ASV Monitor: Creating Comparability of Machine Learning Methods for Content Analysis

Andreas Niekler[1], Patrick Jähnichen[2], and Gerhard Heyer[2]

[1] Faculty of Media, Leipzig University of Applied Sciences (HTWK)
[2] NLP Group, Department of Computer Science, University of Leipzig

aniekler@fbm.htwk-leipzig.de
{jaehnichen,heyer}@informatik.uni-leipzig.de

Abstract. In this demonstration paper we present an application to compare and evaluate machine learning methods used for natural language processing within a content analysis framework. Our aim is to provide an example set of possible machine learning results for different inputs to increase the acceptance of using machine learning in settings that originally rely on manual treatment. We will demonstrate the possibility to compare machine learning algorithms regarding the outcome of the implemented approaches. The application allows the user to evaluate the benefit of using machine learning algorithms for content analysis by a visual comparison of their results.

1 Introduction

Many emerging disciplines of the e-humanities like social sciences, media science, literature science or classical studies rely on content analysis theory. In those scientific fields a strong focus is laid on manual content analysis. Typically researchers in these fields are quite sceptical towards algorithmically driven content analysis approaches. With the presented application we provide a platform to motivate researchers in the humanities to compare and evaluate automatic methods for their own tasks using their own well known corpora. The freedom of importing corpora the contents of which are familiar will increase the acceptance of the automated methods. An almost confusing amount of different possible models for content analysis exist, facing researchers with the following problem: presenting the same data to different models naturally leads to outcomes that differ in its form or quality or both. This implies, that results from different models are not necessarily comparable and thus it is hard (if not impossible) to draw conclusions about the utility of using one model or the other. Another problem is the lack of visualisation of techniques of content analysis. Model implementations typically provide (if any) a rather rudimentary visualisation component, that is mostly comprised of ways to automatically generate HTML pages, showing the results in tables. A general way of visualising model outcome of different models for content analysis is clearly a goal to strive for. In

section 2, we present the architecture of the ASV Monitor[3], a modular platform providing a wide range of machine learning techniques for content analysis that is easily extendable, preprocesses data text data using common preprocessing steps, presents the preprocessed data to different algorithms and produces formalised model outcome. Passing the output to a visualisation component allows pictorial analysis of different models and thus provides a help for choosing the right one. The visualisation component is introduced in section 3 and an outlook on possible future extensions is given in section 4.

2 Architecture

Creating text corpora based content analysis applications requires the close interaction of algorithms and data structures. In our framework, data import is done by making use of a modular architecture which allows us to implement and add new importers for different kinds of sources. Within those, text is extracted and is then passed to sub-sequent modules doing noise reduction, i.e. stopword pruning, stemming and other standard pre-processing steps.

2.1 Algorithms

As content analysis mostly deals with unknown text sources, we implemented several unsupervised state-of-the-art machine learning algorithms to evaluate different aspects of text based content analysis. Analysts generally do not have any information about the nature of the text collection, so the problem to tackle is *not* a traditional classification problem, as in most cases we do not have training data, hence the focus on unsupervised algorithms. We concentrated on capturing the content analysis tasks defined by [1], which are: *summarisation*, *category and topic extraction*, *word context and word usage* and *meaning shift of topics and words over time*. We identified machine learning algorithms that can be adapted and aligned them to one of the above tasks. Again, the modularity of the system allows for new algorithms and visualisations to be included, thus enabling us to test and evaluate new ideas for using machine learning algorithms within content analysis tasks. Our current version comprises the following algorithms to address the different tasks:

Topic Models Latent Dirichlet Allocation[2] and Hierarchical Dirichlet Processes [3], *Tasks:* summarisation, category and topic extraction, meaning shift of topics and words over time
Topic Detection and Tracking [4], *Tasks:* summarisation, category and topic extraction, meaning shift of topics and words over time
Cooccurence analysis [5], *Tasks:* word context and word usage, meaning shift of topics and words over time
Word burstiness [6], *Tasks:* word context and word usage

[3] ASV stands for *Automatische SprachVerarbeitung*, a German translation of Natural Language Processing; the prototype can be accessed at `http://monitor.asv.informatik.uni-leipzig.de`, user credentials: ecml_pkdd12/ecml

2.2 Visualisation

The resulting visualisation is the core functionality of our framework since it provides the main access point to the results of a specific algorithm. Also, it enables us to critically discuss the decision for a particular algorithm with respect to a specific task.

