LTMA: Layered Topic Matching for the Comparative Exploration, Evaluation, and Refinement of Topic Modeling Results

Mennatallah El-Assady[1,2], Fabian Sperrle[1], Rita Sevastjanova[1], Michael Sedlmair[3], and Daniel Keim[1]

[1]University of Konstanz, Germany
[2]University of Ontario Institute of Technology, Canada
[3]University of Stuttgart, Germany

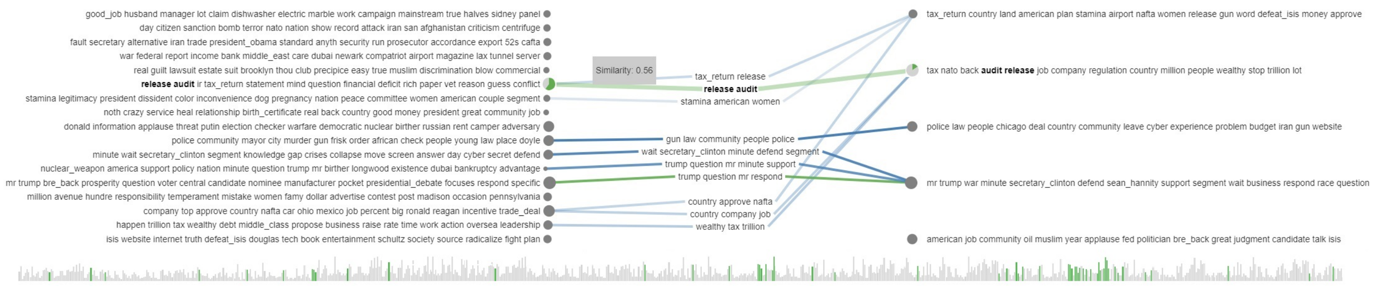


Fig. 1: Topic-match selection, with highlighting of the similarity score, the identical keywords of the selected pair, and the document-overlap (in the bottom bar-chart). The matched topics are from an IHTM [18] with 17 topics on the left, and an LDA [7] model with 5 topics on the right, based on a corpus of presidential debates between *Trump and Clinton* (2016). The number of documents attributed to each topic is mapped to the size of the circle beside the topic, while the matching topic descriptors are shown in-between the results. The relative amount of overlapping documents compared to all documents assigned to a topic, is shown using a pie-chart beside the hovered topic pair. Two matching topics are connected through a line, displaying by its color the type of match: complete-match, similarity-only, or document-overlap-only (not available in current example).

***Abstract*—We present LTMA, a Layered Topic Matching approach for the unsupervised comparative analysis of topic modeling results. Due to the vast number of available modeling algorithms, an efficient and effective comparison of their results is detrimental to a data- and task-driven selection of a model. LTMA automates this comparative analysis by providing topic matching based on two layers (document-overlap and keyword-similarity), creating a novel topic-match data structure. This data structure builds a basis for model exploration and optimization, thus, allowing for an efficient evaluation of their performance in the context of a given type of text data and task. This is especially important for text types where an annotated gold standard dataset is not readily available and, therefore, quantitative evaluation methods are not applicable. We confirm the usefulness of our technique based on three use cases, namely: (1) the automatic comparative evaluation of topic models, (2) the visual exploration of topic modeling differences, and (3) the optimization of topic modeling results through combining matches.**

I. Introduction

In many disciplines such as the humanities and social sciences, the efficient analysis of large quantities of text on a regular basis is essential. To better support these tasks, various automatic text summarization and content modeling algorithms have emerged in recent years. One of the most widely used techniques for gaining insight into the content structure of text corpora is through browsing the results of topic modeling algorithms applied to these texts. By automatically generating overviews of the thematic structure of the texts, the detection of interesting documents and their comparison is facilitated. For a large corpus of documents, such as a collection of news articles, a topic modeling algorithm provides a list of topics, such as politics, economy, or sports. Each topic, in turn, is defined by a set of descriptive words that are ranked according to their importance for the topic. Typically, the output of a topic modeling algorithm consists, besides the topic keyword vectors, of topic distribution matrices over the documents of the corpus.

Due to the vast amounts of topic modeling algorithms that emerged in recent years, a quantitative result comparison of two algorithms on a given corpus would enable scholars to objectively choose the most suitable algorithm for their dataset. The objective and automatic quantification of differences between topic modeling results can open the door for various new applications. Besides guiding users in the choice of suitable topic modeling algorithms for their data, use cases for matching the results of two topic modeling algorithms include: the automatic comparative evaluation of topic models, the visual exploration of topic modeling differences, and the optimization of topic modeling results through combining matches [17].

In this paper, we introduce an algorithm for comparing the results of topic models and calculating the similarities between

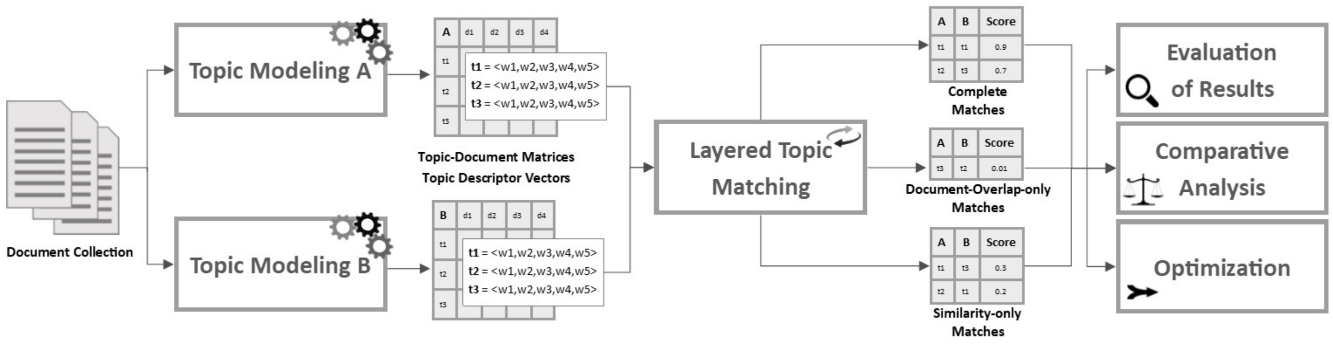


Fig. 2: Processing pipeline to compute topic matches using the results of two topic modeling algorithms. Starting with a document collection, the topic modeling algorithms A and B are applied, resulting in a set of topic descriptor vectors and a topic document matrix for each topic model. These four components are the input of the layered topic matching algorithm, which computes matches of three types: complete matches, similarity-only matches, and document-overlap-only matches. Together with the topic modeling results, the computed match data structure enables applications for comparative, and explorative analysis.

two sets of given topics defined over the same corpus. The ***Layered Topic Matching Algorithm (LTMA)*** is designed to analyze topic matches of three different types. The first type is the *complete match*, where the keywords of two topics, as well as their respective documents, match. The second, and weaker, type is the *similarity-only match*, where only the keywords of two topics match. The third type defines a *document-overlap-only match* of two topics as a strong overlap of their document vectors, but significantly low similarity of their keywords. This can be seen as a descriptor *mismatch*, since the two topics disagree on the labeling of a highly-overlapping set of documents. For the measurement of the match degree, three indicators are computed: the normalized similarity of the keyword vectors, the document match frequency with the first topic, and with the second topic, respectively. These indicators were derived from an observational study with social scientists. The algorithm is therefore designed to capture the intuition of human analysts and combine it with the efficiency of automatic computation. The resulting data structure enables a wide spectrum of new applications, as examplified in Section V.

