Eurographics Conference on Visualization (EuroVis) 2024
W. Aigner, D. Archambault, and R. Bujack
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 3

HyperMap: Sensemaking of Text Corpora using Hypergraphs

submission ID

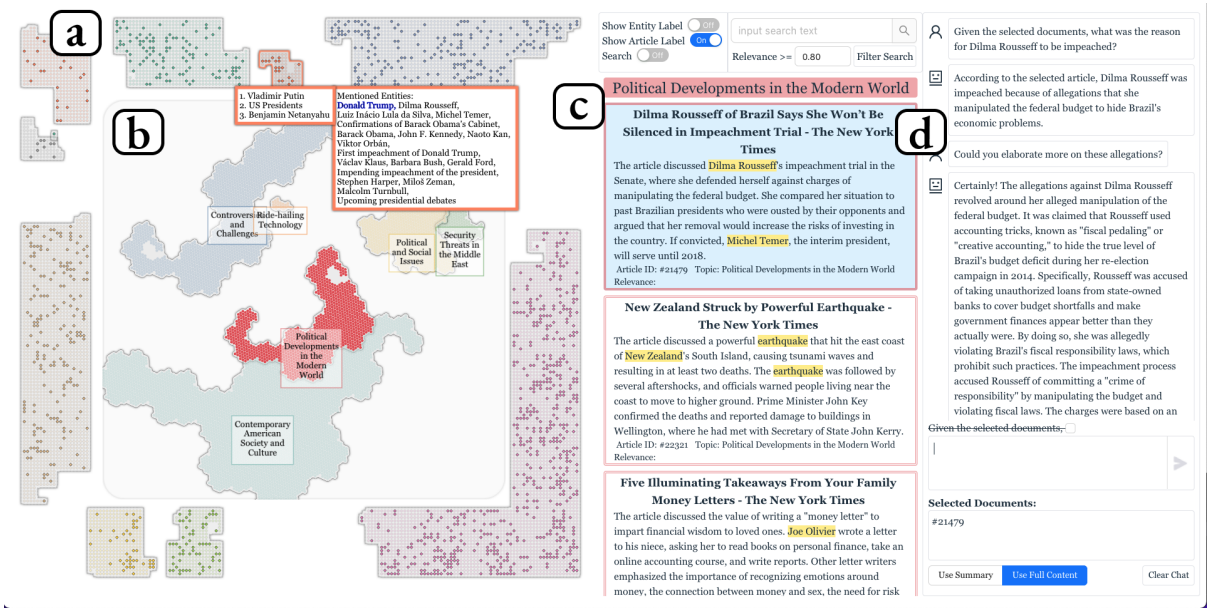


Figure 1: The HyperMap system. (a) The peripheral area of Cluster View shows the mentioned keywords of highlighted documents using Gilbert curves. (b) The center area of Cluster View shows the topic structure of the corpus using Gosper curves. (c) The Document View shows a list of selected documents. (d) The Chatbot View provides a chatbot interface to answer user questions with the option to insert selected documents in the prompt.

Abstract

The ABSTRACT is to be in fully-justified italicized text, between two horizontal lines, in one-column format, below the author and affiliation information. Use the word "Abstract" as the title, in 9-point Times, boldface type, left-aligned to the text, initially capitalized. The abstract is to be in 9-point, single-spaced type. The abstract may be up to 3 inches (7.62 cm) long.
Leave one blank line after the abstract, then add the subject categories according to the ACM Classification Index (see https://www.acm.org/publications/class-2012)

CCS Concepts

• ***Computing methodologies*** → *Collision detection;* • ***Hardware*** → *Sensors and actuators; PCB design and layout;*

1. Introduction

Text data is ubiquitous. From news articles and social media posts to scientific publications, the tremendous amount of text data that is produced poses not only opportunities but also a great challenge to anyone who needs to analyze them. Visual analytics (VA) mitigates this challenge by combining mathematical models and visualizations to automate the sensemaking process and reduce the cognitive load. Chuang et al. [CRMH12] proposed that *Model alignment*, the alignment of analysis tasks, visual encodings and model decisions, greatly affects users' interpretation and trust in visual analytic systems. However, in text analysis, the available models often align poorly with analysis tasks. For example, topic models are commonly used to model the topical structure of text documents. Most topic models characterize *topic* as a probabilistic distribution spanning a given vocabulary [VK20]. This transformation from a *topic*, a

high-level concept that the user seeks to understand, to a *probabilistic distribution*, a low-level concept that mathematical models can operate on, prevents proper model alignment. The misalignment between analysis tasks and models limits the usage of visual analytics systems for users who are not familiar with the underlying models. Previous works that support users to analyze large collections of documents adopt separate models for analyzing topics and entities, but they are often intertwined in the user's mental model. Model alignment requires the model to operate on documents and entities simultaneously, a capability that existing text analysis models lack.

Recent advances in large language models (LLM) present a promising solution to this problem. LLMs have proven successful in various natural language processing (NLP) tasks, especially in question-answering tasks due to their strong capability to understand user intent. Researchers in visualization have adopted LLMs to assist data transformation [WTL23] or directly generate visualization [MS23]. However, they all assumed a clean data format, where the data to be visualized is already in a table format. For unstructured text analysis though, this is rarely the case. Topics [ACS*23], sentiments [BFR21], concepts and entities [PKL*18, CSL*10] are common analysis targets in text analysis that require a data preparation stage to extract them from unstructured text. The capability of LLMs to extract information from documents according to user intent eliminates the need to carefully align the analysis tasks and models in VA systems, because a specific model is no longer needed to prepare the data for the analysis task. In the previous example, instead of relying on abstruse and unfathomable probabilistic models, LLMs can directly process the text data and summarize the topics of the documents. A user can ask a LLM: *'What are the topics of these articles?'*, and the LLM would give a human-like response, such as *'The articles are about Covid-19'*.

In this work, we designed a VA system that models a corpus as a hypergraph, where the nodes are documents and salient keywords extracted from the document using LLMs. We showcase how LLMs are used flexibly to align the data, analysis task and visualization during our design process. The hypergraph is then hierarchically clustered and visualized with enhanced space-filling curve layouts [MM08]. The system supports interactive exploration, reorganization and analysis of the documents. To the best of our knowledge, no published visual analytics system has adopted LLMs to align the data, analysis task and visualization. Using the system, we demonstrate how proper model alignment can be achieved using LLMs.

The contributions of our work are as follows:

- We introduce an LLM-based information extraction pipeline that is capable of extracting topics and salient keywords from a given corpus in a way that fosters interpretation.
- We extend space-filling curve layouts to visualize clusters in large hypergraphs.
- We develop a novel VA system that allows users to effectively make sense of a corpus.

2. Related Works

2.1. Data preparation for large collections of text

TODO: shorten

Topic-based approaches Topic-based approaches employ certain variations of topic models to organize the documents in a meaningful way. Each topic is often presented as a 'bag of words', which can be in the form of a sequence of words [YYQ17, CDW*16, DYW*13, YTJ*19, HPK*22, AKV*14, LKC*12] or word clouds [OSR*14, CDW*16]. The modeling result provides an overview of the dataset for subsequent analysis tasks. Despite their popularity, the use of topic models as an overview, as well as its 'bag-of-word' visualization, is reported by Lee et al. [LSS*17] to be problematic, especially for non-expert users, in a comprehensive user study. Chuang et al. [CRMH12] concluded that these problems arise from a misalignment between the analysis task, visual encoding and model. The sensemaking process becomes challenging without a basic understanding of the model because the 'bag-of-words' representation is too far away from the user's mental model of a topic. This misalignment limits the usage of topic models for non-expert users and makes the system prone to produce false positives.

