Topics2Themes: Computer-Assisted Argument Extraction by Visual Analysis of Important Topics

Maria Skeppstedt[1,2], **Kostiantyn Kucher**[1], **Manfred Stede**[2], **Andreas Kerren**[1]
[1]Department of Computer Science and Media Technology, Linnaeus University, Växjö, Sweden
{maria.skeppstedt,kostiantyn.kucher,andreas.kerren}@lnu.se
[2]Applied Computational Linguistics, University of Potsdam, Potsdam, Germany
stede@uni-potsdam.de

Abstract

While the task of manually extracting arguments from large collections of opinionated text is an intractable one, a tool for computer-assisted extraction can (i) select a subset of the text collection that contains re-occurring arguments to minimise the amount of text that the human coder has to read, and (ii) present the selected texts in a way that facilitates manual coding of arguments. We propose a tool called Topics2Themes that uses topic modelling to extract important topics, as well as the terms and texts most closely associated with each topic. We also provide a graphical user interface for manual argument coding, in which the user can search for arguments in the texts selected, create a theme for each type of argument detected and connect it to the texts in which it is found. Topics, terms, texts and themes are displayed as elements in four separate lists, and associations between the elements are visualised through connecting links. It is also possible to focus on one particular element through the sorting functionality provided, which can be used to facilitate the argument coding and gain an overview and understanding of the arguments found in the texts.

Keywords: Argument extraction, topic modelling, text analysis, argument visualisation, stance visualisation, text visualisation, information visualisation, interaction

1. Introduction

The large amount of opinionated text that is constantly being produced online might help us to better understand why certain opinions are held. For instance, opinions related to consumer behaviour, health decisions or to political and religious radicalisation. These online texts often include arguments to why a particular stance is taken. By extracting these arguments, new insights into reasons and motives for taking this stance might be gained.

It is an intractable task to manually extract arguments from the vast amount of opinionated text that is being produced, e.g., online text in the form of tweets or discussion forum posts, or text in the form of free text answers to survey questions. Fully automatic argument extraction, on the other hand, has been shown difficult, e.g., by Boltužić and Šnajder (2015). Computer-*assisted* methods for argument coding might, however, be a feasible option. For instance, methods that (i) automatically extract the parts of the text collection that contain re-occurring arguments, to minimise the amount of text that the human coder has to read, and (ii) visualise the automatically extracted information in a way that facilitates manual coding.

We here present Topics2Themes, a visual analysis tool for computer-assisted coding of arguments in collections of short, opinionated texts, e.g., online texts.

Topics2Themes is based on previous related research on qualitative text analysis and argument extraction, which will be described more closely in Section 2. This research has shown that coding a subset of a text collection, selected by topic modelling, produces results similar to those obtained by coding the entire text collection (Baumer et al., 2017). There is also previous research on argument extraction that has shown topic modelling to be suitable for semi-automatic extraction of arguments from opinionated texts (Sobhani et al., 2015).

Building on results from these previous studies, we chose topic modelling as the method for selecting which subset of texts in a document collection to manually code. This was implemented through (i) a back-end that uses topic modelling to automatically extract important topics, as well as the terms and texts with which each topic is most closely associated, and (ii) a front-end that presents these texts for manual coding of arguments and visualises associations between topics, terms, texts, and manually coded arguments. This functionality is described in Sections 3.1 and 3.2, respectively.

The design of the graphical user interface is based on previous visualisation research, as well as on an evaluation of the back-end functionality, as described in Section 3.3. Sections 4 and 5 provide a comparison to previous visualisations and suggestions for future extensions of Topics2Themes. Finally, Section 6 concludes this paper.

2. Background

The main functionality of Topics2Themes is based on previous research on qualitative text analysis, argument extraction, and stance detection.

2.1. Qualitative Text Analysis

Among the large body of research on qualitative text analysis (Myers, 2009, pp. 163–180), we here focus on one particular study, conducted by Baumer et al. (2017), on which the main ideas for Topics2Themes are based.

Baumer et al. (2017) used free-text survey responses for performing two totally independent analyses: (i) a grounded theory-based study and (ii) data analysis based on the output of topic modelling. When comparing the output from the topic modelling and the grounded theory analysis, the authors found that "The topic modeling results captured to a surprising degree many of the themes identi-

fied in grounded theory, and vice versa." Each topic produced by topic modelling was, however, often aligned with several grounded theory themes, and each grounded theory theme was typically aligned with several topics produced by topic modelling.

The algorithm used for topic modelling was Latent Dirichlet Allocation (LDA). In addition to the input in the form of a collection of text documents, this algorithm also requires an input parameter in the form of the number of topics that are to be identified in the collection. Ten topics were here requested from the algorithm. The output given from the LDA algorithm is (i) a set of terms from the collection that represents each identified topic, and (ii) a ranking of the text documents according to the probability that an identified topic is present in the document.

