Reception of LGBT in Newspapers

Magdalena Bönisch Till Haubenreißer Maximilian Möller

February 24, 2017

Universität Leipzig, Introduction to Digital Humanities (Dr. Köntges)

Abstract Here are 150 to 200 words which constitute our abstract.

1 Introduction

This is our pretty Introduction.

2 Workflow & Implementation

In the following, the general workflow for analyzing the LGBT reception within newspapers is described. After that, the concrete decisions concerning the implementation are addressed.

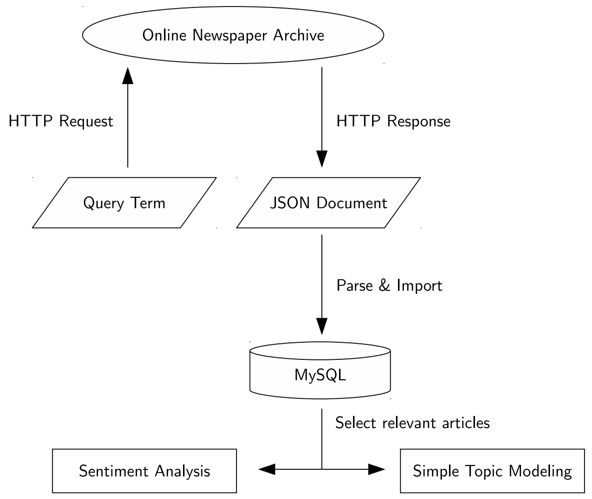

Figure 1: Workflow

Workflow

The overall workflow is shown in Fig. 1. The upper part reflects the first step of collecting newspaper articles and creating a database. For each term from a predefined list of query terms an HTTP response to the API of an online newspaper archive is sent. In this work, the Article Search API of the New York Times[1] is used as it provides access to articles published since its foundation in 1851 and thus enables an extensive time-dependent analysis. Furthermore, the returned JSON documents contain not only a URL to the actual article but also text snippets and lead paragraphs such that an analysis can be based on these text fields. Hence, no further call for the complete article is necessary. Besides meta-data like the number of hits, the API returns a JSON object consisting of the attribute-value pairs of the articles. Since the access to the NYT API is limited per second and per day, the responses are parsed and stored into a MySQL[2] database. A relational database allows a quick access of the data along with SQL as a powerful query language for data selection and summarization. By using the database, subsequent analysis could be based on the whole corpus with no dependency from the API. Furthermore, the proposed workflow is more flexible with respect to adding data from an additional newspaper archive since its response has only to be mapped to schema of the database and, especially, the analysis module (lower part of the workflow in Fig. 1) remains unaffected.

The analyses are based on two dimensions: query term and time. Thus, the input of the sentiment analysis and the simplified topic modeling approach is a set of lead paragraphs containing the query term and contained in articles published at a certain time. For instance, such an input can be all paragraphs from the database which contain the word *homosexual* and were published in the 1990s.

The sentiment analysis assigned each sentence a vector of sentiment probabilities. We considered five sentiments: *very negative*, *negative*, *neutral*, *positive*, and

[1] `https://developer.nytimes.com/`
[2] `https://www.mysql.com/`

very positive. For example, the vector

$$v_s = \langle 0.2, 0.5, 0.1, 0.1, 0.1 \rangle$$

describes the case that a sentence s has a probability of 20% for having a very negative sentiment, 50% for having a negative sentiment, and 10% for having a neutral, positive, and very positive sentiment, respectively. For the topic modeling, the joint occurrence of a target term together with another word (cooccurrence) is analyzed. Target terms are LGBT-related words, like the query terms.[3] Each cooccurrence is assigned its frequency and whether its occurrence is significant. For instance, the cooccurrence

$$c_{2010\mathrm{s}} = \langle \mathrm{bisexual}, \mathrm{rights}, 31, \mathrm{true} \rangle$$

means that within the lead paragraphs from 2010 to 2019, *bisexual* and *rights* occurred 31-times in the same paragraph and that this occurrence is significant. By clustering significant cooccurrences, the topics were manually created in order to yield high-quality topics.