II. Related Work

Comparative analysis of topic models is an open research challenge that has gained little attention so far [5]. We hence focus our discussion of related work on topic models in general, with an eye towards how humans interpret topic modeling results [9], how topic models are typically evaluated [37], and the visual task-driven comparison of topic models [2].

A. Topic Modeling

Emerging from the general task of document clustering, topic modeling approaches aim at uncovering the underlying thematic structures of document collections. Topic modeling narrows down the general document clustering goal from partitioning the documents into meaningful groupings, to generating clusters with the same topical structure. Furthermore, topic models extend the task of grouping and organizing a document collection to additionally labeling these groupings with meaningful descriptors that reflect their content. In theory, topic modeling algorithms can be classified as conceptual document clustering approaches [22]. These models have been used in various domains to answer different research questions. The main tasks associated with topic models are summarizations, exploration, and overview-generation of text corpora [1]. Hence, topic models ought to produce compact, structured, and accurate results.

The most common types of topic modeling algorithms are generative models that assume an underlying distribution of documents in the concept space and attempt to estimate this distribution iteratively [23]. Of those, the most widely-used algorithm is the Latent Dirichlet Allocation [7] (LDA). It uses a bag-of-words model and assumes that the order of words in a document holds no relevant information. One shortcoming of LDA, the fixed number of topics as a hyper parameter, is overcome by the Hierarchical Dirichlet Process [36] (HDP), that can learn the correct number of topics from the data. Other probabilistic models account for topic changes over time [6], correlations between topics [29], or authorship information [32].

Currently, there are very few non-probablisitic topic modelling approaches. The most prominent representative is Non-negative Matrix Factorization [26] (NMF), that has only recently been applied to topic modeling [3]. It decomposes a document collection into a term-by-topic and a topic-by-document matrix, using a parameter to set the expected number of topics. This model has been successfully used to build an interactive visual analytics system [10] for exploring the document space, allowing topic refinements in real-time. The Bayesian Rose Tree [8] is a hybrid approach combining probabilistic topic modeling and hierarchical clustering. Based on the initial results of a flat topic structure generated by a probabilistic model, the hierarchical structure of the rose tree is built. This model has been adapted by Lui et al. [28] and is used as a basis for the creation of an interactive visual analytics system for news exploration [15].

A general discussion on topic models and their differences can be found in the survey of David M. Blei on probabilistic topic models [5] and the book “Mining Text Data” [1].

B. Comparative Analysis of Topic Models

Recent approaches have emerged that visually compare topic modeling results. Crossno et al. [13] introduced "TopicView", a visualization approach for comparing results of multiple topic models. Similarly, Chuang et al. [12] presented "Termite", a matrix-based visualization for assessing the differences between topics. Alexander et al. presented an approach for the task-driven comparison of topic models [2]. "ParallelTopics" by Dou et al. [14] and "LDAExplore" by Ganesan et al. [19] show probabilistic topic distributions in a parallel coordinates plot. While the first focusses on placing similar topics closely together, the latter also visualizes the extend to which each document belongs to its assigned topics. In contrast, our approach quantifies the similarity between topic pairs and generates topic matches by examining not only the descriptor similarity of two keyword vectors but additionally the document overlap of a topic pair. Aside of the visual comparison of topic models, approaches have been presented for exploring different topic models through computing their topic coherence [34], by offering a visual query interface [16] or by generating topic alignments [11]. The latter approach is the most relevant to our paper. However, in this work the topic matches are defined as topic-concept pairs, i.e., matching the topics to a defined thematic concept rather than automatically generating topic-pair matches as computed by *LTMA*. By automatically generating topic matches as proposed by our approach, we do not require any prior knowledge about the content of the analyzed corpus. This method has been successfully deployed as a basis for the visual workspace of our previous work [17], and enabled a guided, comparative topic model optimization.

C. Similarity Measures

To group documents into different topics, the content similarity of the texts has to be captured. The content or semantic similarity is a *"metric defined over a set of documents or terms to measure the distance between them based on the likeliness of their meaning or semantic content"* [20]. According to Tan et al. [35], the three most powerful text similarity functions are the cosine similarity, the Jaccard similarity, and Dice's coefficient. The main advantage of the cosine similarity over the others is that it is robust towards heterogeneous dimensions of documents and topics, allowing a stable comparison of shorter and longer vectors.

In contrast to text similarity measures, for generating topic matches it is essential to compare descriptor keywords based on their rank. This rank similarity of two vectors is closely related to the content similarity of text, however, it increases the impact of the order of the keywords. Examples of rank similarities and distances for text include the minimum edit distance [31], the longest common sub-sequence distance [4], and the Hamming distance [21]. In the domain of statistics other rank correlation coefficients are prominent, such as Spearman's rho [33] or Kendall's tau [24]. However, both the rank distances and rank correlation coefficients consider only one of two aspects, either the order of the words in a vector or their similarity. In this paper, we propose a novel weighted keyword distance measure to compare topic descriptor vectors.

III. Requirements Analysis

Through observing the usage of topic models by humanities and social science scholars, we identified a common approach to validate or optimize their results. Over the course of two years, we have been working closely with six different scientists on a joint project where topic modeling algorithms have been used extensively. This work included weekly meetings and analysis sessions which lead us to our requirement analysis. Most scholars in our observational study declared that when using topic models, they usually read large parts of the texts and manually label documents and topics to ensure that the quality of the results satisfies their expectations. Another common method for creating reliable modeling results is to manually match the result of two well-established topic modeling algorithms to assess their quality or optimize their results. Since manually matching two topic modeling results is a subjective and highly time-consuming task, there is a considerable demand for the development of an automatic matching approach that incorporates the intuition of a person assigned to this task.

As derived from our observational study, each topic match has to fulfill two underlying assumptions to capture the analysts' intuition. First, the **descriptor keywords similarity**: Keywords of the descriptor vectors of both matched topics have to refer to similar concepts (similar topic descriptor vectors). Second, the **document distribution overlap**: Two topics match if their respective document vectors (vectors of the documents assigned to the topics) match. Setting these assumptions as fundamental requirements for the design of an automated topic matching approach, the *Layered Topic Matching Algorithm* was developed.

