Expert Systems

MapIntel: A visual analytics platform for competitive intelligence

David Silva
orcid.org/0000-0003-0357-3085
NOVA IMS, NOVA University of Lisbon, Campus de Campolide, Lisbon, Portugal, , email: dsilva@novaims.unl.pt

Fernando Bação
orcid.org/0000-0002-0834-0275
NOVA IMS, NOVA University of Lisbon, Campus de Campolide, Lisbon, Portugal

This is the peer reviewed version of the following article:

Silva, D., & Bação, F. (2023). MapIntel: A visual analytics platform for competitive intelligence. Expert Systems, [e13445]. https://doi.org/10.1111/exsy.13445

MapIntel: A Visual Analytics Platform for Competitive Intelligence

David Silva*[0000−0003−0357−3085] and Fernando Bação[0000−0002−0834−0275]

NOVA IMS, NOVA University of Lisbon, Campus de Campolide, 1070-312 Lisbon, Portugal
dsilva@novaims.unl.pt bacao@novaims.unl.pt

Abstract. Competitive Intelligence allows an organization to keep up with market trends and foresee business opportunities. This practice is mainly performed by analysts scanning for any piece of valuable information in a myriad of dispersed and unstructured sources. Here we present MapIntel, a system for acquiring intelligence from vast collections of text data by representing each document as a multidimensional vector that captures its own semantics. The system is designed to handle complex Natural Language queries and visual exploration of the corpus, potentially aiding overburdened analysts in finding meaningful insights to help decision-making. The system *searching* module uses a retriever and re-ranker engine that first finds the closest neighbors to the query embedding and then sifts the results through a cross-encoder model that identifies the most relevant documents. The *browsing* or visualization module also leverages the embeddings by projecting them onto two dimensions while preserving the multidimensional landscape, resulting in a map where semantically related documents form topical clusters which we capture using topic modeling. This map aims at promoting a fast overview of the corpus while allowing a more detailed exploration and interactive information encountering process. We evaluate the system and its components on the 20 newsgroups dataset, using the semantic document labels provided, and demonstrate the superiority of Transformer-based components. Finally, we present a prototype of the system in Python and show how some of its features can be used to acquire intelligence from a news article corpus we collected during a period of 8 months.

Keywords: Sentence embeddings · Transformer architecture · Visual analytics · Information retrieval · Topic modeling · Competitive intelligence

1 Introduction

Competitive Intelligence (CI) is the process and forward-looking practices used in producing knowledge about the competitive environment to improve organizational performance (Madureira et al., 2021).

* Corresponding Author

In the last decade, the digitalization of the market and the growth of the data economy have pushed the business environment to an online realm where every action and event is public and thus potentially relevant for decision-making. Despite the increased availability, CI resources are dispersed through a variety of websites and the underlying data is unstructured and noisy. These characteristics add to the difficulty of the CI analyst's task and exacerbate the need for tools to support it.

The goal of this work is to enhance the analyst's task by providing a means to explore, organize and visualize the environmental data present in the array of existing sources.

Various studies have attempted to create systems for exploring and gathering intelligence from extensive collections of textual data (Dey et al., 2011; Esteva et al., 2020; Lafia et al., 2019, 2021; Caillou et al., 2021). These studies have consistently applied Natural Language Processing (NLP) techniques to help users comprehend large volumes of text without requiring to sift through every document. Dey et al. (2011) designed a system for CI that captures data from multiple sources, cleans it, uses NLP to identify and tag the relevant content, stores it, generates consolidated reports, and produces alerts on predefined triggers.

Although the previously mentioned systems have been used for dealing with large amounts of text successfully, insufficient attention has been paid to the exploratory and serendipitous aspects of the analyst's task. Accordingly, we propose an information environment that supports analysts in having stimulating and productive information encounters. Thus, contrarily to previous systems, we center ours around Information Encountering which is defined by Erdelez and Makri (2020) to encompass "finding interesting, useful or potentially useful information when looking for some other information, not looking for any information in particular, or not looking for information at all." This is achieved by incorporating two types of information acquisition tasks: *searching*, consisting of an information retrieval module that allows *ad hoc* queries on the entire document collection, giving the user the ability to seek information actively, and *browsing*, consisting of a visualization module that equips the user with tools to actively or passively acquire information through the visual exploration of the document corpus (and its thematic cohorts) in a two-dimensional map.

With the recent emergence of the Transformer architecture (Vaswani et al., 2017), significant improvements were made in several NLP subdomains, having reached state-of-the-art results in a wide range of tasks (Vaswani et al., 2017). This new architecture is based solely on the attention mechanism, providing parallelization capabilities and thus avoiding the sequential nature of existing Recurrence models (Hochreiter and Schmidhuber, 1997; Cho et al., 2014). The attention mechanism allows incorporating information from the input sequence words into the one it is currently being processed, thus providing "context" to the word from the rest of the sequence. Language models like Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) leverage this architecture, making up a large part of the modern NLP landscape by providing

a powerful off-the-shelf way to create state-of-the-art models for a wide range of tasks. The Transformer flexibility, allied to its reduced training times and improved ability to learn long-range relationships between terms in a sequence, make it one of the pillars of modern NLP research, and we intend to apply this architecture in our work.

This paper explores Transformer-based models for representing documents as semantic vectors. These vectorial representations are commonly denominated as embeddings, and we intend to use them in a CI system as a mechanism for extracting information from environmental data. Furthermore, the system facilitates Information Encountering by incorporating *searching* and *browsing* mechanisms that leverage the document embeddings. We have named the proposed system MapIntel which derives from (Competitive) Intelligence Map.

The remainder of the paper is structured as follows: after briefly discussing the state-of-the-art and similar systems, we give an overview of the MapIntel architecture and functionalities. Then, we look into the evaluation of the system, describing the experimental setup and discussing the corresponding results. In the "Case Study" section we present an initial prototype of the system and discuss how it can be applied in a real-world situation.

2 Related Work

The process of extracting business-related information for anticipating risks and opportunities is an important task for many companies, yet analysts are overwhelmed with extensive amounts of unstructured data. To support CI analysts, we propose an NLP system for exploring and gathering intelligence from large collections of textual data. To situate our contribution, we review existing work on similar systems applied in CI as well as in other domains, in this section.