Entity-based approaches A line of work that makes successful model alignments is the entity-based approach. 'Entities' usually include named entities (people, organizations, locations), or meaningful concepts known to an existing knowledge base. The earliest of such approaches is Jigsaw [SGLS07], where entities are linked if they appear in the same document. FacetAtlas [CSL*10] generalizes the idea of entity to 'facets' which can be entities or any keywords or user's interest. ConceptVector [PKL*18] uses 'concept' to represent a similar idea. Generally, entity-based approaches exhibit better model alignments than topic-based approaches [CRMH12], but the polysemy of natural language makes them prone to produce false positives [PKL*18]. In our work, we use *keywords* to represent entities or concepts that appear in the documents. We ensure that the keywords are salient by exploiting LLM's ability to understand semantic contexts.

Embedding-based approaches Finally, an important line of work organizes documents by directly modeling their semantic similarity [SKK00]. Documents are first projected into a high-dimensional vector space where similarity can be measured, and then a dimensionality reduction technique (e.g.t-SNE) is used to project the dataset onto a two-dimensional space for visualization. Earlier works construct a sparse vector using term-frequency based scores such as *TF-IDF* or BM25 [CLRP13, SNMM18]. More recently, the success of pre-trained language models like BERT [DCLT18] popularizes the idea of embedding documents in a dense vector space [NKWW22, TLW*23, QTW*22]. The embedding can then be used for document retrieval [KOM*20, ICH*22] or visualization. Embedding-based approaches also exhibit healthy model alignment, as the vector space directly models the analysis task (finding similar documents). However, the result often lacks explainability and prevents users from trusting the result. Recently, Raval et al. [RWVW23] proposed to use LLMs to provide explainability to embeddings-mappings visualizations. We adopt a similar approach in our system to provide interpretability to the clustering result.

2.2. Visual Interfaces for Document Corpus

2.3. LLMs for Information Extraction

Information Extraction aims to identify structured information of interest from unstructured text data. Some of its subtasks include

Named Entity Recognition (NER), Relation Extraction (RE) and Event Extraction (EE) [NJM21, XW19]. Although LLMs have proven successful in many NLP tasks, their application to IE is non-trivial. First, the *faithfulness* of LLMs needs to be carefully evaluated. Faithfulness refers to the ability of a model to adhere to the provided information and not use parametric knowledge learned during training to answer user questions [ZZPC23]. When conducting information extraction, it is necessary to ensure that the extracted information is actually from the provided text and not from the model's parametric knowledge. Second, LLMs are known to produce *hallucination*, where LLMs provide answers factually contradicting to input text (intrinsic) or even factually false (extrinsic). In the context of IE, we mainly focus on the intrinsic hallucination problem. A recent evaluation conducted by Bang et al. [BCL*23] found that ChatGPT rarely exhibits intrinsic hallucinations, including the abstractive summarization task from which neural models usually suffer.

More specifically, Li et al. [LFY*23] comprehensively evaluated the capabilities of ChatGPT for common IE tasks. They found that ChatGPT excels under the Open-IE setting, where the model relies solely on user input to extract information from documents, but performs poorly under the Standard-IE setting, where ChatGPT is instructed to choose a correct label. Their findings agree with Zhang et al. [ZLZ23] where ChatGPT is reported to perform poorly on extractive summarization. A common reason for the poor performance of ChatGPT in these tasks is that they are essentially supervised learning tasks, and ChatGPT is not trained to perform them. To make the best use of ChatGPT (or more generally, LLMs) for IE tasks, we need to carefully design the extraction tasks as question-answering tasks instead of supervised learning tasks.

3. Design Rationale

HyperMap is designed for analysts to explore and reorganize a corpus for their analysis. Our design rationale to foster interpretation is based on the design guidelines proposed by Chuang et al. [CRMH12]. We reuse their definitions of *Model Alignment*, *Progressive Disclosure*, and *Unit of Analysis* when describing our design rationale. We first identify common analysis tasks from previous works. Then, we derive our design considerations (DC) from the analysis tasks. We take the DCs into account during the model and visualization design process in section 4.

3.1. Analysis Tasks

We derive our target analysis tasks from topic- and entity-based approaches. Topic-based approaches aim to support document understanding by visualizing the topic structure of the documents. Entity-based approaches support investigative analysis by visualizing entities and their relations. We first generalize entities to *Keywords*, which are the core entities or concepts discussed in the documents. For example, in a news article, the keywords can be named entities that appear in the title or are involved in the news event. In a research article, the keywords can be the concepts or models proposed by the described work. Similarly, the investigation of the keywords seeks to answer the question: *What keywords are discussed in the corpus, and how are they related?*: We aim to support both tasks simultaneously as they are fundamental to subsequent tasks and intertwined in a real-world scenario.

3.2. Design Considerations

To support the aforementioned analysis tasks, we derive the following design considerations (DCs):

- **DC1: Overview of topic structures and keyword connections.** Given a corpus, the topic structures and keyword connections can be complex and cover a wide range of documents and keywords. The overview seeks to cover all the documents and keywords while hiding the details. This sets the ground for the user to discover their targets of interest.
- **DC2: Progressive Disclosure.** To facilitate investigation, it is important to support users to drill down from a high-level overview to intermediate abstractions. This includes disclosure of a specific topic's sub-structure, the containing documents, and the connections to keywords. Users also reorganize a corpus during this process.
- **DC3: Model Alignment.** Our choice of models should align well with the analyst's mental model when conducting the analysis tasks for better interpretability. This means our model should directly operate on the units of analysis, which are documents and keywords.

4. Methodology

As shown in Figure 2, starting from a corpus of unstructured texts, we first use LLMs to extract the main keywords from each document. The keywords are disambiguated and linked to a knowledge base if available. We create and store document embeddings and keyword embeddings for further usage. All LLM-based preprocessing used OpenAI's 'gpt-3.5-turbo-16k-0613' model. We provide all our prompts in supplemental material. In the next stage, we construct a document hypergraph and a keyword hypergraph, which are then clustered separately by combining connectivity similarity and semantic similarity. The clustering result is hosted on the server and visualized in an interactive user interface. Below, we describe each component in detail.

4.1. Data Preparation

The data preparation stage consists of five steps: summarization, keyword extraction, keyword disambiguation, document and keyword embedding, and topic label generation. We use LLMs to perform all five steps as they outperform traditional approaches in both accuracy and flexibility. Note that since the user does not directly interact with the LLM in our system, we do not consider our system as an agent-based system, as will be discussed in section 8.

4.1.1. Summarization

Unstructured text documents can contain a vast amount of information, among which certain information is more important than others. Summarization helps condense the content into a concise form, making it easier for users to make sense of and LLMs to process and extract the most important information. Zhang et al [ZLZ23] reported that human users found the summaries generated by ChatGPT to

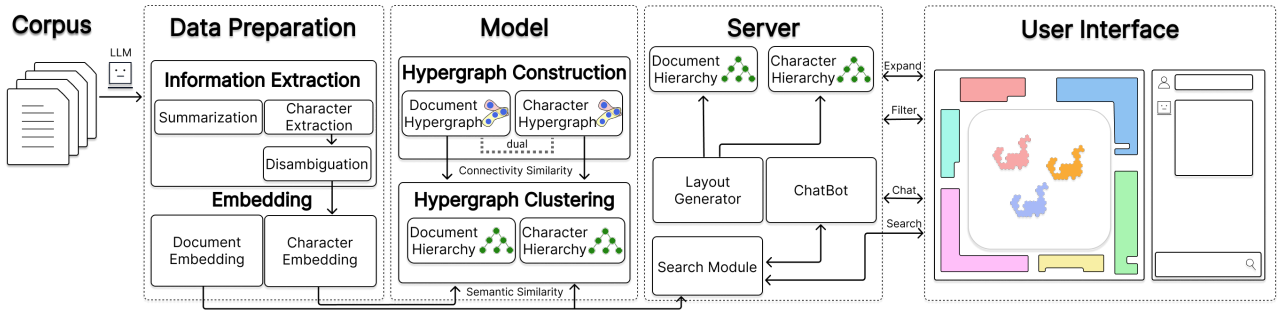


Figure 2: Data processing pipeline of HyperMap. Starting from a corpus of unstructured texts, each document goes through the data preparation stage to extract the salient keywords. Then the documents and keywords are both embedded into a vector space. The model stage constructs a document hypergraph and a keyword hypergraph, which are then clustered separately by combining connectivity similarity and semantic similarity. The clustered hypergraphs are hosted on the server and visualized in the user interface. Users can expand, filter, or search the hypergraphs to explore the corpus and select documents to be analyzed with a chatbot.

be more interpretable and trustworthy than traditional approaches, which makes LLMs a better choice for our purpose. We employed an instruction-based zero-shot prompt, where the model is instructed to act as a text summarizer given the document's content. These summaries are used in subsequent preprocessing stages as well as in the user interface. The summarization module can be replaced with any other summarization model or more advanced LLM summarization techniques if desired, but in early prototyping, we found that a zero-shot prompt is good enough for most datasets.