As shown in Fig. 2, *LTMA* operates on the results of two topic modeling algorithms that process the same document collection. These results are matched in three different types. (1) The **Complete Match;** with a strong match of two topics, where both requirements are fulfilled. (2) The **Similarity-Only Match;** as a weaker match, where the assumptions are relaxed, and only the first requirement is fulfilled. A third type can be defined in which only the second requirement is fulfilled. However, this (3) **Document-Overlap-Only Match** is more a mismatch of two topics, since it features a strong overlap of documents despite a significantly low similarity of keywords.

One of the main challenges in designing the algorithm was to capture the similarities between two keyword vectors as human analysts would perceive them. As discussed in Section II-C, different measures are used to define the concept-similarity between document collections. However, our first attempts at computing topic matches using common semantic and rank similarity measures produced too few or too many matches when compared to the manually annotated results. This is due to their definition of similarity as exact vector similarity, weighted only by frequency counts, not explicitly by the ranks of the keywords in the descriptor vector. To address this problem, a novel similarity measure is proposed in IV-A that relies on measuring a weighted distance of the two vectors, adding a weighted penalty to the distance if a keyword occurs only in one vector. It is worth noting that despite being tailored to capturing the analysts' intuition, our similarity function does not include any additional semantic resources to expand the linguistic context of keyword vectors. Hence, while we observed that humans also capture the similarity of two words through their language-understanding, we did not attempt to model this. This is due to the fact that our approach is designed to operate on the same corpus for the two models, and as such will not incur the effects of different vocabulary, by definition.

Furthermore, another result of our observational study was that in most cases, our experts examined the top 10 to 15 descriptors of a topic while analyzing the modeling results. Also, they typically considered only the top topic for every document, regardless of the type of the underlying algorithm. While these assumptions might be advantageous when comparing a non-probabilistic model to other results, for probabilistic distributions, attributing the most likely topic to a document, is unfavorable. Therefore, when comparing the results of two probabilistic models, leveraging the topic distribution using, for example, the Kullback-Leibler divergence measure [25] might outperform the expert matching. However, in this work, we are focused on deriving an automatic and objective measurement of topic matches as perceived and utilized by humanities and social science scholars. The output of *LTMA* has been made accessible to our collaborators through a web-browser-based, interactive, visual interface which enables them to explore the effects of different parameter settings of the algorithm and adjust them, as well as, analyze the resulting layered topic matches.

IV. Layered Topic Matching Algorithm

LTMA is designed to compute the similarity between topics from different models to form topic matches. These matches can be used as an objective measure for quantifying similarities of topics, based on the perceived similarity of their keyword vectors and their document overlap. In our observational study, we found that manual matching of topic modeling results was based on two factors: the comparison of the different topic descriptor vectors and the analysis of document-overlaps between topics. Subsequently, a strong match is defined as one that consists of a topic pair that has similar keywords and a large document overlap. Based on this, *LTMA* follows three steps to compute matching topic pairs, as well as their degree of similarity and document overlap. In the following, we will describe these steps in more detail, as shown in Algorithm 1.

Step 1: Calculate topic similarities and document overlaps

The distances of the keyword vectors and the document overlap are calculated for each topic pair. The distances between the descriptor vectors build the basis for creating a similarity-based matching and are calculated by a novel method for weighted keyword distance measurement between topic descriptor vectors, as described in Section IV-A. As for the document-overlap-based matching, the intersection of both respective document vectors is computed in this first step.

```
Data: Set of Topics1 , Set of Topics2
Result: List of Topic Matches
Algorithm TopicMatching($top_1$, $top_2$)
       // Calculate similarities & overlaps
1      $distances$ = new empty set of distances
2      $overlaps$ = new empty map of overlaps
3      foreach $(t_1, t_2) \in top_1 \times top_2$ do
           // Ranked and Weighted
           // Penalty Distance
4          $distances$.add(rwpd($t_1$,$t_2$))
           // Document overlap in topics
5          $overlap$ = intersection of doc vectors of $t_1$ and $t_2$
6          $overlaps$.put(($t_1, t_2$), $overlap$)
       // Generate match-candidates
7      $candidates$ = new empty set of candidates
       // Document-overlap-based matching
8      $candidates$.add(docMatching($top_1$, $top_2$))
9      $candidates$.add(docMatching($top_2$, $top_1$))
       // Similarity-based matching
10     $candidates$.add(simMatching($distances$))
       // Evaluate potential matches
11     foreach $match \in candidates$ do
12         if $match$ is a similarity match then
13             if $match$ has document overlap then
14                 add $match$ to $results$ as complete match
               else
15                 add $match$ to $results$ as sim-only match
           else
16             if $match$ has document overlap then
17                 add $match$ to $results$ as doc-only match
18     return $results$
```

Algorithm 1: Layered Topic Matching. *LTMA* computes topic matches in three steps. First, the similarities and document overlaps are calculated based on the weighted keyword distance (see Section IV-A). Second, the potential matches are generated based on document overlaps (see Section IV-B) and on keyword similarities (see Section IV-C). Third, the match candidates are evaluated to the three different match-types.

Step 2: Generate potential topic match-candidates

The core step of the algorithm is the generation of the potential topic match-candidates. To do so, the document-overlap-based matching is calculated for each topic model separately, followed by the calculation of the similarity-based matches. Each match, identified by fulfilling one or both criteria, is added to the set of potential matches to be evaluated.

Step 3: Evaluate potential topic matches

Finally, the matches are generated and classified into three different types. First, the *complete matches*, where a topic pair is a match according to both methods; the document-overlap-based matching and similarity-based matching. Second, the *similarity-only matches*, where only the descriptor vectors of both topics match. Third, the *document-overlap-only matches*, a mismatch, where only documents but not the keywords match.

To reduce the complexity of the algorithm, *LTMA* processes the output of two topic modeling algorithms at a time. However, it can be analogously extended to compare more algorithms. Both the runtime and memory consumption of *LTMA* correlate with the number of topics processed.