As LDA produces different results depending on what random number is used for the initialisation, the LDA algorithm was run 10 times with different initialisations, to verify the consistency of the produced topics.[1]

The results of the topic modelling were presented by showing the 25 most representative terms and the 50 most representative survey texts for each topic. A manual analysis was then performed of this output, through assigning high-level descriptors for each topic. One researcher had to allocate a few hours over two days to perform the topic modelling-based analysis. The analysis based on grounded theory, in contrast, took two researchers several hours of work per week over about two and a half months.

Topics2Themes includes functionality for supporting the procedure of topic modelling-based text analysis that is described by Baumer et al. (2017). The aim is, however, to construct a more generally applicable tool, which can be applied for text analysis of any collection of short texts. The tool should, in addition, provide a graphical user interface with which the results of the topic modelling can be analysed, and which does not require any knowledge of, e.g., programming or topic modelling.

2.2. Argument Extraction

Another addition to the functionality described by Baumer et al. (2017) is that Topics2Themes is mainly meant to be used on opinionated texts, with the aim of extracting arguments.

Topics2Themes can be applied on a text collection for which there is no previous knowledge of what arguments are present. In contrast, most previous studies on argument extraction assume that a set of pre-defined arguments are known, and take on the task of detecting in which debate posts these arguments are used. That is, the argument extraction task is modelled as a standard text classification task. F-scores for the task that range from 0.5 to 0.8 have been reported (Hasan and Ng, 2014; Boltužić and Šnajder, 2014; Sobhani et al., 2015). The results are, however, difficult to compare, since the granularity of the argument categories varies between the different studies.

In the study by Sobhani et al. (2015), topic modelling was applied for carrying out the classification task. Topic modelling was applied to unlabelled data and the extracted topics were then manually mapped to eight pre-defined arguments. The unlabelled data was, thereafter, clustered based on the extracted topics, i.e., a post was assigned to a topic cluster if its probability of containing that topic was above a certain threshold. When the topic modelling approach Non-Negative Matrix Factorization (NMF) was used, an F-score of 0.5 was achieved, which was six percentage points better than the supervised baseline classifier. In contrast, the use of LDA-based topic modelling gave very low results. The authors attribute the difference in results between the two topic modelling approaches to that LDA is better adapted to longer texts than the short discussion posts that were used in the study.

The work by Boltužić and Šnajder (2014) has been extended by performing a hierarchical clustering of argumentative sentences, based on text similarity measured by bag-of-word features and by word embedding features (Boltužić and Šnajder, 2015). The aim of this clustering was to group similar arguments in an unsupervised fashion and thereby automatically come up with the set of arguments that were assumed as pre-defined in the other argument extraction studies. Results achieved for this approach were low, and the authors note that computer-assisted argument extraction, i.e., what we aim for in this study, might be more feasible than a fully automatic extraction.

2.3. Stance Detection

Stance detection is related to the task of argument extraction, but instead of extracting arguments, it is detected which stance is taken. The stance detection task is thus also modelled as a text classification task, typically with the aim of determining whether a text expresses a stance *for*, *against* or whether it is *neutral/undecided* towards a pre-defined proposition or target (Mohammad et al., 2017). Stance classifiers have been trained for detecting stance towards a number of different targets, and on different text genres, including Internet discussion forums (Walker et al., 2012; Hasan and Ng, 2013) and tweets (Mohammad et al., 2017).

As the knowledge of which stance is taken in a text might be useful when performing text analysis for finding arguments, Topic2Themes includes functionality for importing stance information, i.e., each text in the text collection can have a tag attached to it that states if the text is *for*, *against* or *undecided* towards the stance target of interest.

This tagging could either be done automatically by a stance classifier, or manually by annotators. In the case of the texts fed into the tool being free text answers to surveys, the stance tagging could be provided by supplementary closed-ended questions given to the respondents. In the case of two-sided debate texts from online debate forums, stance information is typically provided as meta-data for the debate posts.

3. Functionality of Topics2Themes

Topics2Themes consists of two main parts, (i) the back-end which produces an automatic analysis of a text collection with the help of topic modelling, and (ii) the front-end

[1] Nine topics appeared in all runs. Among them were seven retained, as one of the topics consisted of terms in a non-English language and another was a meta-topic about the survey.