Early work on visualizing and interpreting large collections of documents can be found in Kaski et al. (1998), where WEBSOM - a system that organizes a document collection using Self Organizing Maps (SOM) (Kohonen, 1982), mapping each document into the node of a two-dimensional grid that best represents it, thus providing a reliable and visual representation of the collection - is presented. An improved version of the system is given in Kohonen (2013), where users can perform queries using either a set of keywords or a descriptive sentence, a zooming feature to explore specific regions of the map with finer detail is provided and, when selecting a particular node in the map, the titles of the corresponding documents are displayed. Lafia et al. (2019) also used SOM together with Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to convey the relatedness of research themes in a multidisciplinary university library. LDA topic modeling helps in organizing the corpus into different topics described by their most prevalent words. SOM produces a landscape for exploring the topic space and provides users with an overview of the document collection and the ability to navigate, discover items of interest, change the level of detail, select individual documents and discover relationships between them. Similarly, we intend to use more recent

topic modeling and visualization methods to segment the documents into groups of consistent thematics and visualize them in a two-dimensional plane.

Arguably, the closest method to ours in terms of domain application is Dey et al. (2011). They formulated a system for acquiring competitive intelligence from different web resources, including social media, using a wide array of text mining techniques. They also showed how the system can be integrated with the business data and adopted for future decision-making. Their goal is to help the analyst in the task of reading, extracting information, and organizing the data. The system is composed of four distinct modules: *content acquisition and assimilation* gather and extract relevant content from an array of sources, *data pre-processing* is responsible for cleaning and extracting relevant content from different sources, *content processing* identifies and tags the relevant content from the vast collection, and *alerting* notifies the user on predefined triggers such as a competitor's product launch. The paper presents an approach for labeling news articles according to CI-related topics by applying LDA clustering and extracting entities and relations within each cluster, which are then used to define the respective labels. Despite sharing the same goal, we differentiate in the methods we use, namely on the *browsing* aspect of our work, allowing the analyst to interactively encounter relevant information.

(Lafia et al., 2021) proposed a method for modeling and mapping topics from bibliometric data and built a web application accordingly. The produced map allows users to read a body of research "at a distance" while providing multiple levels of detail of the documents' topics. They also incorporated a time dimension, allowing users to understand the evolution of the topics over time. They applied Non-negative Matrix Factorization (Lee and Seung, 1999) to discover the underlying topics in the data and obtain vectorial representations of the documents, and they employ t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton, 2008) for visualizing the documents, resulting in a two-dimensional representation of the corpus. To allow for different detail levels, the authors produced two maps: a coarse map of 9 topics that gives a general overview of the topics within the data and a detailed map of 36 topics that captures more specific research themes. The web application consists of an interactive dashboard that allows users to explore the map of documents and easily extract information. (Lafia et al., 2021)'s work is an example of how document embeddings can be used together with topic modeling and dimensionality reduction methods to provide an interactive and visual way to explore a corpus of documents. We complement this architecture by integrating a *searching* module that allows the user for a more direct way of finding information while utilizing the same embeddings used in the map.

We based our *searching* module on the Vector Space Model (VSM) (Schütze et al., 2008, p. 120-126), a common framework in Information Retrieval, consisting of representing a set of documents as vectors in a vector space while also allowing full-text queries to be represented in the same space. The model then ranks each document in decreasing order of their similarity with the query. The fundamental assumption of the model is that similar documents will be placed

close together in the vector space, whereas dissimilar documents will be far away. An application of VSM for medical imaging can be found in (Sampathila et al., 2020). They proposed a methodology for Content-based Medical Image Retrieval where Magnetic Resonance Imaging (MRI) images are represented using features such as color, shape, and texture, and the k items with the smallest euclidean distance to a given query image are retrieved. They reported that by using this approach they could classify dementia-affected MRI images with an average precision of 97.5% and a maximum recall of 95%, making it an effective way to diagnose new images. Esteva et al. (2020) presented a different application of VSM for querying COVID-19 literature. They proposed Co-Search, an Information Retrieval system that combines semantic search, question answering, and abstractive summarization. The system uses Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), a Transformer-based model for representing documents as semantic vectors, combining it with approximate nearest neighbors and cosine similarity to return the relevant results for a query. Our approach for querying documents is similar to the one used in this work, however we also focus on the visual aspect of the embeddings, leveraging them to map the documents and find the topics that best describe them.

A more recent work focusing on the frontier between Computer Graphics and Machine Learning is *Cartolabe* (Caillou et al., 2021). *Cartolabe* is a web-based, scalable and efficient system for visualizing and exploring large textual corpora, relying on topic modeling algorithms like LDA (Blei et al., 2003) and LSA (Deerwester et al., 1990) to represent documents as vectors of topics and on UMAP (McInnes et al., 2020) to produce a two-dimensional plane that preserves the original topology and neighborhood of the documents. Additionally, they provided an interactive high-level visualization that allows exploration of the corpus in real-time by offloading most of the computations to the data pre-processing pipeline making the system highly scalable to large collections of documents. We intend to apply the same idea of performing the pre-processing offline to improve the system's responsiveness and user interaction. Contrarily to *Cartolabe*, we aim to explore Transformer-based embeddings instead of topic vectors due to the novelty aspect of this architecture and the improved results it has shown in multiple benchmarks in other NLP subdomains.

3 Methods

We propose MapIntel - Figure 1, a system that supports exploring a document collection while promoting serendipity and satisfying emerging information needs by allowing full-text queries over the entire collection. The system is scalable to large amounts of data, is dynamic as it regularly integrates new data, and is fast. It is composed of three main pipelines: Indexing, Query, and Visualization whose objectives are to get documents and their metadata from a source to a database, retrieve the most relevant results to a user query, and produce an interactive interface for exploring the document collection, respectively.