A. Ranked and Weighted Penalty Distance (`rwpd`)

The computation of the weighted keyword distance between two topic descriptor vectors is designed to approximate manual matching. This *fuzzy* keyword similarity mirrors how humans perceive the similarity of two vectors, where a word at the beginning of the vector is perceived as important and influences the understanding of a topic more than a keyword at the end of the vector. Moreover, a keyword that occurs at the beginning of one vector but is not found in the other vector makes the two topics seem more different. Therefore, the introduced **weight** is proportional to the rank of the keyword in the vector, and missing keywords are punished with a **penalty** when computing the similarities. Hence, we measure the distance d of each common word in both topic vectors while assigning a penalty to words contained only in one vector. A weight w is assigned to each word proportional to its importance for the topic (measured by the rank k of the word in the vector of length l):

$$w(l,k) = \sqrt{l} - \sqrt{k} \qquad\qquad d(i,j) = \frac{|i-j|}{l}$$

The square-root-normalized weighting-function w declines in value with increasing positions in the vector. $d(i,j)$ is the normalized distance between two index positions i and j in the vector. These weights are used as factors to adjust the impact of the distances and penalties to the overall distance score. Let $V^{(t)}$ be a topic descriptor vector and $V_a^{(t)}$ be the keyword at position a in this vector, the penalty p is computed as follows:

$$p(V_i^{(t)}, V_j^{(u)}) = \begin{cases} d(i,j)(w(l,i)+w(l,j)) & \text{if } V_i^{(t)} = V_j^{(u)} \\ w(l,i) & \text{if } V_i(t) \notin V^{(u)} \\ w(l,j) & \text{if } V_j(u) \notin V^{(t)} \end{cases}$$

If a keyword from the one topic descriptor is not contained in the other descriptor, or vice-versa, the weight associated with the index position of the "wrong" keyword is taken as the penalty. If a keyword is contained in both vectors at different index positions, the weighted rank difference is the penalty. The *ranked and weighted penalty distance* between two vectors is then defined as follows:

$$\text{rwpd}(V^{(d)}, V^{(t)}) = \sum_{i=1}^{l}\sum_{j=1}^{l} p\left(V_i^{(d)}, V_j^{(t)}\right)$$

For all keyword pairs in both vectors, the weighted penalty values are calculated and summed up. As *LTMA* is a general algorithm, it enables the comparison of topic modeling results from both probabilistic and non-probabilistic models. Therefore, we do not include any keyword probabilities in our similarity metric and rely rather on word ranks instead.

The following example illustrates the distance computation for topic vectors that share the same amount of equal keywords but produce different distances.

Given three topic vectors: $t_1 =< a,b,c,d,e >$, $t_2 =< b,c,a,f,g >$, and $t_3 =< h,i,b,c,a >$;

$$\forall i,j \in \{1,2,3\} : t_i \cap t_j = \{a,b,c\}$$

Each pair of these topics shares exactly three words; namely, $\{a,b,c\}$. Intuitively t_1 and t_2 should be more similar to each other than any of them to t_3. This result is what *rwpd* would yield, as it introduces a higher weighted penalty when computing the distance of t_1 or t_2 to t_3, since t_3 contains two words in the first positions that none of the other two vectors share.

B. Document-Overlap-Based Matching (`docMatching`)

For each topic $t_1 \in top_1$ from one model, we determine the topic $t_2 \in top_2$ from the other model, with the highest normalized document overlap in this step. We find such a topic by counting the common documents of all topic pairs $\{t_1\} \times top_2$, and normalizing them by the number of documents containing t_1. The resulting value is called *document-match frequency* of top_1. The topic pair (t_1, t_2) with the highest document-overlap is linked as a matching candidate. The routine returns the set of n matching candidates for a model top_1 with n topics. Note, that it is called for all topics of both models, resulting in two document-match frequencies for a topic pair (t_1, t_2); one based on t_1, and another based on t_2. As described in Section III, the computation of document-overlap can be extended to accommodate a probabilistic topic distribution over the documents. This would lead to a partial overlap model, based on the topic-membership degrees of documents. Comparing the effects of such an approach to our current model is part of our future research agenda.

C. Similarity-Based Matching (`simMatching`)

For each topic pair and its measured distance, a normalized similarity value is computed, based on a linear normalization between 0 and 1. Eventually, if a similarity value exceeds a given threshold, its matching pair is considered a potential similarity-match. The threshold can be varied according to the structure of the underlying corpus and is by default set to 0.4. However, it can be interactively changed by the users. Hence, for corpora with very similar topics, this value can be lowered to capture all possible matches. As this routine inverts the normalized distances to compute similarities, it requires the distance values to be min-max-normalized between 0 and 1 to ensure a similarity value in this same range. This, however, implies that the calculated similarity values are relative for each run of *LTMA* and are, thus, not comparable across multiple runs.

V. Expert Case Studies

Since the presented algorithm is motivated by the need to capture the intuition of manual matching of topic modeling results, we conducted a set of expert case studies to systematically evaluate *LTMA*. This formative evaluation compares the matches generated by manual coders to those generated by the algorithm, confirming the accuracy of the generated topic matches. We relied on four different scholars from the humanities and social sciences, each performing two coding sessions. Since the manual matching of topics is a very time-consuming task, we compared the matches of four datasets, two larger corpora and two smaller ones. These were a syntactic corpus of book paragraphs, a corpus of news articles from the associated press, a transcript of a presidential debate, and a transcript of a movie. Each of the participants in our study chose one long and one short corpus. This study has covered comparisons between probabilistic (LDA [7], PAM [29], HDP [36]) and non-probabilistic (NMF [3], IHTM [18]) algorithms, as well as, comparisons within the same algorithm class. However, in this paper, we will focus on the results of matching results of probabilistic and non-probabilistic models, as these are typically more diverging than models from the same class. In the following, we will present a representative use case for the three application areas enabled by *LTMA*, based on the results of our study.

A. Automatic Comparative Evaluation of Topic Models

Validating and evaluating the results of topic modeling algorithms is a challenging task, especially considering that the perfect partitioning and labeling of topics is highly subjective. One of the main problems is the absence of benchmark datasets for comparative evaluation between models, making the application of automated, unsupervised evaluation methods challenging or unfeasible. While traditional unsupervised methods measure the cohesion of each topic and the separation between topics, they are highly algorithm-dependent, making them unsuitable for a comparative evaluation. Consequently, the most common evaluation method to date is a cross-fold validation; a subset of the corpus is held out for later evaluation [5]. This evaluation method verifies the *replicability* and stability of topic models. However, it does not validate the *interpretability* of their results.

LTMA can be used for an alternative evaluation approach, automatically measuring and comparing the interpretability of topic models. Using *LTMA*, differences and common factors of two models can be determined and evaluated. The amount of agreement and disagreement in the topic distribution is, thus, quantified, enabling conclusions based on relative differences between two models, as discussed in our previous work [17].

Beside being task-dependent, topic modeling evaluation can take into account different criteria, like robustness, stability, and replicability; interpretability and intuitiveness; or correctness and accuracy. The automatic evaluation using *LTMA* verifies the *correctness* of a new topic modeling algorithm without benchmark datasets. This is possible through the comparison of a new model to a well-established state-of-the-art approach. By comparing the similarities and, more importantly, the differences between both models, we achieve an objective indication of the accuracy of the new topic model.

Fig. 1 shows the interactive visualization interface, designed to display the results of *LTMA* for comparative analysis. Here, users can select a dataset and the topic modeling algorithms they want to compare. After setting the initial, model-dependent parameters, the results are computed and displayed as shown in Fig. 1. The similarity score of a matching topic-pair is encoded by the opacity of the corresponding line. Hovering over topics or topic matches, users get additional information on their matching keywords and similarity. Additionally, the overlapping documents and the relative amount of documents covered by the match (depicted by pie charts) are highlighted. In addition to exploring a given result, users can interactively adjust all parameters of *LTMA* through the visual interface, which supports a direct, bidirectional feedback loop to the algorithm, updating all views.

