Tracking German News Topics using Latent Dirichlet Allocation

Thomas Werkmeister

Hasso Plattner Institut, Potsdam, Germany

March 20, 2015

Abstract

Latent Dirichlet Allocation(LDA) is a generative model that allows inference of hidden groups in observed data and thus explain similarities in the data. In Natural Language Processing(NLP) it has been used to find topics in large document collections. In this technical report I present my own implementation of the LDA algorithm. Moreover, I used this implementation to detect topics in a corpus of German news articles I collected over the time span of 20 weeks. Finally, I analyze the prevalence of the detected topics over time.

1 Introduction

More and more news are available to us everyday. In addition to traditional online news papers there are now thousands of blogs or twitter users that cover news worthy stories. Moreover, the coverage of the media has broadened. Not only local issues are addressed, but anything that happens in the world which might be relevant to the reader. With the recent advances in the field of robot journalism by companies such as Narrative Science[1] we can expect that the amount of news stories available will increase further. To enable readers to stay on top of events computer assisted ways of finding and consolidating news articles are needed. In this report I will examine the possibilities of automated tracking of topics in traditional online news papers using LDA, an algorithm from the field of computational linguistics. I will especially focus on practical issues when using LDA.

2 Implementation of LDA

Whilst the reasoning behind this algorithm require advanced knowledge in statistics and mathematics the implementation of the algorithm using Gibbs Sampling seems straight forward. Still it is necessary to be absolutely exact, as small mistakes will make the topic distributions of documents and the word distributions of topics converge in wrong ways.

[1] http://www.narrativescience.com/

I used [1], [2] and [3] as instructions for implementing the algorithm. Moreover, I tried to achieve a reasonable performance using the right data structures and filtering out as many words as possible.

2.1 Reducing the number of words

It is crucial to filter out unnecessary words for both effectiveness and efficiency of the algorithm. Every word in the processed corpus entails multiple variable assignments, an iteration over all topics and a draw from a probability distribution. Furthermore, usually the algorithm involves hundreds of iterations over the whole processed corpus. In terms of effectiveness it is important to filter out stopwords such as "die", "der", "das" (engl: the) or "bald" (engl: soon) that do not have relevant semantic for topics. When those words remain in the corpus it is likely that they form topics themselves or take up probability mass in other topics and dilute them.

I filtered out words in two steps. First, German stopwords are filtered out according to a standard German stopword list. Additionally, words that are shorter than 3 characters are filtered out. Secondly, words are filtered out based on the characteristics of the Zipfian distribution of words. Words that appear only in one document are filtered out and also words that appear in more of 25% of the documents. Those measures reduce the vocabulary size (number of unique words) by almost 50%:

stopwords 597 words (daher, sollte, zweiten, sechs, ehrlich, welchen, hatten, ...)

short words 865 words (..., 87, 88, 89, 8j, ... uw, uz, v2, ...)

frequent words 9 words (2014, 2015, deutschland, dpa, euro, foto, ...)

infrequent words 107491 words (1990gern, 19f7khb, 19jahren, ..., abdichten, abdiwali, ..., auslandskunden, auslandsmarkt, ...)

resulting vocabulary size 112554 words

2.2 Performance

The performance of the algorithm highly depends on the data structures being used. My first implementation which was very close to the pseudo code of [1] and used hashmaps for counting the word to topic and topic to document counts. With this implementation a small test run with 100 documents, 100 topics and 100 iterations would take around 3 minutes. As it turned out the hashmap lookups consumed more than 80% of the time. Processing a whole corpus would be infeasible using this implementation. The matrix based approach of [3] led to better results. When combined with keeping an intermediate counter for total number of words assigned to a topic optimization mentioned in [1] the system now processes the same sample in only 6 seconds. For the whole corpus of around 20000 documents, 75 topics and 250 iterations my system needs roughly 35 minutes. Times have been measured using a Intel(R) Core(TM) i7-4600U CPU @ 2.10GHz and 2GB of memory.

2.3 Possible improvements

To improve the effectiveness of the system German compound nouns and the many inflection forms of the words should be addressed. Many compound nouns are sorted out, because they only occur once in this combination, but their single parts have a clear and valuable semantic:

Staats... staatsempfang, staatsetat, staatsexamina, staatsfarben, staatsfeindin

Tabak... tabakgegnerin, tabakgenuss, tabakgesetze, tabakhandel, tabakmarke, tabakproduktrichtlinie

3 Finding topics in German news articles

3.1 Data set

I collected the data used for this experiment via the RSS feeds of different online news papers in Germany. The data covers the politics and economy sections of the Süddeutsche, Spiegel, Stern, Welt, FAZ, RP-online, Zeit, N24 and N-TV. I did not collect articles from the biggest German news paper, Bild, because it does not provide RSS feeds. When a new article is published the article's website is fetched and the main text is extracted using Boilerpipe[2]. The extracted text sometimes has small flaws and might contain fragments from image captions or embedded advertisements.