4.1.2. Keyword Extraction

The keyword extraction builds upon the summaries generated in the previous step. We define a keyword as entities or concepts that are significantly discussed in the documents. For news articles, a keyword can be a named entity such as a person, place or location. For research papers, a keyword can be a model, a technique or an algorithm proposed or used by the paper. **TODO: Move to related works: Previous works use computational metrics such as TF-IDF or *saliency* to extract keywords. However, computational metrics often cannot well encode high-level semantics. Instead, we rely on LLMs' ability to understand natural language to identify significant keywords from unstructured texts.**

We broke down the process of keyword extraction into two steps to avoid the problem of extrinsic hallucination as discussed by Bang et al. [BCL*23]. First, a sentence describing the most important event is generated from the summary. The event should be the primary focus of the news article and should describe the major objective of the news article, such as a geopolitical event like the G20 summit. Then a few-shot prompt was adopted to give the LLM examples of what kind of events would be described as the main event. Next, all the main keywords involved in that event are extracted with another few-shot prompt. The prompt transforms an extraction task into a question-answering task. An example prompt for a news article dataset is *'A major event is reported by a news article: {event_description}, what are the main keywords that are majorly involved in the event?'*. Providing the model with more context yields better accuracy and lower chances of generating a hallucinated output. The extracted keywords will then be disambiguated and used in the *Model* stage.

4.1.3. Keyword Disambiguation

TODO: try new LLM disambiguation method: embed explanation, find close ones, use prompt to disambiguate Once the main keywords are extracted, a disambiguation step is done to create connections between documents. In early experiments, we found that LLMs do not excel at disambiguating keywords. We thus prioritize using an entity linking model to disambiguate keywords. Entity linking is the task of mapping a named entity text mentions to their corresponding entities in a knowledge base [SWH14]. Existing models adopt a supervised learning approach, where a large amount of training data is required, so they are usually limited to a specific domain. For datasets that an existing entity-linking model [ATF*22] is available, we use it to disambiguate keywords. In cases where the keywords cannot be linked to an external knowledge base, we employ a matching algorithm based on embeddings. The algorithm is straightforward: For every main keyword extracted in the previous step, we embed them and create a pair-wise similarity matrix. Then the pairs of keywords with a similarity score above a threshold are considered to be the same keyword. We then use LLMs to generate a unified title for the matched keywords, which is used as the keyword's label in the user interface.

4.1.4. Document and Keyword Embedding

Embeddings are dense vectors that can be used to measure similarities between objects. In text analysis, embeddings are often created on words, sentences or documents. We adopted a similar approach here for both documents and keywords. For documents, we embed the summaries generated in subsubsection 4.1.1. For keywords that can be linked to an external knowledge base, we embed the description section of the keyword in the knowledge base. Otherwise, we first generate a description for the keyword using LLMs and then embed the description. The description is generated with a simple prompt: *'What is {keyword}?'*. In cases where domain knowledge is needed to generate a good description, a RAG-based (**TODO:**

citeRAG) prompt can be used. We used OpenAI's 'text-embedding-ada-002' model for all embeddings. This allows us to measure the semantic similarity between documents, keywords and user queries in the same vector space.

4.1.5. Topic Label Generation

TODO: shorten Later in the preprocessing pipeline, we will use a hierarchical clustering algorithm to cluster the documents and keywords (described in subsubsection 4.2.2). The algorithm outputs a hierarchy, where leaf nodes are documents or keywords and internal nodes are clusters. The clustering result faces a similar problem with embedding-based approaches, that the clusters are not interpretable. To address this issue, we use LLMs to assign human-like labels to each cluster. Our approach is similar to a recent work done by Raval et al. [RWVW23], where they use LLMs to generate explanations for a user-selected set of points. However, our task is more complicated because the prompt needs to (1) consider the hierarchical information, and (2) be able to process large clusters containing thousands of documents without breaking the token limit.

We describe our topic label generation process in a bottom-up manner: (1) At the bottom level (one level above the leaf nodes), assign topic labels to the clusters using the document summaries; (2) At any level above, assign labels to the clusters using the labels of their children and randomly sampled document summaries.

The first step is straightforward, where we simply use the document to generate labels for each cluster. The bottom-level clusters usually contain only a few documents, so the token limit is unlikely to be exceeded. We insert the summarization generated in subsubsection 4.1.1 into the prompt template. The prompt instructs the LLM to generate a label composed of a single noun phrase for the given documents.

Then at any level above, the cluster size increases dramatically and the token limit is likely to be exceeded. To solve this problem, we first insert the labels of a cluster's children instead of the document. This guarantees that the token limit will not be exceeded. In early experiments, we found that the children's labels alone were not enough to generate a meaningful label for the parent cluster. Therefore, we also insert documents randomly sampled from the cluster. We enforce that each child has at least one article being sampled, and distribute the remaining token space proportionally to the size of each child.

4.2. Models

4.2.1. Hypergraph Construction

A hypergraph is a generalization of a graph in which an edge can connect any number of nodes [FFKS21]. A hyperedge thus represents a multi-way relationship between nodes. In our work, we model two types of hypergraphs: document hypergraph and keyword hypergraph. In a document hypergraph, the nodes are documents and the hyperedges are keywords. Similarly, in a keyword hypergraph, the nodes are keywords and the hyperedges are documents. Analyzing the document hypergraph and keyword hypergraph correspond to topic-based and entity-based analysis, respectively. By modeling the corpus as hypergraphs, the system supports users to conduct both types of analysis simultaneously under a unified framework (**DC1**). Below, we explain the relation between these two types of hypergraphs and how they are constructed.

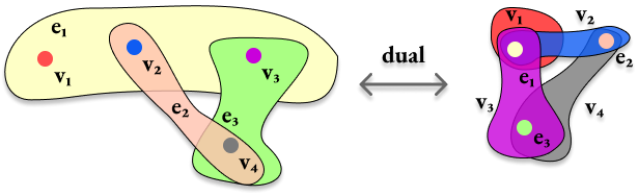


Figure 3: Illustration of the dual of a hypergraph. Left: a hypergraph with 4 nodes v_1, v_2, v_3, v_4 and 3 hyperedges $e_1 = (v_1, v_2, v_3), e_2 = (v_2, v_3), e_3 = (v_3, v_4)$. Right: the dual of the hypergraph, where the nodes and hyperedges interchanged. Nodes in the dual hypergraph are now e_1, e_2, e_3 and hyperedges are $v_1 = (e_1), v_2 = (e_1, e_2), v_3 = (e_1, e_3), v_4 = (e_2, e_3)$.

Following the definition of a hypergraph, a hyperedge can be used to represent two types of multi-way relationships: In a document hypergraph, a hyperedge (connecting document nodes) can be constructed between documents that mention the same keyword. In this case, the hyperedge represents the co-mention of a keyword. In a keyword hypergraph, a hyperedge (connecting keyword nodes) can be constructed between keywords if they are mentioned together in the same document. In this case, the hyperedge represents a co-occurrence relationship between keywords. This means the two hypergraphs are both built from the same corpus, using the same relationship information, but from different perspectives. Once the two hypergraphs are constructed, we further separately cluster them hierarchically. Clusters in the document hypergraph represent topics that are discussed in the dataset. Clusters in the keyword hypergraph represent keywords (entities or concepts) that are similar to each other. For better interpretability of the clustering result, we further assign *labels* to each cluster, as explained in subsubsection 4.1.5.