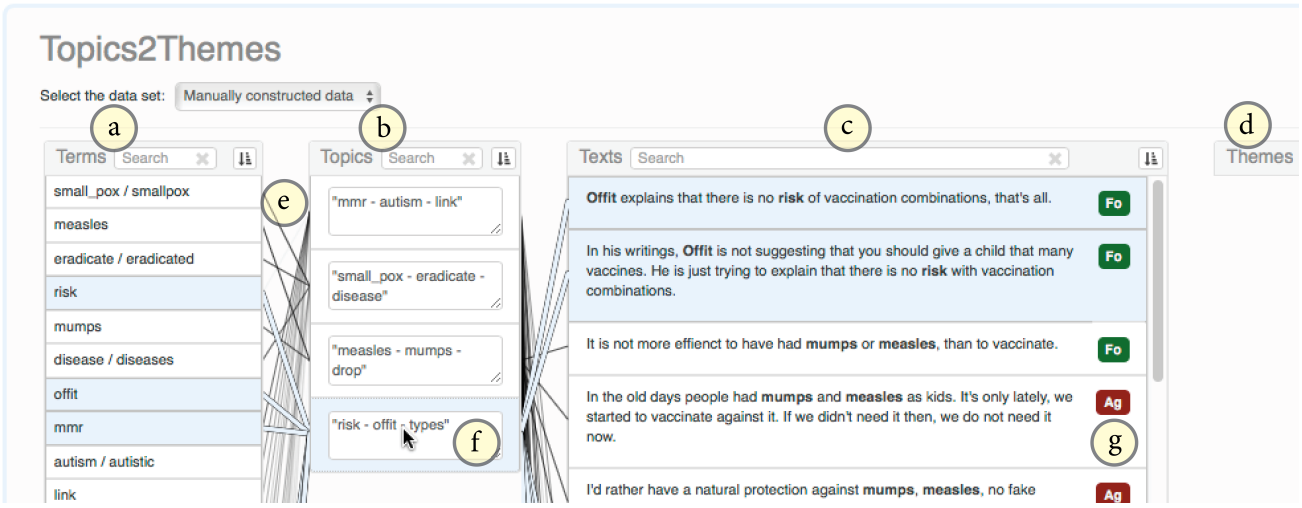

Figure 1: The initial state of the interface, when the topic modelling has been carried out for a text collection, but before any manual coding of themes has been performed by the analyst. The interface includes the following components: (a–d) the panels containing lists of terms, topics, document texts, and user-created themes, respectively; (e) links between the related elements of the respective lists (e.g., terms belonging to a topic); (f) a topic highlighted by the user by hovering; and (g) a stance symbol assigned to the corresponding text.

which consists of the graphical user interface, where the results are presented and where the analysis is carried out.

3.1. Back-end

Functionality to be able to perform the procedure described by Baumer et al. (2017) was implemented in the back-end, but the exact parameters were made user-configurable. That is, the following parameters were made configurable: (i) the maximum number of topics that are to be identified in the text collection, (ii) the maximum number of salient terms to include for each identified topic, (iii) the maximum number of text documents to associate with each topic, (iv) the number of times to re-run the topic modelling algorithm to make sure that the extracted topics are stable, and (v) the amount of overlap between the returned term sets of the different re-runs for a topic to be considered stable.

Some configurable additions to the procedure described by Baumer et al. (2017) were also provided. Since Sobhani et al. (2015) showed that the NMF algorithm is better suited for topic modelling-based argument extraction of short discussion posts than LDA, it was also made configurable whether LDA or NMF is to be used.

In addition, the option to include an English-specific text pre-processing was also provided. This pre-processing consists of (i) a concatenation of collocations that occur frequently in the text collection into one term, and (ii) a replacement of term instantiations of the same concept (morphological variations, synonyms and related terms) with a string that represents the concept. The latter was achieved by applying clustering of word embedding vectors associated with the terms and assigning terms that belong to the same cluster to a joint concept. DBSCAN clustering (Ester et al., 1996) was used, and the maximum distance between two vectors for them to be considered as belonging to the same cluster was also made user-configurable. Examples of the results of the pre-processing is shown in the terms panel in Figure 1(a), where a underscore indicates collocation, and a slash indicates different term instantiations of the same concept.

A standard stop word list is used to remove stop words when constructing the topic models. This list can, however, be extended by the user with domain-specific stop words. In addition, it is possible to configure the automatic removal of frequently or infrequently occurring terms. There is also a user-constructed exception list with terms that are not to be included among the automatically constructed concept clusters. The user can thereby ensure a high quality of the clusters that are used by inspecting a register of automatically constructed clusters and adding terms that have been incorrectly associated with a cluster to the exception list.

Similar to what is described by Baumer et al. (2017), the input to the tool is a collection of text documents. To adapt to the genre of opinionated text, it is also possible to provide a pre-tagging of each text with any of the three stance categories *for*, *against*, or *undecided*.

The back-end was implemented as a RESTful API using the Flask web development framework for Python. The implementations of DBSCAN, text to vector transformation and the two topic modelling algorithms[2] that are available in Scikit-learn were used (Pedregosa et al., 2011). The word embedding vectors were accessed through the Gensim library (Řehůřek and Sojka, 2010), and an out-of-the-box word2vec model[3] trained on Google news was used.

3.2. Front-end

The presentation and user interaction is to support the procedure of qualitative analysis based on topic modelling that is described by Baumer et al. (2017). The tool is mainly aimed to be used on larger collections of short, opinionated texts for extracting arguments. For each of the arguments extracted, it should also be indicated to which stance category the argument is associated. The front-end was im-

[2]Partly following suggestions by Bakharia (2016).

[3]`code.google.com/archive/p/word2vec/`

plemented as a web application with D3[4] to make it easily accessible for the end users.

To provide an example of the functionality of the front-end, we manually authored 50 short texts that were similar in content to discussion posts from British online debates on vaccination, and applied NMF to extract four topics. The example user scenario could thus be that a researcher who studies vaccine hesitancy would like to know what re-occurring arguments *for* and *against* vaccination that are used in online discussions.