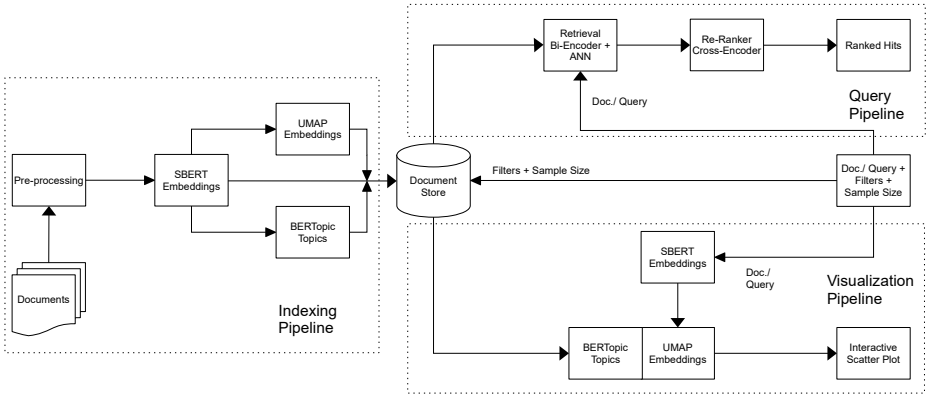

Fig. 1: **MapIntel architecture**. Composed of three main pipelines: Indexing, Query, and Visualization

3.1 Indexing

In this work we decided to focus on how NLP, particularly sentence embeddings, could help organize, explore, and retrieve text documents in the CI domain. Thus, we have not developed the precedent tasks of data collection and pre-processing as much as we believe they are independent of the system and can be easily integrated with it. Nevertheless, it is essential to point out that the system's quality is extremely reliant on these steps, as if we feed it non-ideal data, we will get non-ideal results. In other words, the underlying documents should be complete and diverse such that the analysts can find the necessary information to achieve their task.

Once new documents are fed to the system, their respective embeddings are computed. This process is the basis of our work as it allows the encoding of the semantic identity of the document onto a vector of a given dimensionality. This semantic identity describes the subject of the document, and can be used to compare documents between each other *i.e.*, documents with the same subject will be close in the semantic space and vice-versa. We used SBERT (Reimers and Gurevych, 2019), a derivative of the Transformer-based BERT model, to embed the documents using a pre-trained encoder trained on reducing the distance between queries and relevant results in the MS MARCO dataset (Bajaj et al., 2018). This step takes as input the new documents' text and produces the respective SBERT vectors of 768 dimensions, which we then pass to the Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2020) algorithm to obtain a two-dimensional vector. The goal of UMAP is to minimize the cross-entropy between the high and low dimensional topological representations, resulting in a two-dimensional space that closely imitates the 768-dimensional one. This aspect is another crucial component of MapIntel as it organizes and positions the entire corpus, allowing for a visual exploration and interaction with the data. We opted to use UMAP over other dimensionality reduction techniques

because of its improved map quality, reduction in time required to produce the output map, support for larger data set sizes, and, most importantly, its ability to update the output map with new data without having to rebuild it (McInnes et al., 2020).

We also applied a topic modeling technique called BERTopic (Grootendorst, 2020), based on the work of Angelov (2020). Topic modeling unveils the latent semantic structure of the data and unlike some of the classical techniques such as Latent Dirichlet Allocation (Blei et al., 2003) and Probabilistic Latent Semantic Indexing (Hofmann, 1999), BERTopic leverages the SBERT embeddings and their capacity to encode the semantic attributes of a document to find the most representative topics of a corpus. BERTopic clusters the documents using Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) (McInnes et al., 2017) to find the densest areas of the semantic space while identifying outliers. To overcome the sparsity of the high-dimensional space and the obstacles it creates in finding dense clusters, UMAP is used to reduce the embeddings to a lower dimension (five dimensions by default) prior to the clustering stage, this UMAP is separate from the one we use to visualize the corpus on two dimensions. The primary assumption behind BERTopic is that each dense area in the semantic space is generated by a latent topic shared among the documents that comprise it. Finally, a class-based variant of TF-IDF (Jones, 1972) (c-TF-IDF) is used to extract an importance value of each word for each cluster, which can be used to represent each topic as the set of its most important words. Another advantage of BERTopic over the classical approaches is that we can reduce the number of topics obtained by iteratively comparing the c-TF-IDF vectors, merging the least common topic with its most similar topic, and re-calculate the c-TF-IDF vectors, giving us the option to choose the number of topics.

Finally, we loaded the documents including their metadata, SBERT embeddings, UMAP embeddings, and topics into a database. We used Open Distro for Elasticsearch[1] — an open-source, RESTful, distributed search and analytics engine based on Apache Lucene[2] — to store the data, organize it in an index, and perform full-text search on it. We can think of the described approach as an Indexing Pipeline — Figure 2 — that extracts new raw documents from a data source, pre-processes and manipulates them, stores the results in a database, and indexes the documents for future search tasks.

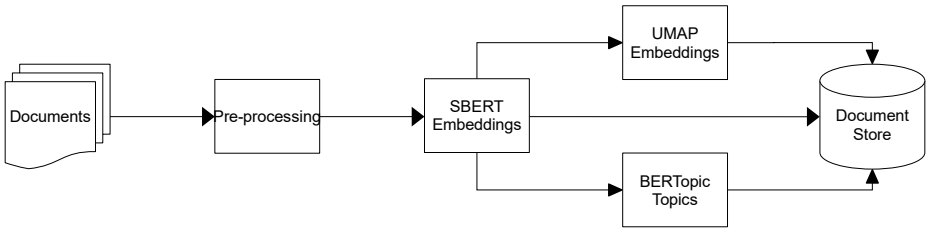

Fig. 2: **Indexing pipeline**. Documents flow through a pre-processing step, followed by an inference step where their SBERT embeddings, UMAP embeddings and BERTopic topics are computed and indexed into a Database for future search tasks.

3.2 Query

Finding meaningful information within a large amount of data is a sizable part of the CI task. The ability to retrieve relevant documents from a large collection of news articles through natural language queries empowers the CI analyst with an easy and intuitive interface to scan the environment.