My data set covers November 2014 until mid March 2015. In total I collected 20211 news articles. Unfortunately, this set does not contain all articles of this time frame. As I gathered the articles on my own machine it was not possible to run the collection program continuously. Also some news articles were not collected due to parsing errors. Figure 1 shows the number of articles grouped by week of year.

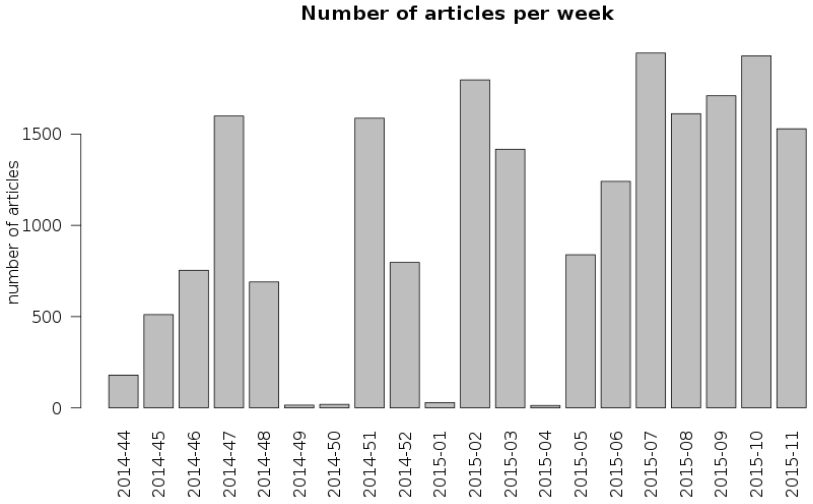

Figure 1: number of articles per week

[2] https://code.google.com/p/boilerpipe/

3.2 Topics in German News

When using LDA a number of topic K has to be specified. Initially I calculated topics for K=100. The following example topics show that this K is too big, as some real topics were split into multiple LDA topics.

Topic #68 griechenland, finanzminister, athen, schäuble, verlängerung, varoufakis, brüssel, griechischen

Topic #97 griechenland, milliarden, athen, varoufakis, griechische, tsipras, griechischen, finanzminister

Topic #84 paris, kouachi, attentäter, polizei, hebdo, charlie, brüder, coulibaly

Topic #89 charlie, paris, hebdo, frankreich, polizei, französische, anschlag, kouachi

Topic #94 charlie, hebdo, anschlag, täter, kopenhagen, attentäter, paris, zeitung

When calculating topics for K=50 it became apparent that this number of topics is too small. The following examples show that multiple real topics get mixed into a single LDA topic.

Topic #5 frauen, frauenquote, schwesig, japan, unternehmen, abe, quote, ebola

Topic #0 nordkorea, usa, video, syrien, kim, staat, assad, sony

Setting K=75 mostly alleviated the former two problems. A problem that remains are topics that do not represent a news topic, but that feature words that are somehow related to the medium online news.

Topic #41 online, fehler, focus, details, teilen, melden, video, informationen

Topic #54 facebook, webseite, browser, direkt, informationen, baut, basf, reiche

For the the rest of the report I will focus on two examples of the well trained topics:

Topic #6 (Charlie Hebdo attack) paris, polizei, terror, frankreich, attentäter, anschlag, coulibaly, islamisten

Topic #58 (renewable energies) strom, energiewende, energie, millionen, energien, gas, co2, kraftwerke

4 Analyzing the life cycle of topics

Given the distribution over topics for each document the life cycle of a topic can be analyzed. It can be observed when topics first come into existence, how their prevalence evolves and when they disappear again.

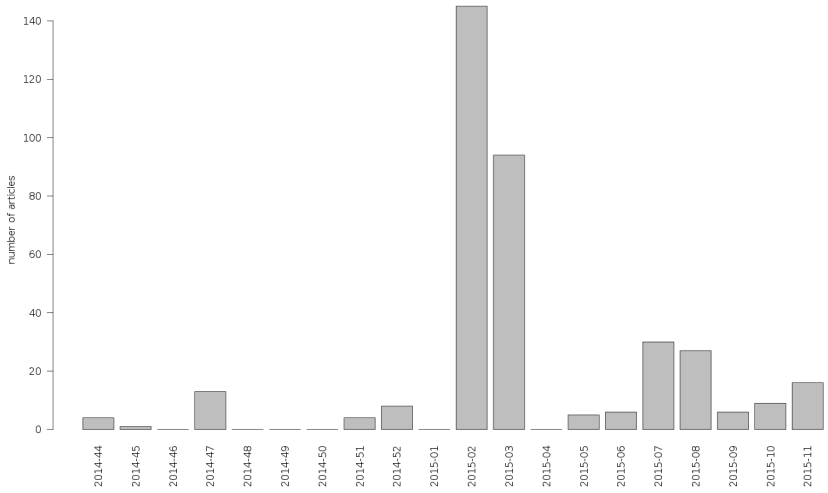

Figure 2: Prevalence of Topic #6 (Charlie Hebdo attack)

4.1 Prevalence of topics

For a topic to be a prevalent topic in a document it should have a certain amount of the probability mass of the topic distribution of a document. [4] used a threshold of 40%. Hence, when a topic has more than 40% of the probability mass of the documents distribution over topics it is considered a prevalent topic. Using a threshold of 40% it turns out that only 13543 out of 20211 or roughly $\frac{2}{3}$ of the documents have such a strong topic assignment. Figures 2 and 3 show the prevalence of the two selected topics. The figures show that topics have different life cycles. The topic on the Charlie Hebdo attack spiked during the event and the immediate follow up then dropped significantly and is still covered with a couple of articles per week. The second topic on renewable energies behaves different. It is generally covered less and is more long lasting. It is being covered almost every week with a few articles and sometimes gets more interest.