We have identified three main user tasks. The user is first to gain an *overview* of what has been extracted by the automatic topic model, thereafter the user is to *analyse the texts* by manually extracting arguments, and finally, the user is to *explore the arguments* that have been extracted.

3.2.1. An Overview of the Topic Modelling Results

After the topic modelling has been carried out, an overview of the model output is provided. Figure 1 shows this initial view which is presented to the user before any manual coding has been carried out. The first panel shows the salient terms, the second one shows the extracted topics, and the third shows the texts (see Figure 1(a–c)). The terms and texts are sorted according to their summed salience for the extracted topics, and could therefore be described as giving an indication of what is generally important in the text collection. The associations between terms/topics/texts and the strengths of the associations are shown through connecting links with different widths, as displayed in Figure 1(e). The figure only shows the top-ranked terms and texts, but lower-ranked terms/texts can be reached in the tool by scrolling down.

As described by Baumer et al. (2017), the set of salient topic terms is not enough to determine the content of a topic. The analyst also needs to be provided with typical text examples. The interface, therefore, gives equal importance to presenting the texts that are associated with a topic as to presenting the terms with which it is associated. When the user lets the mouse hover over a term/topic/text element (for instance, the "risk—offit—types" topic element in Figure 1(f)), its associated elements within the other two categories are highlighted, which makes it possible for the user to explore connections between the three categories. The user can also select any element by mouse click. This has the effect that the elements that belong to the other categories and that are associated with the selected one are sorted as the top-ranked elements within their respective panels. In Figure 2(a), the user had previously clicked on the topic element that is named "smallpox—eradicate—disease", which has had the effect that the associated elements in the other panels are shown on top.

The number of elements in the topics panel shows the user how many topics have been extracted by the topic modelling. Each topic is given a default name that is made up of the three terms with which the topic is most closely associated. The name can, however, be changed by the user to one that better describes the topic.

The "stance symbol" in the right upper corner of a text element indicates the stance category of the text: for instance, in Figure 1(g) the label "Ag" with red background represents the *against* stance. By scrolling through the texts that are connected to a topic, an overview of its associated stances is achieved.

3.2.2. Extract Arguments from Texts

Baumer et al. (2017) found that grounded theory-based themes and topic modelling-based topics did not correspond one-to-one, but with the relation many-to-many. To be able to follow this procedure, the user must be able to define additional categories to the ones automatically extracted by the topic model. A functionality to add an additional category of elements, which are user-defined, is therefore included in Topics2Themes. When referring to these categories that the user creates as elements in the right-most panel, we adhere to the grounded theory-inspired vocabulary used by Baumer et al. (2017) and call these categories "themes". However, since the main purpose of the tool is to extract arguments, these manually extracted themes typically correspond to arguments detected in the texts. Figure 2 shows the tool after the manual user coding has started and themes have also been added. With the "+" button on the themes panel displayed in Figure 2(b), the user can create a new theme. The theme can then be given a description, and texts can be associated with it by drag-and-drop of a text element onto a theme element, as shown in Figure 2(c–d).

In the typical use case, the user extracts arguments from each one of the topics in turn. In the example of Figure 2(a), the user analyses texts that are associated with the topic "smallpox—eradicate—disease", and this topic element is selected. The procedure of detecting arguments is then to analyse each of the texts that are associated with the topic. Terms that are associated with a topic are written in a bold-faced font, which makes them stand out from the rest of the text and which facilitates the analysis (see Figure 2(c)). If an argument is found in a text, the user has two choices: (i) if it is an argument that has previously occurred in the analysis, the text should be assigned to the matching theme that contains this argument, or (ii) if the argument is new, a new theme should be created for this argument, and the user should assign the text to this new theme.

3.2.3. Explore the Arguments

The final task is to explore the arguments that have been created. This task needs to be carried out during the analysis in order to find out whether a text contains an argument that has previously been created. It can also be carried out when the analysis is finished to gain an overview and understanding of the arguments found in the text collection.

Figure 3 shows when the user has selected a theme in order to investigate the argument for which this theme was created. The terms, topics and texts that are associated with this theme are then sorted as the most high-ranked elements in their respective panels. The descriptive text of the theme, as well as the information of with which topic(s) and term(s) it is associated, gives a high-level understanding of the theme. Reading the texts with which the theme is connected, on the other hand, gives a deeper understanding of the theme.