MapIntel provides a search functionality based on Open Distro for Elasticsearch and its k-Nearest Neighbor (kNN) Search module. By utilizing the kNN module, we can leverage the SBERT embeddings by projecting the query string onto the same semantic space as the corpus and computing its k-nearest neighbors *i.e.* finding the k documents whose embedding vectors are closest to the query embedding vector, according to some predefined similarity metric. Since the embedding vectors encode the semantic identity of each document, this method provides semantically relevant results for a given query. Furthermore, the kNN module delivers a highly performant and scalable similarity search engine by leveraging Elasticsearch's distributed architecture and by implementing Approximate Nearest Neighbors (ANN) search based on Hierarchical Navigable Small World Graphs (Malkov and Yashunin, 2018). The kNN module can also be combined with binary filters that help the user obtain focused results based on characteristics of the documents such as publication date and topic. These filters are applied directly to the database, reducing the search space as a result and improving the subsequent search time.

Once again, we can think of the search functionality as a pipeline, illustrated in Figure 3, where we feed a query string and some binary filters, and obtain documents ordered by their relevance to the query. We employ a Retrieve and Re-rank pipeline based on the works of Nogueira and Cho (2020); Kratzwald et al. (2019) composed by a "Retrieval Bi-Encoder + ANN" node that performs a semantic search using Elasticsearch's kNN module as described above, and by a "Re-Ranker Cross-Encoder" node consisting of a BERT model fine-tuned on the MS MARCO dataset that receives a document and query pair as input and predicts the probability of the document being relevant to the query.

The pipeline works by taking advantage of the characteristics of both nodes. The Bi-Encoder, together with ANN search, can retrieve fairly relevant candidates while dealing efficiently with a large collection of records. The Cross-Encoder is not as efficient since it has to be performed independently for each document, given a query. However, since attention is performed across the query and the document, the performance is higher in the second node (Humeau et al., 2019). Therefore, we combined both nodes by retrieving a large set of candidates from the entire collection using the Bi-Encoder, and filtering the most relevant candidates with the Cross-Encoder while removing noisy results.

With this pipeline, we can provide relevant documents to the user given a query and binary filters while ranking them according to a relevancy score. The pipeline is efficient and uses the SBERT embeddings and the Elasticsearch architecture. As an additional feature, we can input a document instead of a query, allowing us to search for semantically similar documents within the collection.

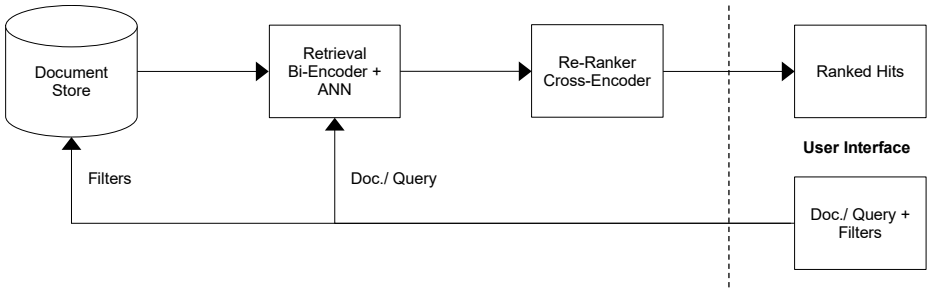

Fig. 3: **Query pipeline**. Given a user query or selected document, its SBERT embedding is inferred and the closest neighbors are computed while a set of binary filters is used to reduce the search space. Following, a Cross-Encoder BERT model is used to re-score the neighbors from the previous step, and they are retrieved together with their similarity score.

3.3 Visualization

We developed a visual interface that organizes and displays the documents to facilitate the environment scanning task, giving the user the ability to browse the data and zoom on particular regions of the semantic space. The interface uses the UMAP algorithm to reduce the dimensionality of the original semantic space to a two-dimensional representation that reliably preserves the original topology.

The methodology employed to produce the interface is described in Figure 4. It begins by taking the same inputs passed to the Query pipeline: a query, and a set of filters. The common inputs create a connection between the two modules — when the user queries the database, the query text is projected onto the two-dimensional map and the filters define which documents are displayed in the

map. In this way, the map can be seen as a graphical extension of the searching mechanism, where the relevant results reside in the neighborhood of the query, giving the user some insight into how the results are obtained. In addition to the common inputs, we require a relative sample size that defines the percentage of randomly chosen documents (after applying the filters) to be displayed in the map. This step is necessary as interaction with the map is hindered by a large number of data points, resulting in a slow and unresponsive experience. Notice that the sample size does not affect the query results, as the search is always performed on the entire collection.

The filters and sample size are used to select the documents to be displayed from the database to produce the interactive scatter plot. We compute the SBERT embedding of the query, followed by its UMAP embedding, thus being able to locate the query in the same space as the documents. An advantage of these two models is that we can efficiently produce embeddings of new text without having to re-train them, making this process quite fast. Once we have the UMAP embedding of the query, we join it with the pre-computed embeddings of the documents from the Indexing pipeline, and we produce the interactive map.

The map provides a means to explore the documents and the different semantic cohorts present within the collection. We color-code the points with the documents' topics identified in the Indexing stage, allowing us to visualize the latent semantic structure of the data, and when hovered, the points display their corresponding title and content attributes.

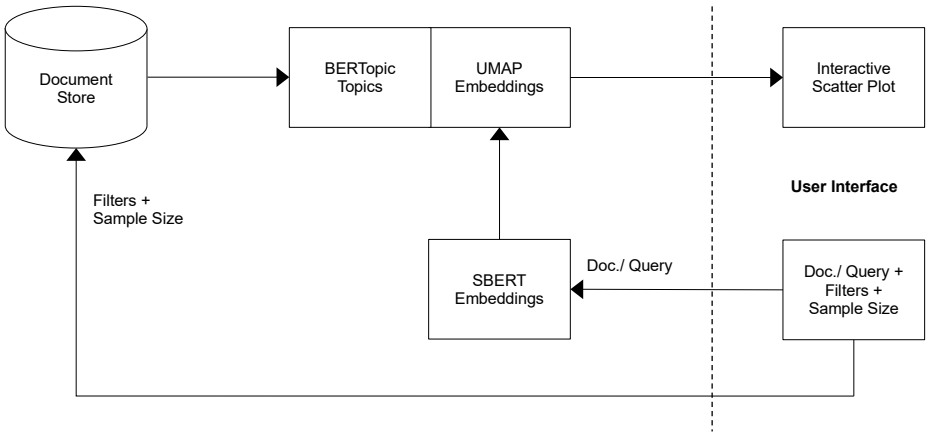

Fig. 4: **Visualization pipeline**. Given a user query or selected document, its SBERT and UMAP embedding are inferred while a set of binary filters and a sample size is used to reduce the number of documents retrieved. Finally, the selected documents' topics and two-dimensional embeddings, together with the query embedding, are displayed in an interactive scatter plot.