4.2 Main topic per week

I define the main topic of a week as the topic that, compared to other topics, is prevalent in the greatest number of news articles published in that week.

2014-44 X (too few articles)

2014-45 Topic #4: bahn, gdl, gewerkschaft, streik, streiks, evg, ...

2014-46 Topic #30: merkel, ukraine, russland, putin, steinmeier, sanktionen, ...

2014-47 Topic #25: dax, punkte, aktien, dollar, index, anleger, ...

2014-48 Topic #73: spd, gesetz, frauen, schwesig, frauenquote, cdu, ...

2014-49 X (too few articles)

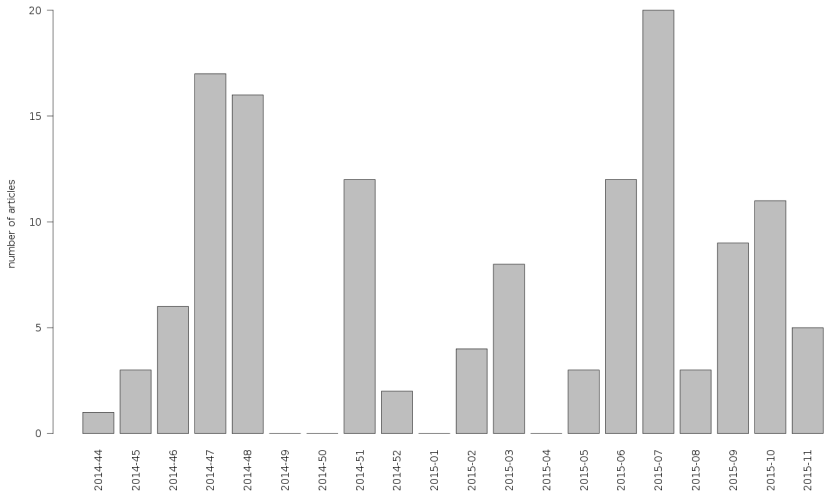

Figure 3: Prevalence of Topic #58 (renewable energies)

2014-50 X (too few articles)

2014-51 Topic #63: edathy, spd, hartmann, sebastian, ermittlungen, untersuchungsausschuss, ...

2014-52 Topic #50: dollar, rubel, russland, ölpreis, milliarden, währung

2015-01 X (too few articles)

2015-02 Topic #6: paris, polizei, terror, frankreich, attentäter, anschlag, ...

2015-03 Topic #6: paris, polizei, terror, frankreich, attentäter, anschlag, ...

2015-04 X (too few articles)

2015-05 Topic #17: griechenland, tsipras, varoufakis, athen, griechische, griechischen, ...

2015-06 Topic #17: griechenland, tsipras, varoufakis, athen, griechische, griechischen, ...

2015-07 Topic #68: ukraine, minsk, poroschenko, kiew, separatisten, ukrainische, ...

2015-08 Topic #51: griechenland, athen, finanzminister, verlängerung, schäuble, griechischen, ...

2015-09 Topic #51: griechenland, athen, finanzminister, verlängerung, schäuble, griechischen, ...

2015-10 Topic #24: russland, putin, russischen, moskau, russische, nemzow, ...

2015-11 Topic #9: syrien, irak, staat, kämpfer, islamischer, stadt, ...

5 Conclusion

I have shown that LDA is an interesting tool for topic and trend analysis in news media. Even with the sub optimal data set the model finds coherent topics and enables us to follow the course of topics in the news. Hence, this procedure could be used to enhance topic oriented information retrieval.

In future work it would be interesting test this approach on a bigger and more complete data set. Also, it might be interesting to use a Hierarchical Dirichlet Process (HDP) to track topics in the news. This kind of model does not require a fixed number of topics to be set, but adds new documents to either already existing topics or creates a new topic for them.

References

[1] W. M. Darling. A theoretical and practical implementation tutorial on topic modeling and gibbs sampling. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 642–647, 2011.

[2] A. Minnaar. Lda in scala part i - the theory. http://alexminnaar.com/latent-dirichlet-allocation-in-scala-part-i-the-theory.html. Accessed: 2015-03-20.

[3] A. Minnaar. Lda in scala part ii - the code. http://alexminnaar.com/latent-dirichlet-allocation-in-scala-part-ii-the-code.html. Accessed: 2015-03-20.

[4] R. K. Nelson. Mining the dispatch. http://dsl.richmond.edu/dispatch/. Accessed: 2015-03-20.