In our study we analyzed a corpus of all presidential debates between Trump and Clinton, 2016. Using the visualization interface, domain experts explored the topic distributions, the influence of individual parameters, and analyzed splits and merges between models. The experts were satisfied to see that the algorithm captured all manually found matches and discovered additional (weaker) ones that were captured by the model but that they did not detect manually. Through the accessibility of the results in the interactive visualization, the experts commented that they gained more understanding of, and awareness for the underlying models. Fig. 1 depicts the match between an IHTM [18] (with 17 topics) and an LDA model [7] (with 5 topics). Based on the extracted topics, *LTMA* detected altogether two complete-matches and eight similarity-only matches. The figure captures a hovered complete-match of a topic pair with two common keywords (*release* and *audit*). The pie charts highlight that most of the documents from the selected IHTM topic are assigned to the matching LDA topic.

B. Optimizing Model Results through Match Combinations

One of the most straight-forward use-cases for the topic matching result is its usage for optimizing topic modeling outputs. Following the idea of consensus clustering [30], the complete matches can be used as reliable topics, whereas topics without any matches can be considered as uncertain topics. By using topic matching as an optimization strategy, a more trustworthy, uncertainty-aware result can be generated. This result, in combination with the results of the input topic models, can be used for enhancing the display of topics in a tool or for user guidance. The complete matching result of *LTMA* can be regarded as a filtering criterion for relevant topics, and thus, be used to optimize the results of a topic modeling algorithm.

Table Ia shows the *LTMA* result applied to the movie transcript of *Cloud Atlas*. This movie consists of six parallel stories that unfold in different years. These stories get more and more intertwined over the course of the movie, since the characters write letters in the past that are found by others in the future. This complex movie plot is visually represented in Fig. 3. Here, the entanglement of the storylines is clearly visible. The movie starts with each storyline separate and, over time, the stories get more mixed. Through matching the results of two topic models, a substantial accuracy improvement can be achieved (see Fig. 5). We used a probabilistic (LDA) and a non-probabilistic (NMF) model for this task.

Regarding each topic modeling result separately, as shown in Tables Ib and Ic, some very general keywords can be found

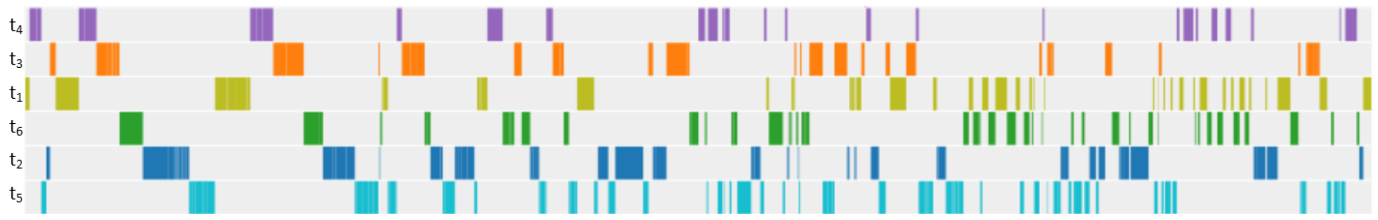


Fig. 3: Visual representation of the story timelines of the movie *Cloud Atlas*[1]. This visualization depicts each movie scene as a colored bar, with its size corresponding to the length of the scene description and its color showing which topic it was assigned to. Each row corresponds to one topic from the matched results of Table Ia, highlighting the intertwined movie narrative.

ID	T1-T2	DS	Keyword Intersection	DMF T1	DMF T2	Matches
t_1	4 - 7	1.000	kona, zachry, catkin, ain, jus, dwell, georgie, horse, yay, bailey, meronym, prescient, abbess	1.000	0.826	38
t_2	1 - 2	0.878	cavendish, mr, ruddy, aurora, house, hotchkiss, noake, ernie, veronica, timothy	1.000	0.618	34
t_3	3 - 6	0.799	sixsmith, vyvyan, chateau, robert, music, vo, hotel, frobisher, ayr	0.964	0.643	27
t_4	2 - 4	0.774	autua, adam, prophetess, goose, ew, molyneux, missa, ewing, horrox	1.000	0.793	23
t_5	6 - 1	0.744	sonmi, transport, yoona, enforcer, archivist, fabricant, chang, papa, interrogation	1.000	0.774	65
t_6	5 - 3	0.711	sixsmith, isaac, luisa, rey, javier, napier, apartment	0.867	0.542	26

(a) Matches between LDA and NMF for the movie *Cloud Atlas*. Descriptor Similarity (DS) and Document-Match-Frequency (DMF) show the match quality.

#	**T1** – Topic Descriptors
1	cavendish, mr, ernie, aurora, house, veronica, meek, dermot, denholme, hotchkiss, noake, ruddy, wither, judd, timothy
2	ew, goose, autua, mr, captain, adam, prophetess, boerhaave, horrox, sir, ewing, moore, molyneux, good, missa
3	frobisher, ayr, music, sixsmith, vo, jocasta, hotel, cont, robert, stop, time, play, vyvyan, chateau, day
4	zachry, meronym, georgie, catkin, kona, ain, abbess, horse, bailey, jus, dwell, sonmi, prescient, yay, rose
5	luisa, rey, napier, javier, sixsmith, man, isaac, letter, megan, smoke, apartment, li, fay, elevator, read
6	sonmi, chang, enforcer, archivist, year, yoona, room, day, song, papa, fabricant, vo, transport, interrogation, boom
7	year, cont, day, night, vo, door, open, room, smile, head, back, hand, begin, close, eye

(b) LDA result for the movie *Cloud Atlas*

#	**T2** – Topic Descriptors
1	2144, sonmi, 451, chang, enforcer, archivist, papa, fabricant, interrogation, pureblood, yoona, transport, boomsook, sook, kim
2	cavendish, 2009, aurora, house, ernie, veronica, mr, nurse, hotchkiss, ursula, office, ruddy, timothy, noake, knuckle
3	luisa, 1973, rey, apartment, napier, swannekke, couch, isaac, yeah, father, sixsmith, lester, power, javier, smash
4	ew, 1846, prophetess, goose, ewing, autua, henry, adam, reverend, missa, horrox, molyneux, fatherinlaw, earn, doctor
5	hopeless, oblivious, unspeakable, history, ago, form, suppose, round, warn, train, corner, tear, ly, adam, window
6	1931, frobisher, ayr, music, chateau, vyvyan, sixsmith, robert, bruge, vo, zedelghem, hotel, memle, sextet, short
7	zachry, 2321, meronym, bailey, catkin, dwell, kona, jus, yay, georgie, ain, horse, prescient, abbess, bout

(c) NMF result for the movie *Cloud Atlas*

TABLE I: *LTMA* results of two topic modeling algorithms (LDA and NMF) applied to the movie transcript of *Cloud Atlas*. This movie has six parallel storylines which were only identified clearly when matching the results of the two topic modeling algorithms. This topic matching shows a relatively high descriptor similarity for all topics and a very high document (scene) match frequency.