4 Evaluation

Our methodology addresses the issues of information dispersion and overload impacting the CI analysts' tasks. The proposed system provides searching and browsing capabilities, contributing to an easier understanding of the business environment by supporting analysts in seeking specific information while promoting undirected information encountering. In this section, we elaborate our choices in the design of the MapIntel system with the results of our experiments and analyze the different components of the system individually.

4.1 Experimental Setup

We evaluate our system quantitatively using the 20 newsgroups (Pedregosa et al., 2011) dataset and the document labels provided. This dataset consists of around 18,000 newsgroups posts on 20 topics divided into six main groups: "Computer," "Recreation," "Science," "Miscellaneous," "Politics," and "Religion." We opted to use this dataset because of the presence of labels that describe the semantic meaning of each document, allowing us to have a reference which we can compare the identified topics with.

Given the inherent difficulty of evaluating the system in its entirety, we decided to deal with each component separately, however since every component of our system depends on the vector representation of the documents, we cannot guarantee an orthogonal evaluation of the components. We focus our experiments in comparing two of the main components of the MapIntel system: the Topic Model and the Sentence Embedding. The following algorithms are compared in our paper:

- **Sentence embeddings**
  1. **Paragraph Vector Model (or Doc2Vec)** (Le and Mikolov, 2014): an unsupervised algorithm that learns both word and document vectors by minimizing the error of predicting the next word in a paragraph (a variable-length piece of text) given the paragraph and previous word vectors;
  2. **SBERT**: a derivative of the Transformer-based BERT model, to embed the documents using a pre-trained encoder trained on reducing the distance between queries and relevant results in the MS MARCO dataset;
- **Topic modeling**
  1. **LDA**: a generative probabilistic model of a corpus in which each document is modeled as a finite mixture over an underlying set of topics, and each topic is characterized by a distribution over words;
  2. **BERTopic**: a cluster-based topic model that unveils the latent document-topic distribution responsible for the existing groups of documents in a semantic vector space;
  3. **Contextualized Topic Model (CTM)** (Bianchi et al., 2021): combines contextualized representations (like SBERT) with neural topic models resulting in more meaningful and coherent topics;

We use three main metrics to guide our model comparison:

- **kNN classifier accuracy for the UMAP projections**: evaluate the quality of the two-dimensional projections by inferring their labels using a kNN classifier and computing its Accuracy for multiple values of k (McInnes et al., 2020). We present the average Accuracy over the range of k values we tried: 10, 20, 40, 80, 160.
- **Normalized Mutual Information (NMI) for the topic assignments**: information theoretic based measure that tells us how much knowing about the topics of the documents reduces our uncertainty about the original labels and vice-versa (Vinh et al., 2009). The NMI ranges between 0 and 1 where the former corresponds to no mutual information, and the latter indicates perfect correlation. The higher the value of this metric, the better we can capture the true topical nature of the documents, reflected by their labels, with the assigned topics.
- **Topic coherence**: measure the quality of the words that describe each topic by applying the C_v metric (Röder et al., 2015) indicating whether the words that compose a given topic support each other. This metric is shown to be correlated the most, from a set of existing metrics, with human ratings on understandability of the topic-describing words, and it ranges between -1 and 1, where the former corresponds to an incoherent topic description, and the latter indicates a coherent one.

We performed hyperparameter tuning using a multi-objective approach to optimize the three metrics specified previously. We used the Tree-structured Parzen Estimator (TPE) algorithm (Bergstra et al., 2011; Ozaki et al., 2020) for sampling the hyperparameter space at each trial t of the optimization process. Contrarily to random search, TPE samples values x_t^α for each hyperparameter α by fitting one Gaussian Mixture Model (GMM) $l(x^\alpha)$ to the set of values associated with the best objective scores in past observed trials, and another GMM $g(x^\alpha)$ to the remaining values. It then chooses the hyperparameter value x_t^α drawn from $l(x^\alpha)$ that maximizes the ratio $\frac{l(x^\alpha)}{g(x^\alpha)}$. For each trial, we evaluated the sampled hyperparameters using five fold cross-validation approach where the folds preserve the percentage of samples of each class. In total, 100 trials were evaluated.

4.2 Results

Our results based on the setup described above are shown in Table 1. For each trial, we report the average results and standard deviations over the cross-validation folds. The table contains the best trials for each of the Topic/Embedding model combinations according to the average of the three objective metrics, which we applied MinMax scaling to avoid any impact of the metrics scale on the choice of the best model. We can see that the combination that uses BERTopic and SBERT outperform the others with respect to both NMI and C_v while having a within standard deviation kNN Classifier Accuracy to the best value. Another interesting observation is that combinations using SBERT have generally

better results. To facilitate results reproduction efforts, we open-sourced the code developed for the experiments at https://github.com/dfhssilva/mapintel_research.

Topic Model	**Embedding Model**	MinMax Average	NMI	Topic Coherence C_v	kNN Classifier Accuracy
BERTopic	Doc2Vec	0.50	0.11 ± 0.01	0.72 ± 0.02	0.16 ± 0.01
BERTopic	SBERT	**0.94**	$\mathbf{0.36 \pm 0.03}$	$\mathbf{0.76 \pm 0.08}$	0.36 ± 0.01
CTM	Doc2Vec	0.56	0.23 ± 0.02	0.55 ± 0.02	0.24 ± 0.03
CTM	SBERT	0.70	0.33 ± 0.02	0.58 ± 0.02	0.28 ± 0.04
LDA	Doc2Vec	0.58	0.25 ± 0.03	0.53 ± 0.03	0.25 ± 0.01
LDA	SBERT	0.71	0.26 ± 0.03	0.53 ± 0.03	$\mathbf{0.37 \pm 0.05}$

Table 1: Hyperparameter tuning best trials per topic and embedding model according to the MinMax average of the multiple objective metrics.