Although these two types of hypergraphs are constructed from different perspectives, we utilize the *dual* of a hypergraph to simplify the construction process. The dual of a hypergraph is simply another hypergraph, where the nodes and hyperedges are interchanged, as shown in Figure 3. Using the duality feature, we first model the documents as nodes and keywords as hyperedges to construct the document hypergraph H_D. The keyword hypergraph H_C can then be easily constructed by taking the dual of H_D. This construction process also allows us to map user interactions on documents and keywords to operations on a single hypergraph (**DC1**). Even better, the duality feature between documents and keywords matches the user's mental model (**DC3**), making the visualization and interaction intuitive.

4.2.2. Hierarchical Clustering

TODO: Move existing works to supplemental material Common clustering algorithms for graphs consider only graph connectivity. However, for the best interpretability of the clustering result, the node embeddings must be also used in the clustering process. This limits our choice of clustering algorithms to attributed node clustering algorithms. Although there are existing approaches that

can cluster attributed nodes on graphs such as EVA [CR20] and iLouvain [CLGEZ15], they are not designed for hypergraphs. In general, hypergraphs can be clustered in two different ways: (1) Directly operate on the hyperedges by generalizing the graph clustering algorithms. For example, Kamiński et al. [BK21] generalizes the modularity metric for graphs to hypergraphs; (2) First transform the hypergraph into a graph and then apply normal graph clustering algorithms [KVA*20]. Although the first approach is more intuitive, it is less scalable and hard to incorporate node attributes. Thus, we have decided to design our clustering algorithm following the second approach.

Considering all the above, we implemented our hierarchical clustering algorithm by first transforming the hypergraph into a graph following the edge re-weighting process proposed by Kumar et al. [KVA*20]. Then an agglomerative clustering algorithm [SKK00] is applied on the re-weighted graph. In agglomerative clustering, the key is to define node similarity and cluster similarity. We can easily incorporate node attributes into the clustering process by defining the similarity between nodes and clusters as a weighted sum of attribute similarity S_s and connectivity similarity S_c. Since we're dealing with texts, we refer to the attribute similarity between nodes as semantic similarity.

The semantic similarity $S_s(i,j)$ is the cosine similarity of the embeddings (dense vectors) of the two nodes, denoted as vec_i and vec_j. The connectivity similarity S_c is the weighted Topological Overlap (wTO) [GVF*18], which is a weighted generalization of the Overlap Coefficient [VK16], as shown in Equation 1.

$$\mathbf{S_s}(i,j) = \frac{vec_i \cdot vec_j}{||vec_i|| \cdot ||vec_j||}, \quad \mathbf{S_c}(i,j) = \frac{\sum_{u=1}^{N} w_{i,u} w_{u_j} + w_{i,j}}{\min(k_i, k_j) + 1 - |w_{i,j}|} \tag{1}$$

where $k_i = \sum_{j=1}^{N} |w_{i,j}|$ is the total weight of the edges connected to node i. Finally, a weighting factor α is used to balance the two similarities, as shown in Equation 2.

$$\mathbf{S} = \alpha \mathbf{S_s} + (1-\alpha)\mathbf{S_c} \tag{2}$$

For the similarity between clusters, we used centroid similarity, i.e. the similarity between two clusters is the similarity between the centroids of the two clusters. The algorithm takes a hypergraph $H = (V,E)$ and the embeddings of each node Vec as input, and outputs a sequence of partitions $P = P_1, P_2, \dots P_k$. Each partition corresponds to a level in the hierarchy, as shown in Algorithm 1.

Algorithm 1 Agglomerative Clustering

Input: $H = (V,E)$, $Vec = \{vec_i | i \in V\}$
Output: Sequence of partitions $\mathbf{P} = P_1, P_2, \dots P_k$

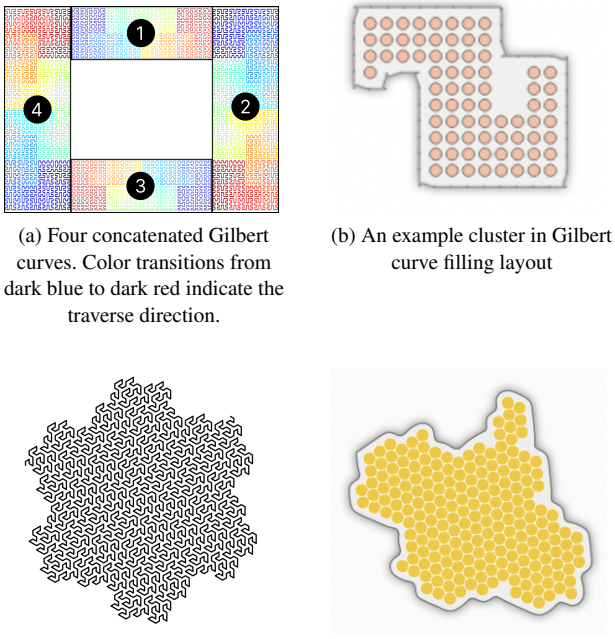
while $|V| > 1$ **do**
  $P_k = init\_partition(V)$ ▷ Initialize each node as a cluster
  $\mathbf{S_s} = cosine\_similarity(V \times V, Vec)$
  $\mathbf{S_c} = wTO(V \times V, E)$
  for $i \in V$ **do**
    $j = most\_similar\_node(i, S_s, S_c)$
    $P_k = merge\_clusters(i, j)$ ▷ Merge the two clusters
  end for
  $H' = (V', E') = construct\_hypergraph(P_k)$ ▷ clusters are the new nodes
  $Vec' = centroid\_similarity(V')$
  $V = V', E = E', Vec = Vec'$ ▷ Update for next iteration
end while

5. Visualization of HyperMap

5.1. SFC for HyperGraph

In the design guidelines derived by Abdelaal et al. [ASA*22] in a recent network visualization evaluation study, node-link-based approaches are recommended when: (1) tasks involve the identification of network clusters, and (2) the network is sparse. Condition (1) is fulfilled as explained in section 3, and (2) is guaranteed by the keyword extraction process described in subsubsection 4.1.2. Therefore, we decided to use node-link-based approaches for our system.

TODO: Move existing works to supplemental material Although there are a variety of node-link-based approaches for hypergraph visualization [FFKS21], we find the extra-node representation



(a) Four concatenated Gilbert curves. Color transitions from dark blue to dark red indicate the traverse direction.

(b) An example cluster in Gilbert curve filling layout

(c) An order-4 Gosper curve

(d) An example cluster in Gosper curve filling layout

Figure 4: Illustrations of the space-filling curves and example clusters formed by the curves.

introduced by Ouvrard et al. [OGM17] most flexible and intuitive. An extra-node representation improves existing clique-expansion of hypergraphs by adding extra nodes to represent hyperedges. The extra-node representation effectively transforms the hypergraph visualization problem into a bipartite graph visualization problem. After that, any node-link-based graph visualization method can be applied. In our system, we use the space-filling curve (SFC) layout method to lay out the extra-node representation of the hypergraph. The SFC layout method uses pre-computed clustering to order nodes in a sequence and then applies a space-filling curve on the node sequence to map it to a two-dimensional screen space [MM08]. SFC approaches are known for their efficiency and aesthetics in visualizing large graphs [MM13]. Moreover, SFC layouts support progressive disclosure (**DC2**) organically, as the layout is generated based on the clustering result (**DC3**). After the preprocessing and modeling stage described in section 4, we have two hypergraphs: the document hypergraph H_D and the keyword hypergraph H_C, each having its hierarchical cluster. Combining the extra-node representation and SFC layout, we visualize the H_D and H_C as two separate SFCs, as shown in Figure 1.

Specifically, we divide the layout space into two parts: the peripheral and the center area. For the peripheral area, we concatenate four Gilbert curves [Če19]. A Gilbert curve is a generalized version of the Hilbert curve, which instead of filling a squared area, can traverse any rectangular region in a way similar to the Hilbert Curve. In Figure 4-a, the first Gilbert curve starts from the lower left (dark blue) and ends at the lower right (dark red). Through rotation and flipping, the start and end curve points for neighboring Gilbert curves can be concatenated smoothly. The use of concatenated Gilbert curves allows us to fill the peripheral space while having the efficiency and aesthetics of SFC layouts.