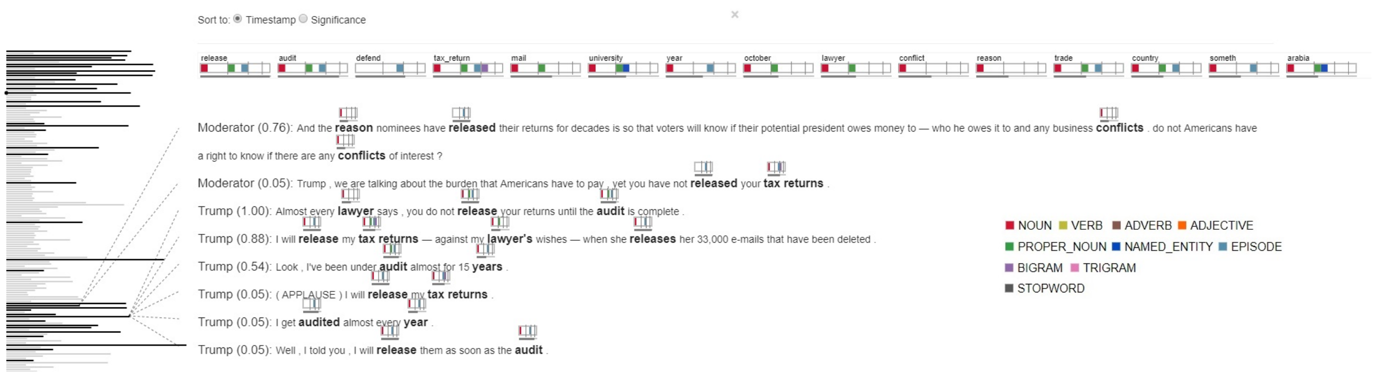
among the topic descriptors. Although the movie consists of six parallel storylines, the topic modeling results of both models did not segment the transcript into six topics correctly with the *number of topics* parameter set to six. Instead, it had to be adjusted to seven in order to account for noise in the data. The results show that LDA creates, aside from the six storylines, a seventh topic with keywords related to time and date, which is probably due to the constant jumps between the timelines. NMF however, creates an additional topic (t_5) that obviously groups together the scenes with the letters, since these are shown across the storylines and are usually written in one timeline, but read in another one. This example illustrates how topic matching can enhance the quality of topic modeling results and optimize the topic generation process. All scenes of this movie have been manually annotated within the study, and the manual matches found between the topic modeling results were all derived automatically by *LTMA*. Here, the experts emphasized that they usually combine the results of different algorithms in their daily work to get a more reliable base for analysis.

C. Visual Exploration of Topic Modeling Differences

Topic models are widely used to create visualization tools that explore the content of large text corpora in combination with other meta-data. Using a topic matching data structure, a new level of exploration can be enabled, since the results of multiple topic models are put in relation to each other. This opens a wide variety of tasks, for example comparison and optimization of results, visual evaluation and validation of topic models, and an enhanced interactive exploration of topics through topic ranking and recommendation.

One useful scenario for understanding topic modeling results is topic summary analysis. Here, the user can explore how matching topics have been generated, and what the reasons for their (dis)similarities are. Various systems and visualizations exist to explore topic modeling differences; the example we implemented for our study is shown in Fig. 4. It is designed to enable the exploration of topic model differences based on their

[1]http://www.imdb.com/title/tt1371111/ accessed on the 2nd of June 2018



Moderator (0.76): And the **reason** nominees have **released** their returns for decades is so that voters will know if their potential president owes money to — who he owes it to and any business **conflicts** . do not Americans have a right to know if there are any **conflicts** of interest ?

Moderator (0.05): Trump , we are talking about the burden that Americans have to pay , yet you have not **released** your **tax returns** .

Trump (1.00): Almost every **lawyer** says , you do not **release** your returns until the **audit** is complete .

Trump (0.88): I will **release** my **tax returns** — against my **lawyer's** wishes — when she **releases** her 33,000 e-mails that have been deleted .

Trump (0.54): Look , I've been under **audit** almost for 15 **years** .

Trump (0.05): (APPLAUSE) I will **release** my **tax returns** .

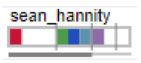
Trump (0.05): I get **audited** almost every **year** .

Trump (0.05): Well , I told you , I will **release** them as soon as the **audit** .

Fig. 4: The topic summary visualization shows the most representative sentences with highlighted topic keywords for a selected topic. In this example, a summary of the *tax-release* topic from a *Trump-Clinton debate*, based on an LDA model, is displayed.

input features. Before topic modeling algorithms are executed, users can specify a set of features to be used for complying the document descriptor vectors. Such features could include word classes, as depicted in Fig. 4. These features are specified for each topic model separately, thus, the influence of different feature settings on the extracted topics can be analyzed.

Fig. 4 shows a summary of the *tax-release* topic, as discussed in the first presidential debate between Donald Trump and Hillary Clinton in 2016. To extract summary sentences, we apply a summarization algorithm from our previous work [17]. In the visualization, a topic is depicted as a list of *keyword-glyphs*, while a single glyph highlights the particular features a keyword satisfies. The significance (probability) score of each keyword is displayed as a scaled line underneath the glyph. The score is normalized according to all keywords of all topics, enabling their comparison.



This summary gives an overview of the context in which keywords have been used in the observed documents. As the summarization can be applied to both the modeling results and the LTMA match result, it enables users to see how changing input paramters influence the topic modeling, enabling further optimization. The topic matching step is necessary, as seemingly small changes to the model parameters can have large, unexpected effects on the output [27]. With the underlying, automated topic matching, users are freed from having to search similarities and changes between the models on the topic-level, and can instead focus on the semantic implications of the changes.

VI. Discussion and Validation

Table II shows the results of *LTMA* applied on a corpus of news articles from the associated press. This corpus was processed by two topic modeling algorithms, the probabilistic LDA [7] model (T1) and the non-probabilistic IHTM [18] (T2). Each model produced 100 topics. The main topics discussed in these news articles are clearly visible in the matching descriptor keywords. If we consider the scenario of evaluating the newer IHTM without any gold-standard data, a comparison to the output of the well-established LDA can give an indication of the quality of IHTM. As many easily understandable topic keyword intersections appear in the table as complete matches,

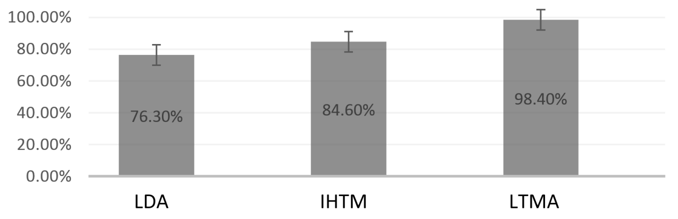


Fig. 5: Accuracy of the topic modeling results in comparison. These values correspond to the relative amount of documents assigned to the correct storyline. The bar-chart displays the accuracy of the topic modeling results of LDA (Table Ib) with 76.3%, IHTM (Table Ic) with 84.6%, and the accuracy of their topic matching *LTMA* result (Table Ia) with 98.4%.

we can conclude that the two models do indeed produce similar outputs, giving the same labels to topics that are attributed to similar documents. However, the table also reveals differences, listed under "similarity-only matches". Here, topics with the same keywords have been attributed to different topics, highlighting dissimilarities between the models. The domain experts participating in our studies have found such a table helpful, and have made several interesting discoveries.