Despite having validated our system design decisions, we additionally discuss some possible shortcomings and limitations of the algorithms that we use in the system. SBERT is a pre-trained model and thus can be fine-tuned to possibily achieve even better results in a specific domain (Thakur et al., 2021). In our experiments, we used the original SBERT bi-encoder so this could be something to test in the future, as it is likely to improve the overall system. Regarding topic modeling, BERTopic assumes that each document only contains a single topic, an assumption difficult to justify most of the times as a document can inherently have multiple topics. In addition, another shortcoming of BERTopic are the topic descriptions. Even though, it uses a contextual representation of the documents to find out the topics they belong to, the words that constitute a given topic are just a reflection of the most important words of that group, which can lead to redundant words in the topic label. This problem is already discussed in Grootendorst (2022) and we want to explore some of the proposed solutions in future versions of the system. Finally, UMAP is used to reduce the high dimensionality of the SBERT embeddings to merely two dimensions. This is done in a way that preserves the distance between points and where the dimensions produced don't have any particular meaning. Even though interpretability of the axes in the visual map is not a big concern in our use case, we believe this is a question that might come up from the user and thus we think it is important to clarify this detail.

Furthermore, we present the UMAP two-dimensional maps of the documents in the 20 newsgroups dataset. Figure 5 shows the comparison between the distribution of the original labels and the topics assigned by the best performing model according to the MinMax average score for the train data (see Table 2 for exact hyperparameter values). Likewise, Figure 6 shows the same comparison for the test data and demonstrates the ability of the model to generalize to unseen samples. We can see that the identified topical cohorts are mostly matching with the original groups, indicating that the embeddings have learned the original labels in a fully unsupervised way.

Model	Hyperparameter	Value
-	embedding_model	sentence-transformers/msmarco-distilbert-base-v4
-	topic_model	BERTopic
HDBSCAN	cluster_selection_epsilon	0.0551652
HDBSCAN	cluster_selection_method	leaf
HDBSCAN	min_cluster_size	110
BERTopic	min_topic_size	33
UMAP	metric	cosine
UMAP	n_components	10
UMAP	n_neighbors	14

Table 2: Hyperparameter values with the highest MinMax average of NMI, Topic Coherence C_v, and kNN Classifier Accuracy for training data according to Table 1.

Additionally, it is possible to see that semantically similar topics are located close to each other in the map. This is the case of all the computation related topics such as *window.server.windows.motif.display* and *format.files.graphics.file.gif*. Finally, there is also an agreement between the topic meaning given by the top 5 words describing the topics and the original label description. For example, the same points that have the label *sci.space* also have the topic *space.launch.nasa.orbit.shuttle*. Figure 9 in the appendix shows how much correspondence there is both in terms of labels and positioning for technological-related documents - we can see how the distribution of the original label *comp.graphics* matches the distribution of the topic *format.files.graphics.file.gif*, likewise *comp.windows.x* matches *window.server.windows.motif.display*, and *sci.electronics* matches *amp.radio.powed.voltage.use*, whereas the remaining original labels are lost, giving birth to new, and more semantically coherent, groups of documents (note how the distribution of the original label *comp.sys.ibm.pc.hardware* is very dispersed, even overlapping with other groups and how the distribution of the topics occupying the same region is very clear and compact).

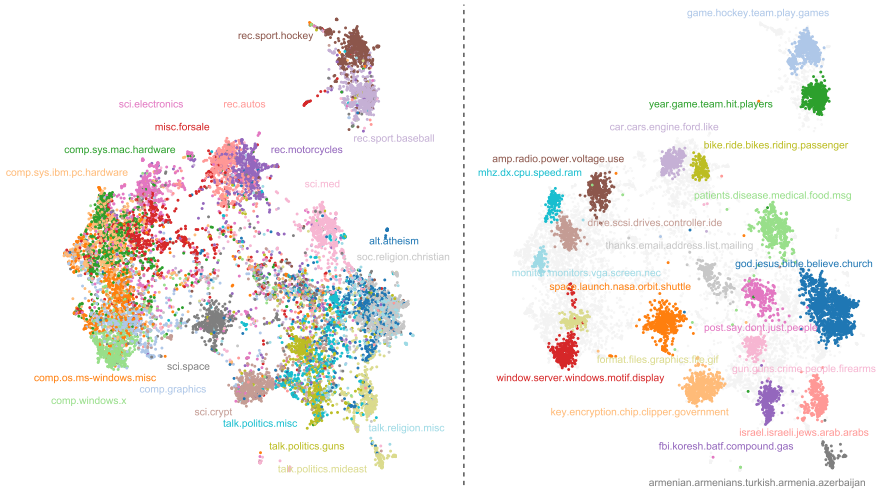

Fig. 5: Comparison between UMAP planes of **train data** with original (left) and topic labels (right).

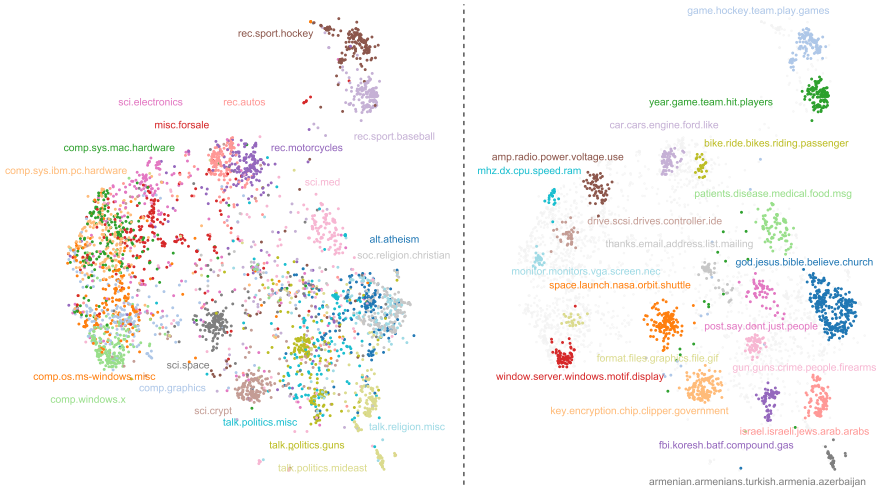

Fig. 6: Comparison between UMAP planes of **test data** with original (left) and topic labels (right).