Implementation

The whole workflow is implemented in Java as a Maven[4] project. The project is managed on GitHub.[5] It consists of one module for collecting and one for analyzing the data. The database is accessed by using the Java Database Connectivity (JDBC) API[6].

Collecting Module The used query terms are *bisexual*, *gay community*, *homosexual*, *lesbian*, *transgender*, and *transsexual*. However, the term *gay* was not considered as a query term because it appeared too often in non-LGBT-related contexts like as a given name or surname. In order to obey the time limit of the NYT API, only one request per second was sent. Almost all requests (95.4%) were successful. They returned a non-empty JSON object which could be inserted into the database. Failures of requests were due to denied accessed to the archive (HTTP 403) or gateway timeouts (HTTP 504). Because such failures were very rare, failed requests were not sent for a second time. All in all, the built corpus consists of 44,485 articles whereby 93,7% of them contain a non-empty lead paragraph on which the analyses are based.

The database models the mapping of query terms and keywords to articles. Keywords represent interesting additional information to the articles like associated persons, organizations, or geographical information. Approximately, 75% of all articles have at least one keyword. Moreover, an article has a URL which leads to its HTML representation in two-thirds of the cases; otherwise the article is only accessible as a PDF document.[7] Further attributes are the publication date, the actual text type (for instance, article, interview, or biography) and headline, abstract, lead paragraph, and snippet. Since the headline is very short, the abstract is missing for 62% of the articles, and the snippet is mostly the same as the lead paragraph, the analysis is based on the lead paragraph.

Analysis Module Essentially, the analysis module of the Maven project comprises the sentiment and the topic model package. They both rely on the sentence extraction task. For a certain search word and a certain publication date, this task selects windows of sentences from the paragraphs in the database. Given a window size of 2, for instance, additionally to the sentence containing the search word the two directly preceding as well the two directly succeeding sentences are extracted.[8] For the sentiment analysis, we chose a window size of 0, i.e. only the containing sentence, and for the topic modeling a size of 1. Both of the analyses depend on the natural language processing library Stanford CoreNLP[9]. This library contains annotators for tokenization, sentence splitting, part-of-speech tagging, and sentiment analysis. Since the sentiment annotator is based on the sentence structure, it is expected that it returns better results than approaches which only count the occurrences of particular negatively or positively connotated words or phrases while ignoring their syntactic context [SPW+13]. However, determining the sentence sentiment is a very time consuming operation. Therefore, we chose the minimal window size.

The simplified topic modeling approach consists of three steps: preprocessing, cooccurrence counting, and

[3] The query terms are used for building an LGBT-corpus. But within this corpus it possible to search for other terms than query terms as well. For instance, *lesbian* is a query term as well as a target term. *gay* is only a target term.

[4] `https://maven.apache.org/`

[5] `https://github.com/macksimiljan/lgbt-news`

[6] `http://www.oracle.com/technetwork/java/javase/jdbc/index.html`

[7] Older articles are archived as PDF not as HTML.

[8] For the given corpus, a window size greater than 4 will have no difference to a smaller size since the paragraphs consists only of a few sentences. For instance, not more than 40% of the paragraphs consists of two or more sentences.

[9] `http://stanfordnlp.github.io/CoreNLP/`

the statistic evaluation. The preprocessing is executed for the corpus only once. For all paragraphs, the contained words and their number of occurrence is determined. Stop words and numbers are excluded from this word statistics. The stop-word list is based on [MS03, p.533] whereas numbers are recognized by a regular expression. After creating the word statistics, it is possible to define the list of context words. A context word is a meaningful word which occurs in a cooccurrence with a target word. A target word is a LGBT-related term. A word is assumed to be meaningful if it is not a stopword or a number and has a minimal frequency of 3. The frequency condition enforces that words being typographical errors are excluded from the analysis. After preprocessing, the word statistics consists of approximately $66 \cdot 10^3$ word types and $1.2 \cdot 10^6$ word tokens.[10] 43.4% of all words occur only once or twice. Thus, there are approximately $29 \cdot 10^3$ context words.