[4] d3js.org

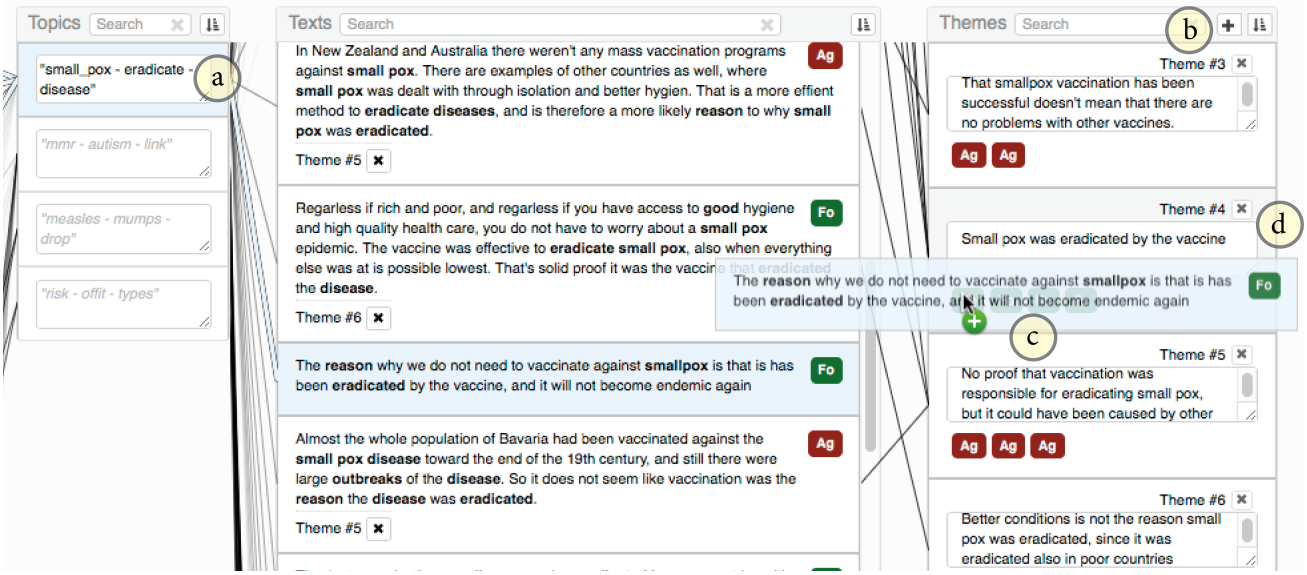

Figure 2: The user interface during the analytical session: (a) the user has focused on a specific topic by clicking; (b) the user has then created several themes by using the button in the themes panel; (c) the user is dragging a document text element to a theme element to create an association; (d) the target of the drag-and-drop operation is theme #4.

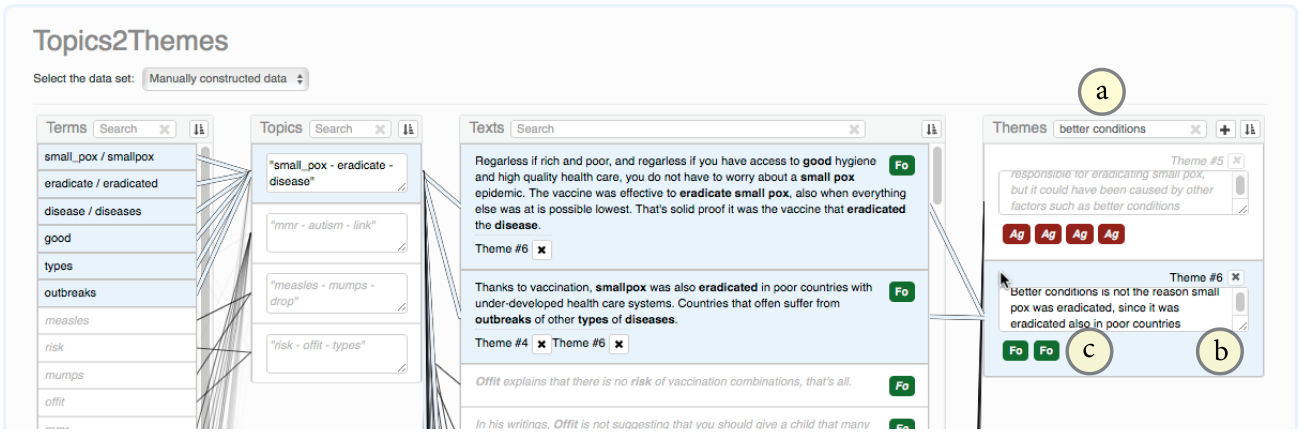

Figure 3: Exploring arguments: (a) Certain themes are searched for; (b) theme #6 is selected by clicking to show with which terms, topics and texts it is associated; (c) stance symbols indicate stances taken in the texts connected to a theme.

For each of the texts that are associated with a theme, the stance symbol of the text is displayed at the bottom of the theme element, as shown in Figure 2(e). These symbols have two functions: (i) as indicators of to which stance category the argument is typically associated, and (ii) to show how frequently the argument occurs in the texts that have been analysed.

3.3. Design Decisions for the Front-end

The main inspiration for the front-end presentation of Topics2Themes was the List View visualisation of the Jigsaw system (Stasko et al., 2008). We also carried out a manual evaluation of the functionality of the back-end part of Topics2Themes in order to gain further inspiration for how the front-end was to be designed.

3.3.1. The Jigsaw System

Jigsaw aims at helping analysts to search, review, and understand the content of a text collection, e.g., a collection of case reports in a police investigation. The main functionality of Jigsaw consists of an interactive visualisation that focuses on identifying and highlighting connections between entities present in the documents. Typical entities are people, places and dates that are automatically detected by a named entity recognition system. A connection between two entities means that they co-occur in the same document. The List View displays these automatically extracted entities in the form of vertical lists. Connections between different entities are displayed by lines that connect the list elements and by highlighting of the elements. Jigsaw also provides several ways of sorting the lists.