An important characteristic of BERTopic is that it is able to identify noise, leading to a topic assignment where a portion of the observation are classified as outliers. This produces a cleaner map to explore the documents at the loss of samples that are not given a topic. In Figure 5 (right) the percentage of documents classified into the aforementioned category is 51.4% - these are the light grey points scattered across the map.

5 Case Study

This use case is based on a real-world example, related to the recurrent tasks that the intelligence analysts at AICEP - Portuguese Trade & Investment Agency have to perform. AICEP "is a government business entity, focused in encouraging the best competitive foreign companies to invest in Portugal and in contributing to the success of Portuguese companies abroad in their internationalization processes or export activities." Given this mandate AICEP needs to monitor the world news and make sure it is updated with world events. Additionally, analysts at AICEP are called upon to produce reports on specific markets and industries, in order to guide the Portuguese state's diplomatic efforts and private company investments. These reports are pre-formatted, however the use and integration of text data remains a challenge. This is unfortunate as recent news and text documents allow a more detailed understanding of specific problems and nuances that are difficult to capture in an alphanumeric data table. The objective of this use case is to show how MapIntel can be used to mitigate this limitation and allow the inclusion of relevant text data in the report.

In this case study, we will show the most important tools that the MapIntel system offers to explore and select relevant documents to include in a market or industry report. These tools allow the analyst to screen the news and select those that are relevant to include in the report. This process cannot be strictly seen as an information retrieval process, in fact it is much more an exploration process, in which the analyst's criterion plays a crucial role. The process usually starts with a natural language query that the analyst defines based on his/her knowledge of the market or industry. The query is used by the system as the seed in the process of exploring the corpora. The query is projected onto the two-dimensional surface produced by MapIntel, and the neighbor documents are defined in the semantic space, thus generating a number of candidate results relevant for the report. From this starting position the analyst can interact visually with the results and the entire corpus, and guide his/her interest with successive queries, focusing in the most interesting or promising instances to include in the report.

To evaluate the MapIntel system with real data, we gathered news articles from multiple international sources with the use of an API[3]. As already stated, there are multiple sources of CI, and different information can be obtained from these. Dey et al. (2011) shows in Table 3 what kind of information can be acquired from these sources, particularly the ones that are easily available through the web. We decided to work mainly with news articles as they provide a gen-

eral and accessible means of information about the environment, however, our methodology is easily extensible to data from different sources.

Type of Competitive Intelligence	**Web Resources**
People event	News, company web-sites
Competitor strategies, Technology investment, etc.	News, Discussion forum, Blogs, Patent search sites
Consumer sentiments	Review sites, Social networking sites
Promotional events and pricing	Social networking sites
Related real-world events	News, Social networking sites

Table 3: Competitive intelligence resources on the web (Dey et al., 2011).

The API employed retrieves news articles and their metadata, including attributes such as source, author, title, description, content, category, URL, and publication date and time. We used this API to feed the system with updated data on a schedule while focusing on articles written in English from a set of predefined categories (business, entertainment, general, health, science, sports, technology). The system was fed with a total of 334,925 articles during the period between October 2020 and June 2021.

Due to API limitations, the retrieved data has its content truncated to 200 characters. To overcome this, we treat a single document unit as the concatenation of title, description, and content, providing us a semantically loaded piece of text that we can use for testing the system. Despite this limitation, we give the user the possibility of accessing the full article through its URL. We ensure that each document is unique, is written in English, and doesn't have any HTML tags or any strange pattern.

Once the data is cleaned, it follows the Indexing pipeline - Figure 2 -, so it can be used downstream by the Query pipeline - Figure 3 - and the Visualization pipeline - Figure 4.

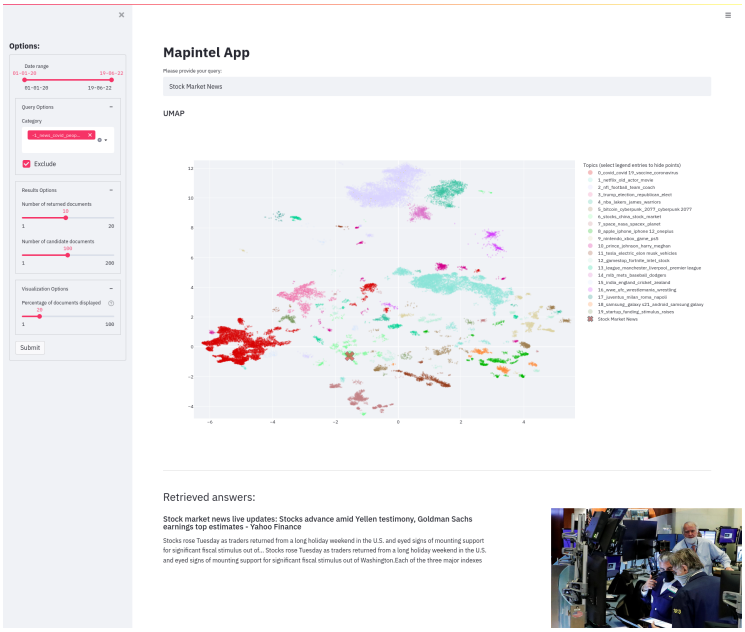

Fig. 7: **MapIntel webapp interface**. Composed of four main components: search box, interactive scatter plot, parameter sidebar, and list of results.

We also built a simple web application in Python based on the MapIntel system. The web application is containerized and composed of three main modules: an Elasticsearch instance container that stores and indexes the documents, their metadata, their topics and their embeddings, a container responsible for all the computations involved in the three pipelines of the system that communicates with the user interface through a FastAPI[4] server providing the necessary endpoints, and a third container consisting of a Streamlit[5] application that interacts with the data through HTTP requests. For transparency, the code used for the web application is made public at github.com/NOVA-IMS-Innovation-and-Analytics-Lab/mapintel_project.