In the next step, all cooccurences of a target word w_{target} with one of the n context words are counted in the paragraphs containing w_{target} and published in a certain time span t. To yield the cooccurrences and their number, these paragraphs were tokenized and cleansed by removing all words from the sentences which are not context words and transforming it lower case. For instance, the sentences

The president argues for gay rights. He is tolerant.

is mapped to

president argues gay rights tolerant

This sequence of words is used to built a context vector $v_t(w_{\text{target}}) = \langle c_1, c_2, \ldots, c_n \rangle$ for each w_{target} and t such that c_i represents the (absolute) frequency of the cooccurrence of w_{target} with the context word at list position i. Because cooccurrence is not a reflexive relation between two words, $c_i = 0$ if the context word at position i equals w_{target}. In the example, assume that the list of context words L consists of six words:

$$L = [\text{argues, gay, president, rights, tolerant, usa}]$$

Depending on the chosen maximal distance $d_{\max}$, the context vector $v_t(\text{gay})$ for the target word *gay* is built. Let $d_{\max}$ be 2, then there can no more than $d_{\max} - 1 = 2 - 1 = 1$ word be between the target word and its cooccurrence partner. This yields the context vector

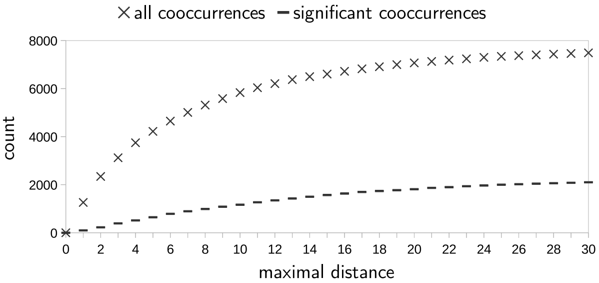

Figure 2: Convergence of the number of cooccurrences.

$$v_t^1(\text{gay}) = \langle 1, 0, 1, 1, 1, 0 \rangle$$

because every word except *usa* occurs with *gay* exactly once. For $d_{\max} = 1$, only the direct neighbors of *gay* are considered resulting in the context vector

$$v_t^2(\text{gay}) = \langle 1, 0, 0, 1, 0, 0 \rangle.$$

Studies suggest that the average sentence length in written prose is between 20 and 25 tokens, for instance [Sic74]. The maximal distance $d_{\max}$ is not sentence-sensitive, i.e., it ignores sentence boundaries. Nevertheless, as the paragraphs are only a few sentences, there is a maximal distance $d'_{\max}$ such that for all distances greater than $d'_{\max}$ the number of cooccurrences is the same. Fig. 2 shows this for the count of cooccurrences with *gay* in the 2010s. The number of all cooccurrences as well as the number of significant cooccurrences converge to 7650 and to 2300, respectively. These values are reached for a maximal distance greater than 56 (not shown in the figure). Tests with further target words lead to similar results. However, the smaller the maximal distance, the more semantically connected the words of the cooccurrence are expected to be. Choosing a great distance, the cooccurrences can connect words which probably are in different sentences. Since the sentence boundary is a semantic boundary as well (each sentence represents a logical statement on its own), a great maximal distance results in cooccurrences which might include words which should not be interpreted as being semantically connected. Thus, we run the program with a maximal distance of 6 and of 14 for all target words and times. For $d_{\max} = 6$ ($d_{\max} = 14$), approximately 67% (85%) of all cooccurrences and 40% (67%) of all significant cooccurrences are detected.

The last step is the calculation significant cooccurrences. According to [Bor08, MS03], three significance measures were implemented: mutual information, log-likelihood, and t-score. For test data, the t-score yields

[10] Since we followed a simple approach, we did not run a lemmatizer. For comparison, the Oxford Dictionary counts about $200 \cdot 10^3$ (lemmatized) word types (inclusive stop words) in the English language [OD].