One of the main cues for exploring the output of a topic modelling algorithm, as well as for exploring the result of the text coding, is to explore associations, i.e., connections between terms, topics, texts and the themes created. Since the Jigsaw List View is focused on displaying connections, we assess that a similar functionality is suitable for visualising the results of the topic modelling and the manual coding. Instead of using the List View design for displaying entities, we thus display terms, topics and themes, as well as the actual texts, and how they are connected.

3.3.2. Evaluation of the Back-end Functionality

In order to come up with further ideas of how to design the user interface, we first implemented the back-end functionality of Topics2Themes. We thereafter used the topic modelling output of the back-end to perform a manual coding, i.e., a coding without the help of a graphical user interface, according to the procedure described by Baumer et al. (2017). As data, we used a previously compiled resource with debate posts from the British online debate forum Mumsnet[5] (Skeppstedt et al., 2017). The resource consists of 1,190 debate posts from six discussion threads on the topic of vaccination, where the posts are manually classified as expressing a stance *for* or *against* vaccination or as being *undecided*. The back-end was configured to use NMF to extract ten topics, with a required overlap of 70% between ten re-runs of the algorithm in order for a topic to be considered stable. This resulted in six stable topics. The 50 most typical texts, for each one of the six topics identified by the topic model, were coded, one topic at a time. Thus, a set of re-occurring arguments was identified for each one of the topics (38, 23, 51, 61, 40, and 33 arguments, respectively). There was a large semantic coherence between the texts that were selected for analysis for a topic, and there were only occasional occurrences of arguments that were identified as associated to more than one topic (Skeppstedt et al., 2018).

The main difficulty of the manual analysis was the cognitive load of remembering which arguments had been previously identified. To be able to go through a set of semantically coherent texts, i.e., those that belonged to the same topic, limited the set of possible arguments to look for, and, thereby, also led to a decreased cognitive load.

It can thus be observed that the use of topic modelling for text selection and sorting has two main benefits. First, only a subset of the document collection has to be read in order to find re-occurring arguments, which makes it possible to analyse large text collections also when the time available is limited. Second, the possibility to focus on one topic at a time facilitates the analysis, as the user only has to remember the limited set of arguments that have previously been extracted for the topic that is currently being analysed.

The cognitive load of remembering previously defined arguments was, however, still too large, despite the support from the topic modelling. With the themes panel in Topics2Themes, we therefore aim to further decrease this cognitive load. The extracted arguments/themes are not only listed with a description, but it is also easy to use the interface to explore the arguments that have been extracted previously. These arguments can, for instance, be sorted according to whether they are associated with other texts that contain certain terms, e.g., the terms of the text that is currently being analysed. In addition, in order not to add to the cognitive load required for the task of extracting and remembering arguments, we aimed to construct a clean interface with a minimum of distractions.

From the explanation given by Baumer et al. (2017), we had interpreted the coding task as a task of associating topics to a corresponding theme. Although the authors emphasise the importance of reading the texts for finding themes, our draft interface provided the functionality of associating a topic to a theme. The manual coding showed, however, that although the topics are important for sorting and selecting texts and for giving a topical focus for the analysis, the user does not associate topics directly with themes. Instead, when analysing the texts, the coder associates the *text* to a theme. In Topics2Themes, the task of the user is, therefore, to associate a text to a theme.

The design choice of associating texts to themes also has the advantage of providing a better traceability of the analysis performed. It is still easy to obtain high-level information in the form of which themes are associated with a particular topic, i.e., a topic can be selected and all associated elements in the themes list can be examined. If the analyst instead would like to read the original text in order to trace the reasons to why a certain theme has been identified, the current design enables the texts, from which the theme originated, to be easily identified.

4. Comparison to Previous Visualisations

There are a number of examples of related visualisation tools, but to the best of our knowledge, there is no previous tool that offers the support for computer-assisted argument extraction and the same type of overview of associations between elements that is provided by Topics2Themes.

The output of topic models has previously been visualised and made available for user interaction. The Termite tool (Chuang et al., 2012), for instance, aims at enabling quality assessments and improvements of produced topic models. Salient terms and topics are shown in a grid that indicates which terms belong to which topics. The Serendip tool, in contrast, has exploration of texts as the main aim, and the grid view design is used there instead for displaying topics and their connected documents, which can be sorted according to a number of different criteria (Alexander et al., 2014). Visualisations for showing alignment between two sets of topics (Chuang et al., 2013), as well as non-interactive visualisations (Tangherlini et al., 2016), are other examples of topic model visualisations.

The focus of these previous approaches is, however, not to use the output of the topic models as support for text coding. Therefore, the functionality of Topics2Themes that supports computer-assisted argument extraction is not available. For instance, the functionality of adding new categories in the form of themes, displaying stance, or the functionality of exploring data through selecting an element to show with which other elements it is associated.