The first interesting finding is, that several topic pairs have a comparatively high document overlap and a relatively high keyword similarity, e.g. $\{12-174\}$. These pairs consist of topics that are well-separated from the rest of the corpus and that are identified by both models. On the other hand, several topic pairs containing the same topic from one model, paired with different topics from the other model, exist. Some of these pairs are, for example, $\{8-46, 8-264\}$, $\{16-7, 16-69\}$, $\{14-219, 6-219\}$. They indicate that, for one concept, one topic model is grouping the documents into one topic, while the other model is separating them into (at least) two different topics. This effect does not necessarily mean that the model grouping topics together produces better or worse results. While it would make sense to combine the topics from the previous examples, documents also get grouped together because of shared keywords, while dealing with different concepts. Such instances can, for example, be found in the match-pairs $\{9-130, 9-4\}$ and $\{4-71, 4-142, 4-146\}$. *LTMA* found almost

ID1	ID2	DS	Keyword Intersection	DMF T1	DMF T2	Matches
Complete Matches						
7	68	1.000	bush, republican, dukakis, democratic, campaign, presidential, governor, president	0.422	0.593	54
13	90	0.897	corp, chairman, executive, million, billion, company, share	0.466	0.540	54
9	130	0.871	german, germany, east, west, berlin	0.400	0.700	14
20	276	0.821	communist, leader, election, republic, political, gorbachev, reform, party	0.296	0.872	34
15	180	0.816	kuwait, iraq, iraqi, saudi, arabia, gulf	0.278	0.465	20
8	46	0.811	committee, bill, house, senate, sen, vote	0.188	0.760	19
12	174	0.802	yen, market, trade, price, exchange, stock	0.842	0.970	96
18	95	0.663	increase, percent, rate, price, report	0.493	0.628	71
14	219	0.480	kill, wound, police	0.350	0.405	49
16	7	0.558	plane, flight, pilot, aircraft	≤ 0.1	1.000	6
8	264	0.510	committee, republican, house, rep	≤ 0.1	0.471	8
4	146	0.541	parent, student, school	≤ 0.1	0.440	11
16	69	0.438	plane, air, accident, force	≤ 0.1	0.889	8
17	170	0.426	star, movie, film	≤ 0.1	0.733	11
20	83	0.423	election, opposition, vote, party	≤ 0.1	0.727	8
4	71	0.498	medical, doctor, patient, hospital, care	≤ 0.1	0.600	12
14	219	0.480	kill, wound, police	0.350	0.405	49
4	142	0.547	student, university, school	≤ 0.1	0.360	9
Similarity-Only Matches						
9	4	0.556	german, britain, european, france, europe	≤ 0.1	≤ 0.1	1
6	219	0.479	murder, arrest, kill, police	≤ 0.1	≤ 0.1	41
20	232	0.463	union, soviet, gorbachev	≤ 0.1	≤ 0.1	17

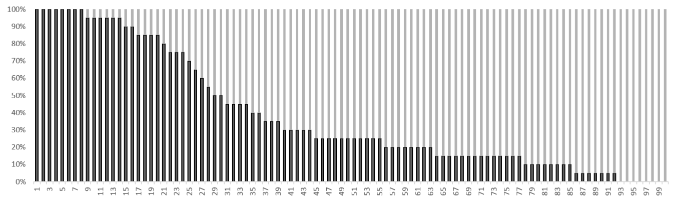
TABLE II: (Partial) Topic matching results of a corpus containing 2246 documents from the Associated Press[2]. For each pair of topics, the table displays the (normalized) descriptor similarity (DS), keyword intersections, topic match frequencies (DMF) for both models, and the count of overlapping documents.

all topic pairs that were matched manually by domain experts. In some cases, it was also able to identify additional matches that had not initially been discovered by the analysts. They were, however, found to be correct after a thorough analysis.

Another key aspect for assessing the quality of *LTMA* results was the ranking of the matched topic pairs, which reflected the same tendency of the manual ranking. Out of 163 manually (by at least 2 experts) annotated matches, the algorithm found 158, resulting in an overall recall of 0.96. Furthermore, out of the 214 matches found by the algorithm, 187 were marked as relevant by at least one expert, resulting in an overall precision of 0.87. Since the matches are scored based on their descriptor similarity, the experts found the lower precision value not to be an obstacle for the analysis.

The quality of the topic matching results is reflected in the keyword intersections extracted from the topic pairs. Fig. 6 shows the relative amount of non-descriptive keywords per topic for both topic models and the Associated Press corpus. Averaged over both models, 33.5% of the keywords were found to be non-descriptive for their respective topic in a manual annotation session. Especially LDA produced some topics that were found to be too general or too noisy. In comparison to that, results of the topic matching presented in Table II contain almost no irrelevant words. This shows how the results of both topic models were enhanced by *LTMA*.

[2]http://www.cs.columbia.edu/~blei/lda-c/ accessed on the 2nd of June 2018


(a) Percentage of non-descriptive keywords in each LDA topic

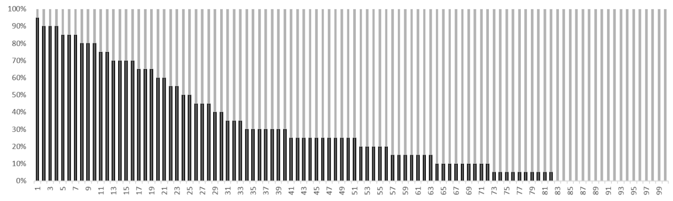

(b) Percentage of non-descriptive keywords in each IHTM topic

Fig. 6: Relative rate of non-descriptive keywords (black bar) in each topic. While the keywords of topic matches shown in Table II are mostly descriptive, participants in our study found 38% and 29% of all LDA (6a) and IHTM (6b) topic descriptors to be too general, respectively.

For the Cloud Atlas corpus, we calculated the precision of the topic modeling results in comparison with the *LTMA* results. Fig. 5 shows the amount of documents that were correctly assigned to their topics. While LDA and IHTM were able to correctly assign 76.3% and 84.6% of the documents correctly, LTMA was able to remove noise and assign 98.4% of the documents correctly. This example also highlights that *LTMA* can improve the classification precision of topics and can enhances the results of both algorithms by combining them.