The curve to be used for the center area is technically unbounded. In early prototyping, we found that using the same curve as the peripheral region was confusing for the user, as it was hard to distinguish between the peripheral and center areas. We decided to use a simple Gosper curve (Figure 4-c) to lay out the nodes for better aesthetics. The resulting visualization looks similar to GosperMap [AHL*13], but we did not employ the advanced techniques proposed in GosperMap. The interactions to support the exploration and reorganization of the dataset are the main focus of the system, which are also not limited to any specific curve.

After the curves are generated, we can apply the curves on the node sequences to generate the two-dimensional layout. We chose to put the document hypergraph in the center area because the documents are the main analysis targets for the user. Consequently, the keyword hypergraph is put in the peripheral area.

5.2. Improving the readability

Automatic Cluster Expansion In most cases, the default clustering result is not optimal for the user's targeted analysis tasks. We identify two common problems in early prototyping: (1) clusters may be too big, weakening the semantic meaning that the clusters can convey. (2) clusters may have only one sub-cluster, which makes the parent cluster redundant. Both problems can be mitigated by automatically expanding a cluster, i.e.breaking the cluster into sub-clusters Figure 6-d. We employ rule-based detection to identify clusters that need to be expanded. For a cluster C, we expand it if the following conditions are met: (1) C has only one sub-cluster C_s; (2) C has more than $n = kN$ nodes, where $k \in [0,1]$ and N is the size of the hypergraph. Through trial and error, we find that $k = 0.3$ gives the most balanced results.

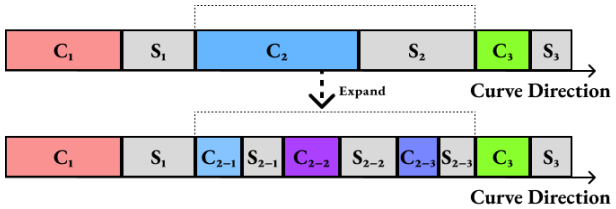


Figure 5: Spacing strategy of the SFC layouts. The space of each cluster S_i is proportional to the cluster size. When C_i is expanded, its sub-clusters redistribute S_i to ensure the expansion only affects locally.

Spacing Strategy Spacing between each node is important for the readability and aesthetics of SFC layouts. To highlight different clusters, we employ our spacing strategy on clusters instead of nodes. Given a space-filling curve of a specific order, we first calculate the length of the curve L. L represents the total amount of space available for the nodes and thus, $L-N$ represents the amount of space to be redistributed, where N is the size of the hypergraph. Our goal is to distribute the space between clusters to ensure the best readability. In early prototyping, we found that distributing the space proportional to the cluster size gives the best readability as well as stability. Specifically, we define the space of a cluster as the blank space it has behind it on the curve, which is calculated by $(L-N)\frac{N_c}{N}$, where N_c is the size of the cluster (Figure 5). Another consideration when designing our spacing strategy is stability. We want to ensure that the layout change is minimized when clusters are expanded. Naively, when a cluster is expanded, the whole layout needs to be recalculated because now the cluster sequence is changed. To avoid such recalculation, we design our spacing strategy in this way so that the sub-clusters can simply take over the space of their parent cluster, as shown in Figure 5. Since the total volume of the sub-clusters is the size of the parent cluster and the space is proportional, the sub-clusters can take over exactly the space of the parent cluster without any overflow or underflow. This ensures a local change in the layout when a cluster is expanded.

Concave Hull Approximation After applying the SFC layout, we use a concave hull algorithm [PO12] to generate an approximation polygon for each cluster. The polygons are used to generate borders and calculate label positions for the clusters. The concave hull is generated based on a cluster of points, in our case, the nodes in a cluster. This means that the algorithm can be applied to any curve we used for the SFC layout. This is a desirable property because we are using two different curves in our system, and the center area curve choice is flexible. Using the same algorithm guarantees a unified aesthetic across curves.

The original concave hull algorithm is designed for approximating points, but we need to approximate circles. Naively we can use the center of the circles as the points for the algorithm, but this would

result in a polygon that is too small and crosses the circles on the boundary. To address this issue we use a simple trick: we add extra points to the cluster by extrapolating the original points. For example, a Gilbert curve moves perpendicularly, so the resulting polygon would have perpendicular corners. We can therefore add eight extra points around each original point so that the extrapolation forms a three-by-three grid. On the boundary of the cluster, these extra points prevent the concave hull from passing across the original points. Since the concave hull algorithm has an $O(n \log n)$ time complexity, the performance overhead introduced by the extra points is negligible.

Borders The borders are generated by applying a smoothing algorithm on the polygons. For Gosper curves, we use the polygon as control points to generate a cubic basis spline as the border. For Gilbert curves, we use a similar approach but with a cubic Bezier curve. More specifically, for each pair of consecutive points, we use a smoothing factor to interpolate the control point. This results in a sketchy style at the border corners. Two examples are given in Figure 4. **TODO: add overlap prevention**

Label Position Labeling the clusters is essential for users to explore the dataset. We use the topic assignment described in sub-subsection 4.1.5 to label the clusters. When using SFC layouts, determining the label position automatically is challenging because the shape of the clusters can be irregular. For a cluster, the label position is simply the centroid of the polygon. When a cluster is expanded through user interaction, the sub-clusters within need to be clearly labeled as well. Using the centroid of the sub-cluster as the label position is not a good choice because the label would cause a serious cluttering issue. Therefore, for a sub-cluster, we first calculate its centroid, and then we extend the line from the parent cluster centroid to the sub-cluster centroid. Once the intersection point of the extended line and the parent border is found, we extend the line by a fixed amount to avoid any overlapping issues. This results in a radial layout for the sub-cluster labels, as shown in Figure 6-d. The generalizability of the concave hull algorithm makes our labeling position calculation applicable to any curve we use for the SFC layout.

6. HyperMap System Design

Below, we describe the HyperMap frontend system, including the views, visualizations and interactions, and how they assist exploration and reorganization of a corpus. The HyperMap system consists of three main views: Cluster View, Article View and Analysis View. The Cluster View visualizes the corpus as two hypergraphs using the SFC layout described in section 5. The Document View shows the articles along with necessary statistics when the user makes a selection in the Cluster View. The Analysis View integrates an LLM-based chatbot to assist the user in analyzing the selected articles.

6.1. Cluster View

Cluster View is the main view of the HyperMap system (Figure 1-a and -b). It visualizes the corpus as two hypergraphs using the SFC layout described in section 5. Using the Cluster View, users can explore the topic structure and keyword connections simultaneously (**DC1**). Below, we mainly discuss user interactions and the coordination with other views.

6.1.1. Interactions

Hover and click By default, the cluster labels are hidden to reduce clutter. Hovering over the clusters will trigger a highlight effect and show the cluster label, indicating to users that it is an interactive object. Users can select a cluster by clicking on the cluster label. This will trigger the Article View to show articles in the cluster and the mentioned keywords (**DC1**). Additionally, clicking on the cluster will temporarily expand the cluster to expose its sub-structure (**DC2**), as shown in Figure 6-b. The sub-structure is colored in different colors while maintaining the original cluster's shape. The labels of each sub-cluster are also shown radially. When the user clicks on any cluster or cluster label, the mentioned keywords are highlighted and others fade out. Under such cases, hovering over the keyword clusters will show not only the cluster label but also the highlighted keywords in a list, as shown in Figure 1-a. Users can also click on the mentioned keywords to filter the articles in the Article View.

Expansion The expansion operation breaks a cluster into smaller clusters. This operation is necessary because the agglomerative clustering result is not always semantically optimal, and some clusters may be too vague for the user. By breaking down a cluster, users can investigate a level deeper into the topic structure (**DC2**). We use (Cmd + Click) or (Ctrl + Click) to expand a cluster. When clicked, a smooth animation triggers showing how the parent cluster is broken down into smaller pieces. The sub-clusters will redistribute the spacing of their parent cluster proportionally to their size. This ensures that only local changes in the layout need to be made when expanding a cluster.