There are also different types of visualisations of stance taking in text, as well as of the related concept of sentiment in text. For instance, visualisations to show changes in sentiment over time, the text elements that have resulted in the stance/sentiment classifications carried out by the underlying natural language processing tools, or the topics/targets towards which the sentiment is directed (Kucher et al., 2018). There is also work on the use of topic modelling for extraction of topics from text, as well as visualisations of sentiment towards these extracted topics (Hoque and Carenini, 2014). The aim of extracting topics in these previous studies is, however, very different from the aim

[5] www.mumsnet.com/Talk

of the topic modelling in Topics2Themes. These previous studies extract topics to find out towards what the stance or sentiment is directed, whereas Topics2Themes is to be applied on texts where stance related to a pre-defined issue is expressed. The aim of topic modelling for Topics2Themes is instead to facilitate the process of argument extraction from these texts.

There are also previous tools that are specifically aimed at visualising argumentative text. A tool constructed by Wyner et al. (2015) helps an analyst to extract important arguments by highlighting information in the text that might be relevant for this task. This information includes named entities, terms important for the domain of the text and for expressing sentiment, as well as terms belonging to topics derived from LDA-based topic modelling. In contrast to Topics2Themes, the tool by Wyner et al. (2015) is not focused on analysing the text on the basis of extracted topics. Therefore, the tool does not include the functionality of ranking the text sections according to their relevance for the extracted topics. The potential of saving time in the analysis through only reading the most important parts of the text, as in the process described by Baumer et al. (2017), is, thereby, not achieved.

The VisArgue framework is another example of visualisation for argumentative text (El-Assady et al., 2016). VisArgue uses topic modelling (among other techniques) for visualising arguing patterns. In contrast to Topics2Themes that aims at facilitating extraction of re-occurring arguments, VisArgue aims at providing an overview of an entire debate or of an entire single argumentative document with respect to topic changes throughout the document and in the course of the debate. There are also other aspects of the argumentative genre that can be visualised, and that are not included in the aims for Topic2Themes, for instance, the quality of the argumentation (Gold et al., 2017).

5. Future Directions

The next step to generate new ideas for improvement is by extensively using the front-end part of Topics2Themes for text coding. After the initial use of the tool, we already have a number of ideas for possible extensions.

A subset of a text collection might consist of longer texts, even in a text genre that mainly contains shorter texts, e.g., the genre of discussion forum posts. These texts are not suitable to show "as-is" in the text panel of Topics2Themes, since this panel is meant to give an easily scrollable overview of several texts from the collection. We therefore aim to instead show summaries of the longer texts when the user scrolls through the text panel. The user should also be provided with the option to associate a theme to a substring of a longer text. This would give the user the possibility to trace the origin of an extracted argument to an exact text snippet in a longer text.

Another important functionality to add is the possibility to set topic modelling parameters through the user interface. These parameters are currently changed through text-based property files, which is not optimal from a usability perspective. In addition, the functionality for visualising the connections between terms and their corresponding topics and texts might be useful for determining some of the topic modelling parameters, e.g., whether a term should be included in the stop word list. The visualisation of term associations could also be useful for determining which terms to exclude from the automatically constructed concept clusters. The user interface should, therefore, also be extended by functionality for fine-tuning of the concept clusters.

The assignment of themes might be further facilitated by automatically sorting previously created themes according to the likelihood of them being associated with the text that is to be analysed. For a typical use case, only a small number of texts are manually associated with each theme, which makes it difficult to train a high-precision machine learning classifier to automatically associate texts to themes. To train a classifier to rank themes might, in contrast, be feasible even with a small training data set.

Finally, since Topics2Themes is meant to be applied on online texts, it could also be relevant to add newly produced texts to an existing text collection. Support for including newly detected topics into an existing analysis must then be added, as well as support for visualising changes in the prevalence of different topics over time.

6. Conclusion

We here presented Topics2Themes[6], a tool for carrying out computer-assisted coding of arguments in opinionated text. The tool is meant to be applied on larger text collections consisting of short texts, for instance, online texts in the form of tweets or posts from Internet discussion forums.

Topics2Themes uses topic modelling to automatically extract frequently occurring topics in the text collection. A subset of the collection, formed by texts most likely to discuss the extracted topics, is presented for manual coding. The coding of large text collections using limited manual resources is thereby made possible.

The coding is further facilitated by the graphical user interface provided by Topics2Themes. The user is able to create theme elements for each detected argument type, and to associate these elements to the texts in which the arguments occur. Associations between the automatically extracted elements and the manually coded elements are visualised, i.e., associations between the topics, terms representing the topics, texts and themes. Thereby, an overview of the text collection content is provided, as well as of the arguments that are used in the text collection.

7. Acknowledgements

We would like to thank the reviewers for their valuable input. We would also like to thank the Swedish Research Council (Vetenskapsrådet) that funded this study, mainly through the project "Navigating in streams of opinions: Extracting and visualising arguments in opinionated texts" (No. 2016-06681) and partly through the StaViCTA project, framework grant "the Digitized Society – Past, Present, and Future" (No. 2012-5659).