Table 1: Size of the data basis.

term	#articles	#paragraphs
bisexual	4, 920	2, 973
gay	–	10, 392
gay community	1, 818	377
homosexual	15, 086	5, 263
lesbian	15, 249	5, 086
queer	–	162
transgender	6, 138	2, 973
transsexual	1, 229	279

the best results. Let n be the number of context words, n_{t} and n_{c} the number of occurrences of the target word and of a context word, respectively. Furthermore, n_{tc} is the count of the cooccurrence of w_{target} and w_{context}. Then the t-score is defined according to Eq. 1.

$$\mathrm{t}(w_{\mathrm{target}}, w_{\mathrm{context}}) = \frac{n_{\mathrm{tc}} - \frac{n_{\mathrm{t}} \cdot n_{\mathrm{c}}}{n^2}}{\sqrt{n_{\mathrm{tc}}}} \tag{1}$$

Consequently, a cooccurrence is defined as significant if it passes the t-test for $p = 0.005$. Additionally, all cooccurrences were selected occurring more often than on average in order to get further hints for topics.

//TODO: keyword analysis

3 Underlying Data

4 Results

4.1 Sentiment Analysis

This is our spectacular sentimental analysis.

4.2 Simple Topic Modeling

This is our great topic modeling.

5 Discussion

This is our insightful discussion.

5.1 Implementation Issues

For comparing the reception of LGBT-related terms in different newspapers, it is necessary to integrate further newspaper articles into the database. Especially for The Guardian[11], this is no problem since the Guardian API returns less information than the NYT API, e.g. no publication date or lead paragraph. However, the most relevant information is the web URL to the article. Thus, in order to get the publication and textual information, the workflow has to be extended by a further request for receiving the HTML page of the full article as well as by a parser for extracting relevant information from this HTML document. When such an extension exists, it can be used to get the full article also from the NYT archive. The full article body would enable a more comprehensive topic modeling. Attention should be paid that the Guardian API provides only access to publication dating back to 1999.

A general problem with the newspaper archive APIs is that the access is limited. Assume that there are $150 \cdot 10^3$ articles for a certain query term; as it is the case for *gay* in the NYT archive. Since the NYT API allows requesting only $10 \cdot 10^3$ articles per day, collecting the whole result set takes 15 days. The same amount of articles is no bigger problem for the Guardian API where $250 \cdot 10^3$ articles can be requested.[12] Additionally, only the first 1200 articles can be addressed within requests to the NYT API. Consequently, in order to get all articles, each call should not return more than that. In this approach, the articles of a query term were requested for a certain period of time.

5.2 Content Issues

5.3 Future Work

6 Conclusion

This is our awesome conclusion.

References

[Bor08] Stefan Bordag. A comparison of co-occurrence and similarity measures as simulations of context. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 9th International Conference*, pages 52–63. Springer, Berlin, 2008.

[MS03] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing.* MIT Press, Cambridge (Mass.), 2003.

[OD] online Oxford Dictionary. How many words are there in the English language?

[11] `http://open-platform.theguardian.com/`

[12] The Guardian allows $5 \cdot 10^3$ calls per day and a maximum amount of 50 articles per call.

https://en.oxforddictionaries.com/explore/how-many-words-are-there-in-the-english-language. last call: 2017, Feb. 20th.

[Sic74] HS Sichel. On a distribution representing sentence-length in written prose. *Journal of the Royal Statistical Society. Series A (General)*, pages 25–34, 1974.

[SPW+13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment tree-bank. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Who did What?

Task	M. Bönisch	T. Haubenreißer	M. Möller
general decisions	✓	✓	✓
programming			✓
basic data analysis			✓
diagram creation		✓	
background recherche	✓	✓	
evaluation of sentiment			
topic evaluation			
introduction text			
workflow & implementation text			✓
underlying data text			
results text			
technical discussion text			
content discussion text			
future work text			
conclusion text			
typesetting			✓