Filtering At any point in the exploration when users find that they have found the target of interest, they can click the filter button to remove irrelevant documents and keywords from the view. The filtering functionality effectively creates a sub-hypergraph based on node selection, supporting any interactions that are available in the original hypergraph. Note that we only support filtering based on document node selection, and keyword nodes that are not mentioned in the selected documents are filtered accordingly. Although filtering based on keyword nodes is technically possible, we do not find the operation intuitive. We decided to remove this feature to prevent users from losing themselves in the exploration process.

Searching HyperMap supports searching by document embeddings. Users can create any query in natural language, and the system will return the most relevant documents (above Figure 1-c). The search functionality is implemented by ranking the documents based on their relevancy to the query using cosine similarity. The user query is first embedded into the same vector space as the documents, then the relevancy score between the query and each document is calculated. The server returns the documents sorted by relevancy score to the front end, and the user can control the number of documents to be highlighted by a relevancy threshold. The highlighted documents and mentioned keywords are visually distinguished in the Cluster View (Figure 6-b).

6.2. Document View

The Document View displays a list of selected articles (Figure 1-c). It can be a list of documents in a cluster or a list of documents that are returned by the search functionality. Each document is an interactive card, with the title, summary, ID and relevancy score (if available) shown. The main keywords of the document are highlighted in yellow in the summary to assist quick browsing. If the user is under search mode, documents are sorted according to relevancy and those above the relevancy threshold are highlighted.

7. Usage Scenario

We illustrate the use of HyperMap through a usage scenario where a journalist is collecting materials for a news story. In section 8, we evaluate our system for conducting a literature review. In this scenario, we use the *All The News* [all] dataset to demonstrate the effectiveness of HyperMap in assisting the exploration of the topics and keywords covered in news articles during a specific time period. We randomly sampled 8192 news articles published in 2016 from the dataset for this case study.

Alice is a journalist currently working on a story about the US presidential election. She wants to collect articles about the 2016 election as materials for her story. She can only vaguely remember certain events back when the election was happening, so she wants to freshen up her memory and find real evidence to reference. Alice begins by opening the HyperMap system displaying the All The News dataset and searches 'US presidential election'. The Cluster View highlights relevant articles in each of the six topics with clear labels and colors (Figure 6-a), and she can easily identify topics that are not related to the election. such as 'Ride-hailing Technology'. Before she filters the whole corpus with current search results, she wants to make sure that the relevancy threshold is reasonable so that she does not miss too many relevant articles. The topic 'Security Threats in the Middle East' caught her attention because she reckons that the seemingly unrelated topic might be related to the election since foreign policy towards the Middle East has always been a hot topic in the election. She clicks on the topic label to highlight the topic and keywords. She hovers over each of the highlighted keyword clusters, but she does not find any keywords related to the election. Upon quickly scanning through the titles of the highlighted articles, she confirms that the topic is not related to the election and raises the threshold until the cluster is filtered out.

After that, she finds the remaining topics are too vague to tell what they are about. She decides to first drill down the topic 'Controversies and Challenges' by clicking and expanding it (Figure 6-b). She finds that 'Criminal Justice' and 'Political Protests against the Trump election' are clearly two sub-topics that are related to the election. Upon reading the summaries of the articles, she confirms the accuracy of these two sub-topic labels and marks them down.

After such an inspection, she is quite confident with the current relevancy threshold. She then clicks the 'Filter Search' button to reorganize the corpus based on the search results (Figure 6-c). Once again, the topic Concerns and Controversies in the United States seems to contain a lot of related articles, but she could not tell what the topic is about. To dive deeper, she splits the cluster into smaller, separated sub-clusters (Figure 6-d). The sub-clusters contain a wide range of topics, one of which being 'The 2016 US Presidential Election and Controversies'. Upon inspecting the mentioned keywords, she becomes curious about how 'Vladimir Putin' is related to the election, so she clicks the keyword label to inspect the articles. She also finds several interesting topics, such as 'Presidential Eligibility Confusion'. She could not remember such controversies at first, but by browsing the highlighted keywords and reading the titles of the articles, she understands that it was a controversy between Donald Trump and Ted Cruz. The wide range of topics brings her memory back, and she continues to click on every topic she deems relevant to the election to collect materials for her own story.

This case study demonstrates that HyperMap provides highly interpretable visualization of the topic structure and keyword connections in a corpus. Alice can quickly understand the topic if the labels are accurate, and she can always drill down to a deeper level to find more specific topics if the labels are too vague. The keyword connections as well as article cards provide her with enough information to decide whether a topic is relevant to her story. In the next case study, we demonstrate how the Analysis View can assist users in analyzing the selected documents.

8. User Study

9. Limitations and Future Work

TODO: shorten The case studies demonstrate that HyperMap provides highly interpretable visualization of the topic structure and keyword connections and flexible interaction to reorganize and analyze a corpus. During the usage, the clusters and borders help users clearly distinguish different clusters. The topic labels assigned by LLMs coupled with keyword connections provide rich semantic information for users to make sense of the topics. Most importantly, the user is able to focus on their domain-specific analysis tasks and does not need to understand the underlying model. Analysts from all disciplines are not expected to understand LLMs, hypergraphs or clustering algorithms, but they can still explore, reorganize and analyze the corpus. We attribute this outcome to a good model alignment. Hypergraphs allow us to directly map user interactions on both documents and keywords, two seemingly unrelated units of analysis, to operations on one unified model. LLMs assist in generating clean and interpretable preprocessed data for the modeling process. Combined, the visualization can be designed in a way that all preprocessing and model details are irrelevant to the user during analysis.

However, there are still several limitations in the current approach. To use LLMs for information extraction, the prompts need to be redesigned for every dataset. Although the general template structure is the same, optimizing prompts is a time-consuming trial-and-error process. To make it worse, the evaluation of the accuracy of a prompt relies on human judgment. This means that the preprocessing result is not guaranteed to be error-free. Addressing the evaluation challenge is a promising direction for future work. We envision a combination of LLMs, computational evaluation metrics and visualizations to assist prompt evaluation. Another limitation is the amount of user interactions needed to converge on a satisfying corpus organization. Although previous studies [BKH*22] suggest that users gain more trust and confidence while interacting with the system, it

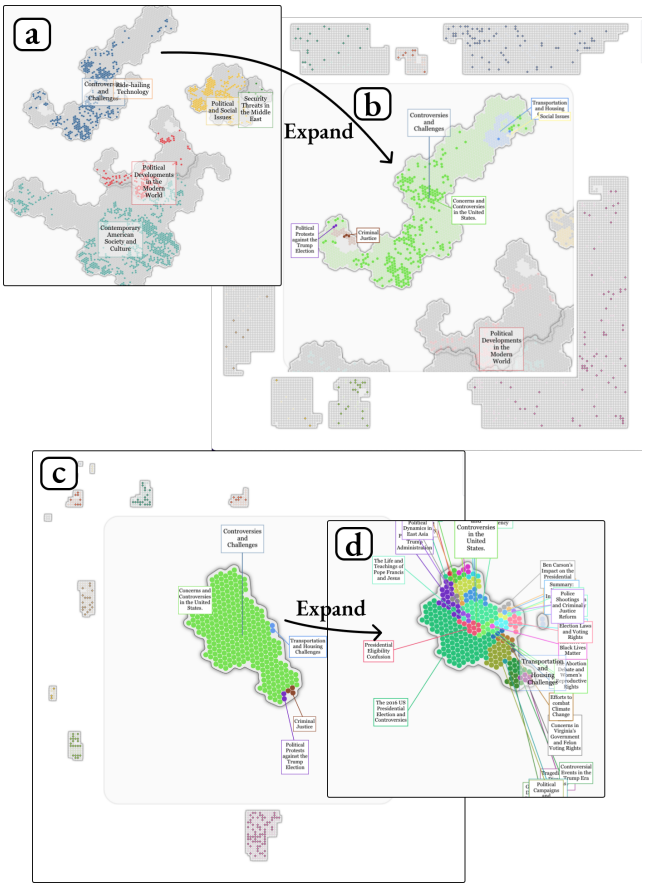


Figure 6: User interactions in usage scenario (a) The search result of 'US presidential election'. Relevant articles are highlighted. (b) The user expands the topic 'Controversies and Challenges', showing the sub-topics within and mentioned keywords in the peripheral area. (c) After clicking 'Filter Search', irrelevant articles are removed, and the layout is regenerated. (d) The user expands a sub-topic 'Concerns and Controversies in the United States'.

would be better if the system could automatically generate a satisfying organization based on user feedback. Our current automatic cluster expansion strategy only deals with expansion, and it does not consider user intent. We believe LLM's ability to understand user intent can be utilized to address this limitation. Finally, the Analysis View can adopt approaches from the prompt engineering facilitation tools [PTC23, DGLB23], or integrate more advanced document retrieval techniques [QTW*22] to provide better support for analysis Overall, with high interpretability and generalizability, it is possible to posit the users as the center of the analysis, opening up many possibilities for future work.