The main screen of the application is a dashboard which allows the user to retrieve and interact with the results of the different algorithms as described above. We use Apache Flex[4] technology, because it provides a large number of ready to use components for the implementation of a data driven dashboard. Within the dashboard, the user is able to select a text corpus and review the results of different algorithms. To investigate time varying effects within a text collection, the framework provides access to corpora that span over a time period via a calendar-like interface. We split the main screen into a data grid centering on an informative overview of the algorithm's results (Fig. 1 top) and a detailed view showing the actual visualisations of word level aspects of the results (Fig. 1 bottom). There exist visualisations within the ASV Monitor for each of the

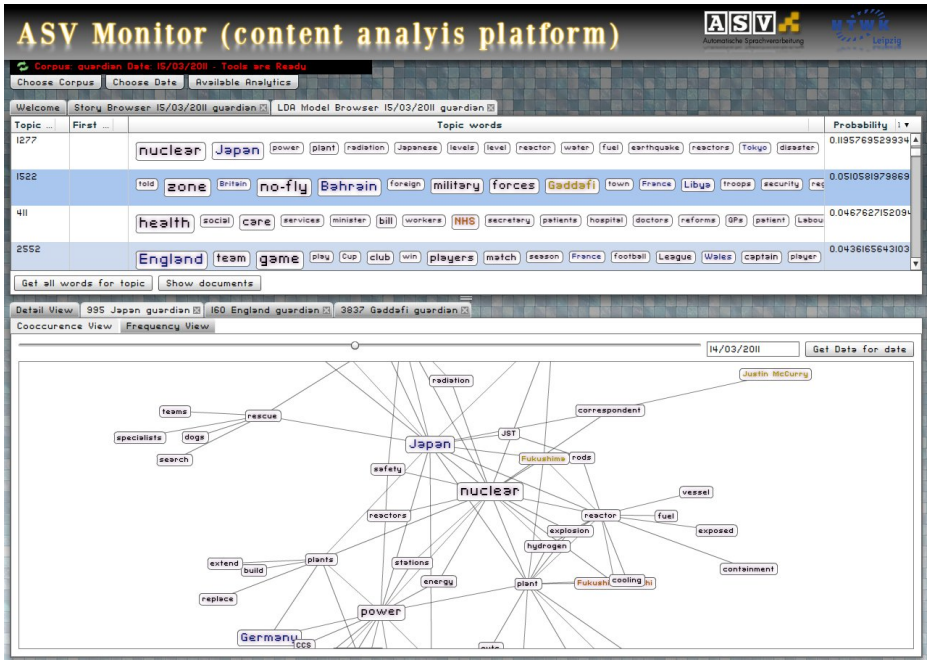

Fig. 1. Screenshot of the main screen: Within the visualisation of summarisations and categories we use word clouds in different ways to visualize results.

implemented approaches. These include terms in different size according to their

[4] http://incubator.apache.org/flex/

importance (as defined by the algorithm), different colors according to their NER labels.

2.3 Comparison

In combination with text sources allocated by the user our aim is to provide means of comparing results of the analysis based on the algorithms and visualisations. This enables the user to compare and judge on the results against a previously known reference corpus. With a well known corpus, it is then easy to choose the best suited algorithm for a given task thus raising the acceptance of automated machine leaning methods. We focus on that task by providing the ability of presenting the different results side by side within the same browser window. We rely on tabbed browsing as a well-known user interface to compare different visualisations to reach that goal.

3 Summarisation and outlook

The ASV-Monitor enables the user to compare the outcome of machine learning algorithms used to solve different content analysis tasks. It also provides an intuitive interface and the ability to use new text sources to evaluate algorithms on them. With this, it is easier to suggest appropriate algorithms and evaluate their weaknesses and strengths concerning certain tasks of content analysis. Our current version only uses algorithms that utilize the text content. Thus, an interesting extension could be the integration of methods that are able to analyze the behavior of time varying units like word frequencies or categories' document counts, as these are another important aspect that can be of use in content analysis e.g. for trend detection. Moreover, the whole domain of time series analysis could provide deeper insight for tasks dealing with diachronic data sources. Evaluation of those methods on real and recent text data provides better support to select the best approach for any given task.

References

1. Krippendorff, K.: Content Analysis : An Introduction To its Methodology. 2nd edn. Sage, Thousand Oaks Calif. (2004)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research **3** (2003) 993–1022
3. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. Journal of the American Statistical Association (2006)
4. Allan, J.: Introduction to Topic Detection and Tracking. In Allan, J., Croft, W.B., eds.: Topic Detection and Tracking. The Information Retrieval Series. Springer US (2002)
5. Heyer, G.: Text Mining: Wissensrohstoff Text : Konzepte, Algorithmen, Ergebnisse. W3L-Verlag, Herdecke [u.a.] (2006)
6. Kleinberg, J.: Bursty and Hierarchical Structure in Streams. In: KDD '02: Proceedings of the eighth ACM SIGKDD. (2002)