8. Bibliographical References

Alexander, E., Kohlmann, J., Valenza, R., Witmore, M., and Gleicher, M. (2014). Serendip: Topic model-driven

[6] A supplementary video demo of Topics2Themes is available at `https://vimeo.com/257474950`

visual exploration of text corpora. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*, VAST '14, pages 173–182. IEEE, October.

Bakharia, A. (2016). Topic modeling with Scikit Learn. https://medium.com/@aneesha/topic-modeling-with-scikit-learn-e80d33668730 (Accessed January 10, 2018), September.

Baumer, E. P. S., Mimno, D., Guha, S., Quan, E., and Gay, G. K. (2017). Comparing grounded theory and topic modeling: Extreme divergence or unlikely convergence? *Journal of the Association for Information Science and Technology*, 68(6):1397–1410, June.

Boltužić, F. and Šnajder, J. (2014). Back up your stance: Recognizing arguments in online discussions. In *Proceedings of the First Workshop on Argumentation Mining*, pages 49–58, Stroudsburg, PA, USA, June. ACL.

Boltužić, F. and Šnajder, J. (2015). Identifying prominent arguments in online debates using semantic textual similarity. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 110–115, Stroudsburg, PA, USA, June. ACL.

Chuang, J., Manning, C. D., and Heer, J. (2012). Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 74–77. ACM.

Chuang, J., Hu, Y., Jin, A., Wilkerson, J. D., McFarland, D. A., Manning, C. D., and Heer, J. (2013). Document exploration with topic modeling: Designing interactive visualizations to support effective analysis workflows. In *NIPS Workshop on Topic Models: Computation, Application and Evaluation*.

El-Assady, M., Gold, V., Hautli-Janisz, A., Jentner, W., Butt, M., Holzinger, K., and Keim, D. A. (2016). VisArgue: A visual text analytics framework for the study of deliberative communication. In *Proceedings of the International Conference on the Advances in Computational Analysis of Political Text*, PolText 2016, pages 31–36. University of Zagreb.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, KDD '96, pages 226–231, Palo Alto, California, USA. AAAI Press.

Gold, V., El-Assady, M., Hautli-Janisz, A., Bögel, T., Rohrdantz, C., Butt, M., Holzinger, K., and Keim, D. (2017). Visual linguistic analysis of political discussions: Measuring deliberative quality. *Digital Scholarship in the Humanities*, 32(1):141–158, April.

Hasan, K. S. and Ng, V. (2013). Stance classification of ideological debates: Data, models, features, and constraints. In *Proceedings of the International Joint Conference on Natural Language Processing*, IJCNLP '13, pages 1348–1356, Stroudsburg, PA, USA. ACL.

Hasan, K. S. and Ng, V. (2014). Why are you taking this stance? Identifying and classifying reasons in ideological debates. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 751–762, Stroudsburg, PA, USA, October. ACL.

Hoque, E. and Carenini, G. (2014). ConVis: A visual text analytic system for exploring blog conversations. *Computer Graphics Forum*, 33(3):221–230, June.

Kucher, K., Paradis, C., and Kerren, A. (2018). The state of the art in sentiment visualization. *Computer Graphics Forum*, 37(1):71–96.

Mohammad, S., Sobhani, P., and Kiritchenko, S. (2017). Stance and sentiment in tweets. *ACM Transactions on Internet Technology (TOIT)*, 17(3):1–23, June.

Myers, M. D. (2009). *Qualitative research in business & management*. SAGE, London.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Řehůřek, R. and Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Paris, France, May. European Language Resources Association (ELRA).

Skeppstedt, M., Kerren, A., and Stede, M. (2017). Automatic detection of stance towards vaccination in online discussion forums. In *Proceedings of the International Workshop on Digital Disease Detection using Social Media*, pages 1–8, Stroudsburg, PA, USA, November. ACL.

Skeppstedt, M., Kerren, A., and Stede, M. (2018). Vaccine hesitancy in discussion forums: Computer-assisted argument mining with topic models. Accepted for publication at Medical Informatics Europe.

Sobhani, P., Inkpen, D., and Matwin, S. (2015). From argumentation mining to stance classification. In *Proceedings of the 2nd Workshop on Argumentation Mining*, Stroudsburg, PA, USA. ACL.

Stasko, J., Görg, C., and Liu, Z. (2008). Jigsaw: Supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132.

Tangherlini, T. R., Roychowdhury, V., Glenn, B., Crespi, C. M., Bandari, R., Wadia, A., Falahi, M., Ebrahimzadeh, E., and Bastani, R. (2016). "Mommy blogs" and the vaccination exemption narrative: Results from a machine-learning approach for story aggregation on parenting social media sites. *JMIR Public Health Surveill*, 2(2):e166, November.

Walker, M. A., Anand, P., Abbott, R., and Grant, R. (2012). Stance classification using dialogic properties of persuasion. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 592–596, Stroudsburg, PA, USA. ACL.

Wyner, A., Peters, W., and Price, D. (2015). Argument discovery and extraction with the Argument Workbench. In *Proceedings of the 2nd Workshop on Argumentation Mining*, Stroudsburg, PA, USA. ACL.