10. Conclusion

In this paper, we propose HyperMap, a visual analytics system that assists users in exploring, reorganizing and analyzing a corpus. Previous works have shown that topic- and entity-based analysis are essential to sensemaking on a corpus, but existing text analysis tools

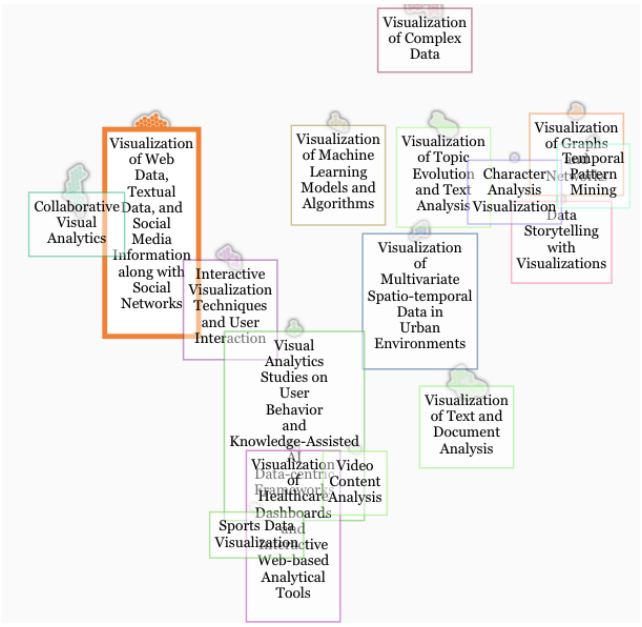


Figure 7: The VisPub corpus visualization filtered by the keyword 'text analysis' and reorganized by the user in usage scenario 2. After expansion and filtering, the remaining topics are at a reasonable level of granularity. The user is currently selecting the orange cluster 'Visualization of Web Data, Textual Data and Social Media Information' to prompt the Analysis View for more detail.

do not provide a unified interface for both types of analysis. We fill in this gap by combining state-of-the-art large language models and hypergraph analysis and visualization techniques. We introduce an LLM-based pipeline to extract topics and keywords from unstructured text documents. We model the corpus as hypergraphs and apply an agglomerative clustering algorithm. The clusters are visualized by building upon existing space-filling curve layouts, which exhibit a high level of visual scalability and aesthetics. Moreover, we emphasize the importance of Model Alignment in the design of visual analytics systems. The generalizability and effectiveness of HyperMap are demonstrated in two case studies, analyzing datasets from two different domains. Our future research aims to utilize the capabilities of LLMs to understand user intent to support more advanced analysis tasks.

References

[ACS*23] ATZBERGER D., CECH T., SCHEIBEL W., TRAPP M., RICHTER R., DÖLLNER J., SCHRECK T.: Large-scale evaluation of topic models and dimensionality reduction methods for 2d text spatialization. *arXiv preprint arXiv:2307.11770* (2023). 2

[AHL*13] AUBER D., HUET C., LAMBERT A., RENOUST B., SALLABERRY A., SAULNIER A.: Gospermap: Using a gosper curve for laying out hierarchical data. *IEEE transactions on visualization and computer graphics 19*, 11 (2013), 1820–1832. 7

[AKV*14] ALEXANDER E., KOHLMANN J., VALENZA R., WITMORE M., GLEICHER M.: Serendip: Topic model-driven visual exploration of text corpora. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)* (2014), pp. 173–182. doi:10.1109/VAST.2014.7042493. 2

[all] All The News. https://components.one/datasets/all-the-news-2-news-articles-dataset. 9

[ASA*22] ABDELAAL M., SCHIELE N. D., ANGERBAUER K., KURZHALS K., SEDLMAIR M., WEISKOPF D.: Comparative evaluation of bipartite, node-link, and matrix-based network representations. *IEEE Transactions on Visualization and Computer Graphics 29*, 1 (2022), 896–906. 6

[ATF*22] AYOOLA T., TYAGI S., FISHER J., CHRISTODOULOPOULOS C., PIERLEONI A.: ReFinED: An efficient zero-shot-capable approach to end-to-end entity linking. In *NAACL* (2022). 4

[BCL*23] BANG Y., CAHYAWIJAYA S., LEE N., DAI W., SU D., WILIE B., LOVENIA H., JI Z., YU T., CHUNG W., ET AL.: A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023* (2023). 3, 4

[BFR21] BEASLEY Z. J., FRIEDMAN A., ROSEN P.: Through the looking glass: insights into visualization pedagogy through sentiment analysis of peer review text. *IEEE Computer Graphics and Applications 41*, 6 (2021), 59–70. 2

[BK21] BOGUMIŁ KAMIŃSKI P. P. . F. T.: Community detection algorithm using hypergraph modularity. In *Complex Networks & Their Applications IX: Volume 1, Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020* (2021), Springer, pp. 152–163. 6

[BKH*22] BACH T. A., KHAN A., HALLOCK H., BELTRÃO G., SOUSA S.: A systematic literature review of user trust in ai-enabled systems: An hci perspective. *International Journal of Human–Computer Interaction* (2022), 1–16. 9

[CDW*16] CHO I., DOU W., WANG D. X., SAUDA E., RIBARSKY W.: Vairoma: A visual analytics system for making sense of places, times, and events in roman history. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (2016), 210–219. doi:10.1109/TVCG.2015.2467971. 2

[CLGEZ15] COMBE D., LARGERON C., GÉRY M., EGYED-ZSIGMOND E.: I-louvain: An attributed graph clustering method. In *Advances in Intelligent Data Analysis XIV: 14th International Symposium, IDA 2015, Saint Etienne. France, October 22-24, 2015. Proceedings 14* (2015), Springer, pp. 181–192. 6

[CLRP13] CHOO J., LEE C., REDDY C. K., PARK H.: Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 1992–2001. doi:10.1109/TVCG.2013.212. 2

[CR20] CITRARO S., ROSSETTI G.: Eva: Attribute-aware network segmentation. In *Complex Networks and Their Applications VIII: Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019 8* (2020), Springer, pp. 141–151. 6

[CRMH12] CHUANG J., RAMAGE D., MANNING C., HEER J.: Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of the SIGCHI conference on human factors in computing systems* (2012), pp. 443–452. 1, 2, 3

[CSL*10] CAO N., SUN J., LIN Y.-R., GOTZ D., LIU S., QU H.: Facetatlas: Multifaceted visualization for rich text corpora. *IEEE transactions on visualization and computer graphics 16*, 6 (2010), 1172–1181. 2

[DCLT18] DEVLIN J., CHANG M.-W., LEE K., TOUTANOVA K.: Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018). 2

[DGLB23] DANG H., GOLLER S., LEHMANN F., BUSCHEK D.: Choice over control: How users write with large language models using diegetic and non-diegetic prompting. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (2023), pp. 1–17. 10

