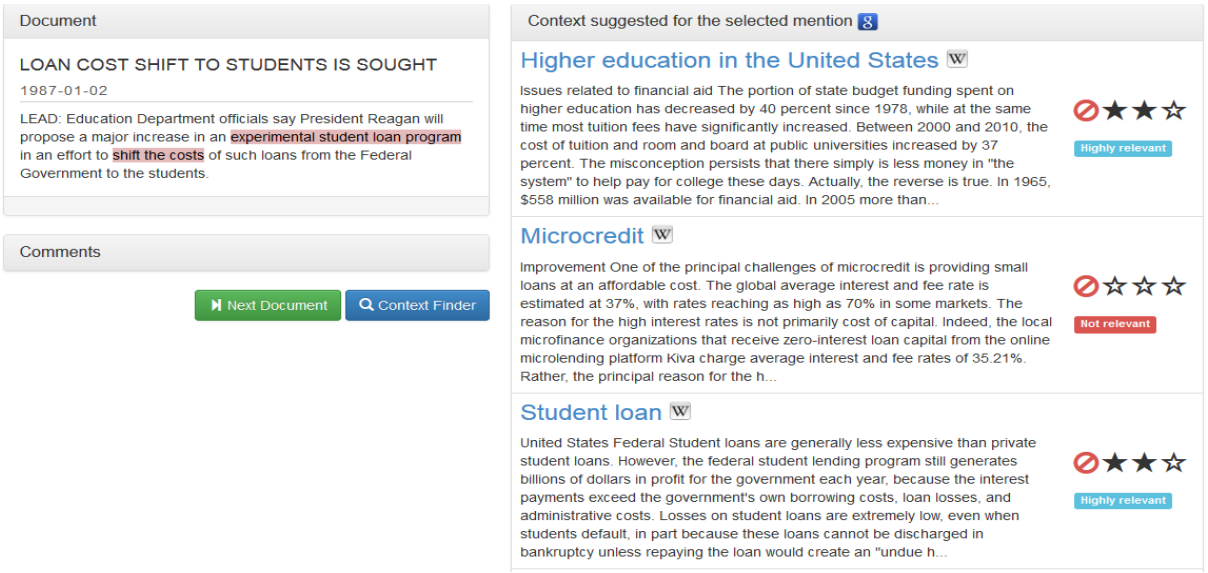

Figure 5: An example of the document with non-entity hooks and its contextualization candidates.

highlighted phrases as shown in Figure 5 (*"Higher education in the United States"* and *"Student loan"*). Again, the user can also rate the relatedness of the candidates which can be used to improve the learning process.

4. RELATED WORK

Previous work has focused on detecting entity name mentions within text and links them to the corresponding entities in a knowledge base [1, 2, 5]. In contrast to our approach, both Wikification and entity linkage approaches lack two ingredients of time-aware contextualization, (a) they do not take into account the temporal aspect of the text to be enriched and (b) the additional information provided is rather general (e.g., Wikipedia articles about an entity) and not focused to the topical information need resulting from the text under consideration.

Retrieving and processing external information to be added to documents have obtained increasing interest in the recent years. Kanhabua et. al, [4] proposed to enrich news articles with related predictions. Other works [8, 9] exploit social media (e.g, Twitter) as external source to discover utterances that discuss a given news article. In contrast to those approaches, our work adds another dimension to the contextualization task, namely time. We are not looking for more information on the current context, but we try to reconstruct the original context of a document.

The work most related to ours is contextual insights [3] that provides users with information that is contextually relevant to the content that they are consuming. However, they do not take into account temporal dimension. To the best of our knowledge, we are the first to present a web-based system that automatically provide temporal context for a given document in a meaningful way.

5. CONCLUSIONS

We present *Contextualizer*, a web-based system for automatically contextualizing a news article by providing complementing information to the article, which reflects required, but not expressed, context for fully understanding it. Our system allows users to select their interested keywords, and automatically constructs appropriate queries to retrieve contextualization candidates, then re-ranks the candidates before presenting them to users in a interactive manner. While our initial illustration of the Contextualizer system has focused on news articles and Wikipedia, others sources such as qualitative studies and news archive can be also applied.

Acknowledgements The work was partially funded by the European Commission for the FP7 project ForgetIT under grant No. 600826 and the German Federal Ministry of Education and Research (BMBF) for the project *"Gute Arbeit" nach dem Boom (Re-SozIT)* (01UG1249C).

6. REFERENCES

[1] D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Dexter: An open source framework for entity linking. In *ESAIR '13*, 2013.

[2] P. Ferragina and U. Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *CIKM '10*, 2010.

[3] A. Fuxman, P. Pantel, Y. Lv, A. Chandra, P. Chilakamarri, M. Gamon, D. Hamilton, B. Kohlmeier, D. Narayanan, E. Papalexakis, and B. Zhao. Contextual insights. In *WWW '14*, 2014.

[4] N. Kanhabua, R. Blanco, and M. Matthews. Ranking related news predictions. In *SIGIR '11*, 2011.

[5] D. Milne and I. H. Witten. Learning to link with wikipedia. In *CIKM '08*, 2008.

[6] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98*, 1998.

[7] N. K. Tran, A. Ceroni, N. Kanhabua, and C. Niederée. Back to the past: Supporting interpretations of forgotten stories by time-aware re-contextualization. In *WSDM '15*, 2015.

[8] M. Tsagkias, M. de Rijke, and W. Weerkamp. Linking online news and social media. In *WSDM '11*, 2011.

[9] T. Štajner, B. Thomee, A.-M. Popescu, M. Pennacchiotti, and A. Jaimes. Automatic selection of social media responses to news. In *KDD '13*, 2013.