VII. Conclusion

In this paper, we have introduced a layered topic matching algorithm that operates on the results of topic models to generate a topic match data structure, containing three types of matches: complete matches, similarity-only matches, and document-overlap-only matches. We have exemplified the usage of the *LTMA* results based on expert case studies, in addition to a discussion of visual exploration possibilities and other potential application areas. Motivated by the need for an automatized method that captures matching topics, mimicking the human intuition of similarity, our algorithm enables the objective comparison of topic modeling results based on the quantitatively defined *ranked and weighted penalty distance*. Moreover, the resulting data structure enables pairing individual topics and providing metrics about their match. Such a data structure has been successfully utilized in our previous work [17] to compile topics of interest. In addition, this paper confirmed the effectiveness of the proposed method based on a long-term observational study, followed by several expert studies using tailored visualizations for the different application areas. In our future work, we will extend the algorithm to enable matching the results of more than two topic models at a time. Additionally, we plan to extend it to incorporate probabilistic topic models, enabling probabilistic topic match data structures.

References

[1] Charu C. Aggarwal and ChengXiang Zhai. *Mining text data*. Springer Science & Business Media, 2012.

[2] Eric Alexander and Michael Gleicher. Task-driven comparison of topic models. *IEEE Trans. on Visualization and Computer Graphics*, 22(1):320–329, 2016.

[3] Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models–going beyond svd. In *IEEE Symp. on Foundations of Computer Science*, pages 1–10, 2012.

[4] Lasse Bergroth, Harri Hakonen, and Timo Raita. A survey of longest common subsequence algorithms. In *Proc. IEEE Symp. on String Processing and Information Retrieval*, pages 39–48, 2000.

[5] David M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, April 2012.

[6] David M. Blei and John D. Lafferty. Dynamic topic models. In *Proc. 23rd Int. Conf. on Machine Learning*, pages 113–120, 2006.

[7] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. of Machine Learning Research*, 3:993–1022, 2003.

[8] Charles Blundell, Yee Whye Teh, and Katherine A. Heller. Bayesian rose trees. *Proc. of the 26th Conf. on Uncertainty in Artificial Intelligence*, pages 65–72, 2012.

[9] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-Graber, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *Proc. Advances in Neural Information Processing Systems*, pages 288–296, 2009.

[10] Jaegul Choo, Changhyun Lee, Chandan K. Reddy, and Haesun Park. Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Trans. on Visualization and Computer Graphics*, 19(12):1992–2001, December 2013.

[11] Jason Chuang, Sonal Gupta, Christopher Manning, and Jeffrey Heer. Topic model diagnostics: Assessing domain relevance via topical alignment. In *Proc. of the 30th Int. Conf. on Machine Learning*, volume 28, pages 612–620, 2013.

[12] Jason Chuang, Christopher D Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Proc. of Int. Conf. on Advanced Visual Interfaces*, pages 74–77. ACM, 2012.

[13] Patricia J. Crossno, Andrew T. Wilson, Timothy M. Shead, and Daniel M. Dunlavy. Topicview: Visually comparing topic models of text collections. In *Proc. IEEE Int. Conf. on IETools with Artificial Intelligence (ICTAI)*, volume 23, pages 936–943, 2011.

[14] Wenwen Dou, Xiaoyu Wang, Remco Chang, and William Ribarsky. ParallelTopics: A probabilistic approach to exploring document collections. In *Proc. IEEE Conf. on Visual Analytics Science and Technology*, pages 231–240, 2011.

[15] Wenwen Dou, Li Yu, Xiaoyu Wang, Zhiqiang Ma, and William Ribarsky. Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. *IEEE Trans. on Visualization and Computer Graphics*, 19(12):2002–2011, 2013.

[16] Jacob Eisenstein, Duen Horng Chau, Aniket Kittur, and Eric Xing. TopicViz: Interactive topic exploration in document collections. In *Extended Abstracts of SIGCHI Conf. on Human Factors in Computing Systems*, pages 2177–2182, 2012.

[17] Mennatallah El-Assady, Rita Sevastjanova, Fabian Sperrle, Daniel Keim, and Christopher Collins. Progressive Learning of Topic Modeling Parameters: A Visual Analytics Framework. *IEEE Trans. on Visualization and Computer Graphics*, 24(1):382–391, 2018.

[18] Mennatallah El-Assady, Fabian Sperrle, Oliver Deussen, Daniel Keim, and Christopher Collins. Visual Analytics for Topic Model Optimization based on User-Steerable Speculative Execution. *to appear, IEEE Trans. on Visualization and Computer Graphics*, 2018.

[19] Ashwinkumar Ganesan, Shimei Pan, and Jian Chen. LDAExplore : Visualizing Topic Models Generated Using Latent Dirichlet Allocation. *Proc. IUI Workshop on Visual Text Analytics*, 2015.

[20] Viswanath Gangavaram and Darshan Hedge. Search term clustering, 2014. US Patent App. 14/300,093.

[21] Richard W Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.

[22] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*, volume 3. Morgan-Kaufmann (Elsevier), 2011.

[23] Thomas Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196, 2001.

[24] Maurice George Kendall. Rank correlation methods. 1948.

[25] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 03 1951.

[26] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[27] Tak Yeon Lee, Alison Smith, Kevin Seppi, Niklas Elmqvist, Jordan Boyd-Graber, and Leah Findlater. The human touch: How non-expert users perceive, interpret, and fix topic models. *Int. J. of Human-Computer Studies*, 105:28–42, 2017.

[28] Shixia Liu, Xiting Wang, Yangqiu Song, and Baining Guo. Evolutionary bayesian rose trees. *IEEE Trans. on Knowledge and Data Engineering*, 27(6):1533–1546, 2015.

[29] David Mimno, Wei Li, and Andrew McCallum. Mixtures of hierarchical topics with pachinko allocation. In *Proc. of the Int. Conf. on Machine Learning*, volume 24, pages 633–640. ACM, 2007.

[30] Stefano Monti, Pablo Tamayo, Jill Mesirov, and Todd Golub. Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning*, 52(1-2):91–118, 2003.

[31] Eric Sven Ristad and Peter N. Yianilos. Learning string-edit distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(5):522–532, 1998.

[32] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proc. of Conf. on Uncertainty in Artificial Intelligence*, pages 487–494. AUAI Press, 2004.

[33] Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

[34] Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring topic coherence over many models and many topics. In *Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 952–961. ACL, 2012.

[35] Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar. *Introduction to data mining*, volume 2. Pearson Addison Wesley Boston, 2006.

[36] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M Blei. Hierarchical Dirichlet Processes. *J. of the American Statistical Association*, 101(476):1566–1581, 2006.

[37] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proc. of the A. Int. Conf. on Machine Learning*, volume 26, pages 1105–1112. ACM, 2009.