[DYW*13] DOU W., YU L., WANG X., MA Z., RIBARSKY W.: Hierarchicaltopics: Visually exploring large text collections using topic hierarchies. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2002–2011. 2

[FFKS21] FISCHER M. T., FRINGS A., KEIM D. A., SEEBACHER D.: Towards a survey on static and dynamic hypergraph visualizations. In *2021 IEEE visualization conference (VIS)* (2021), IEEE, pp. 81–85. 5, 6

[GVF*18] GYSI D. M., VOIGT A., FRAGOSO T. D. M., ALMAAS E., NOWICK K.: wto: an r package for computing weighted topological overlap and a consensus network with integrated visualization tool. *BMC bioinformatics 19*, 1 (2018), 1–16. 6

[HPK*22] HAN D., PARSAD G., KIM H., SHIM J., KWON O.-S., SON K. A., LEE J., CHO I., KO S.: Hisva: A visual analytics system for studying history. *IEEE Transactions on Visualization and Computer Graphics 28*, 12 (2022), 4344–4359. doi:10.1109/TVCG.2021.3086414. 2

[ICH*22] IZACARD G., CARON M., HOSSEINI L., RIEDEL S., BOJANOWSKI P., JOULIN A., GRAVE E.: Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research* (2022). URL: https://openreview.net/forum?id=jKN1pXi7b0. 2

[KOM*20] KARPUKHIN V., OGUZ B., MIN S., LEWIS P., WU L., EDUNOV S., CHEN D., YIH W.-T.: Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), Association for Computational Linguistics, pp. 6769–6781. URL: https://aclanthology.org/2020.emnlp-main.550, doi:10.18653/v1/2020.emnlp-main.550. 2

[KVA*20] KUMAR T., VAIDYANATHAN S., ANANTHAPADMANABHAN H., PARTHASARATHY S., RAVINDRAN B.: A new measure of modularity in hypergraphs: Theoretical insights and implications for effective clustering. In *Complex Networks and Their Applications VIII: Volume 1 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019 8* (2020), Springer, pp. 286–297. 6

[LFY*23] LI B., FANG G., YANG Y., WANG Q., YE W., ZHAO W., ZHANG S.: Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633* (2023). 3

[LKC*12] LEE H., KIHM J., CHOO J., STASKO J., PARK H.: ivisclustering: An interactive visual document clustering via topic modeling. In *Computer graphics forum* (2012), vol. 31, Wiley Online Library, pp. 1155–1164. 2

[LSS*17] LEE T. Y., SMITH A., SEPPI K., ELMQVIST N., BOYD-GRABER J., FINDLATER L.: The human touch: How non-expert users perceive, interpret, and fix topic models. *International Journal of Human-Computer Studies 105* (2017), 28–42. 2

[MM08] MUELDER C., MA K.-L.: Rapid graph layout using space filling curves. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1301–1308. 2, 7

[MM13] MA K.-L., MUELDER C. W.: Large-scale graph visualization and analytics. *Computer 46*, 7 (2013), 39–46. doi:10.1109/MC.2013.242. 7

[MS23] MADDIGAN P., SUSNJAK T.: Chat2vis: Generating data visualisations via natural language using chatgpt, codex and gpt-3 large language models. *IEEE Access* (2023). 2

[NJM21] NASAR Z., JAFFRY S. W., MALIK M. K.: Named entity recognition and relation extraction: State-of-the-art. *ACM Computing Surveys (CSUR) 54*, 1 (2021), 1–39. 3

[NKWW22] NARECHANIA A., KARDUNI A., WESSLEN R., WALL E.: vitality: Promoting serendipitous discovery of academic literature. 2

[OGM17] OUVRARD X., GOFF J. L., MARCHAND-MAILLET S.: Networks of collaborations: Hypergraph modeling and visualisation. *CoRR abs/1707.00115* (2017). URL: http://arxiv.org/abs/1707.00115, arXiv:1707.00115. 7

[OSR*14] OELKE D., STROBELT H., ROHRDANTZ C., GUREVYCH I., DEUSSEN O.: Comparative exploration of document collections: a visual analytics approach. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 201–210. 2

[PKL*18] PARK D., KIM S., LEE J., CHOO J., DIAKOPOULOS N., ELMQVIST N.: Conceptvector: Text visual analytics via interactive lexicon building using word embedding. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (2018), 361–370. doi:10.1109/TVCG.2017.2744478. 2

[PO12] PARK J.-S., OH S.-J.: A new concave hull algorithm and concaveness measure for n-dimensional datasets. *Journal of Information science and engineering 28*, 3 (2012), 587–600. 7

[PTC23] PETRIDIS S., TERRY M., CAI C. J.: Promptinfuser: Bringing user interface mock-ups to life with large language models. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (2023), pp. 1–6. 10

[QTW*22] QIU R., TU Y., WANG Y.-S., YEN P.-Y., SHEN H.-W.: Docflow: A visual analytics system for question-based document retrieval and categorization. *IEEE Transactions on Visualization and Computer Graphics* (2022). 2, 10

[RWVW23] RAVAL S., WANG C., VIÉGAS F., WATTENBERG M.: Explain and trust: An interactive machine learning framework for exploring text embeddings. *IEEE Transactions on Visualization and Computer Graphics* (2023). 2, 5

[SGLS07] STASKO J., GORG C., LIU Z., SINGHAL K.: Jigsaw: Supporting investigative analysis through interactive visualization. In *2007 IEEE Symposium on Visual Analytics Science and Technology* (2007), pp. 131–138. doi:10.1109/VAST.2007.4389006. 2

[SKK00] STEINBACH M., KARYPIS G., KUMAR V.: A comparison of document clustering techniques. 2, 6

[SNMM18] SHERKAT E., NOURASHRAFEDDIN S., MILIOS E. E., MINGHIM R.: Interactive document clustering revisited: A visual analytics approach. In *23rd International Conference on Intelligent User Interfaces* (2018), pp. 281–292. 2

[SWH14] SHEN W., WANG J., HAN J.: Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering 27*, 2 (2014), 443–460. 4

[TLW*23] TU Y., LI O., WANG J., SHEN H.-W., POWAŁKO P., TOMESCU-DUBROW I., SLOMCZYNSKI K. M., BLANAS S., JENKINS J. C.: Sdrquerier: A visual querying framework for cross-national survey data recycling. *IEEE Transactions on Visualization and Computer Graphics* (2023). 2

[VK16] VIJAYMEENA M., KAVITHA K.: A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal 3*, 2 (2016), 19–28. 6

[VK20] VAYANSKY I., KUMAR S. A.: A review of topic modeling methods. *Information Systems 94* (2020), 101582. 1

[WTL23] WANG C., THOMPSON J., LEE B.: Data formulator: Ai-powered concept-driven visualization authoring. *IEEE Transactions on Visualization and Computer Graphics* (2023). 2

[XW19] XIANG W., WANG B.: A survey of event extraction from text. *IEEE Access 7* (2019), 173111–173137. 3

[YTJ*19] YAN Y., TAO Y., JIN S., XU J., LIN H.: An interactive visual analytics system for incremental classification based on semi-supervised topic modeling. In *2019 IEEE Pacific Visualization Symposium (PacificVis)* (2019), pp. 148–157. doi:10.1109/PacificVis.2019.00025. 2

[YYQ17] YANG Y., YAO Q., QU H.: Vistopic: A visual analytics system for making sense of large document collections using hierarchical topic modeling. *Visual Informatics 1*, 1 (2017), 40–47. 2

[ZLZ23] ZHANG H., LIU X., ZHANG J.: Extractive summarization via chatgpt for faithful summary generation. *arXiv preprint arXiv:2304.04193* (2023). 3

[ZZPC23] ZHOU W., ZHANG S., POON H., CHEN M.: Context-faithful prompting for large language models, 2023. arXiv:2303.11315. 3

[Če19] ČERVENÝ J.: https://github.com/jakubcerveny/gilbert/commits/master generalized hilbert ("gilbert") space-filling curve for rectangular domains of arbitrary (non-power of two) sizes., 2019. URL: https://github.com/jakubcerveny/gilbert/commits/master. 7