We highlight the user interface of the app in Figure 7. It is composed of four main components: a **search box** at the top of the page that the user can use to write any natural language query, an **interactive scatter plot** that shows the UMAP projection of the document embeddings and allows the exploration of the corpus, a **sidebar** at the left where the user can specify any additional parameter or binary filter to the query and map, and a **list of results** at the bottom that contains the documents with the highest scoring for the specified query.

The app enables information searching by receiving a query through the search box together with the filters and parameters selected in the sidebar. This allows the CI analyst to write queries in Natural Language while having a finer control of the results through sidebar parameters like including or excluding

specific topics and specifying a date range. We show in Table 4 some query examples (without any filters) a CI analyst could make and their respective results. Take, for example, the last query "Myanmar Coup", we should expect to receive documents about the current situation in Myanmar and the ongoing (2021) *Coup d'état*. The top 3 results confirm our expectation.

Query	**Top 3 Results (Title)**	**Score**
"Suez Canal World Trade"	Suez Canal blockage felt across the world as trade comes to a pause.	4.90
	Suez Canal ship partially refloated after huge effort to unblock key global trade route.	4.70
	Egypt 'seizes' container ship over $1bn Suez claim.	4.42
"Indian Elections"	India elections: Modi party defeated in West Bengal battleground - BBC News.	5.36
	At 103, India's first voter casts vote in Himachal panchayat radish polls.	4.77
	Vote count in five Indian states under way as pandemic rages.	4.44
"Oil Prices"	Oil prices near 2-year highs above $70 as investors expect OPEC+ to confirm its supply policies.	7.63
	Oil Prices Rally Towards $70 As Demand Outlook Improves - OilPrice.com.	7.62
	Oil prices to reach $72 by summer: Goldman Sachs - Fox Business.	7.45
"Myanmar Coup"	Myanmar Coup: With Aung San Suu Kyi Detained, Military Takes Control - NPR.	8.03
	Myanmar's military has detained leader Aun San Suu Kyi in a coup. Here's what you need to know - CNN.	8.03
	News24.com — Myanmar military stages coup and declares state of emergency, detains leader Aung San Suu Kyi.	7.94

Table 4: Example queries and their top 3 results and respective relevancy scores. The results are computed according to the methodology described in 3.2.

An additional feature of the app is the ability to browse documents through the scatter plot. This visualization shows the two-dimensional UMAP projection of the 768-dimensional embeddings of the documents. It can be used by the analyst to explore the document collection and the underlying thematic groups. The map can be interacted with by zooming, panning and hovering specific regions and documents, allowing the analyst to see the content of each data point. A representation of the hovering capability and a more detailed image of the document map can be seen in Figure 8. In addition, we integrate the *searching* and *browsing* modules of the system by plotting the query embedding in the UMAP plane - represented with an "X" shaped marker -, allowing a visual

exploration of the relevant results *i.e.* the data points close to the query. Finally, we allow searching of similar documents by providing the option to click on any point in the map and obtain their most similar documents in the list of results below. We found that this feature enhances exploration and finding of relevant documents.

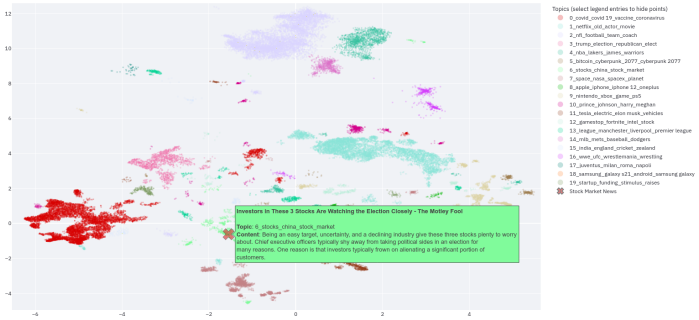

Fig. 8: **UMAP projection of the corpus**. The documents' SBERT embeddings are projected on two dimensions with UMAP and the data points are colored according to their assigned topic. The map can be used interactively by zooming, panning, hovering, and selecting points. On hovering, the document text and some metadata is displayed and on selection the document is queried against the rest of the corpus. The query is marked with an "X" in the map.

Besides the searching and browsing functionality, the system can also organize the documents into semantically similar cohorts and automatically label them. This is achieved with the help of topic modeling - particularly BERTopic - and the resulting topic labels can be seen in Table 5. The topic labels can be particularly useful to the analyst to understand the different subjects covered by the corpus at a glance since these contain the most discriminant words of each set of documents, giving an idea of the content of the documents without having to read each one individualy. For example, we can infer using the words in topic 20 that the respective documents correspond to articles about new companies and businesses, as well as fundings they might have obtained lately. In addition, by looking at Figure 8 we can notice that these topics represent clusters of documents in the semantic space, which confirms the main assumption that semantically similar documents have similar embeddings.

Topic Number	Topic Words	Topic Coherence
−1	- (outliers)	-
1	covid, covid 19, vaccine, coronavirus	0.644
2	netflix, old, actor, movie	0.223
3	nfl, football, team, coach	0.359
4	trump, election, republican, elect	0.12
5	nba, lakers, james, warriors	1
6	bitcoin, cyberpunk, 2077, cyberpunk 2077	0.336
7	stocks, china, stock, market	0.413
8	space, nasa, spacex, planet	0.314
9	apple, iphone, iphone 12, oneplus	1
10	nintendo, xbox, game, ps5	1
11	prince, johnson, harry, meghan	1
12	tesla, electric, elon musk, vehicles	0.943
13	gamestop, fortnite, intel, stock	0.212
14	league, manchester, liverpool, premier leagu	1
15	mlb, mets, baseball, dodgers	0.356
16	india, england, cricket, zealand	1
17	wwe, ufc, wrestlemania, wrestling	1
18	juventus, milan, roma, napoli	1
19	samsung, galaxy s21, android, samsung galax	1
20	startup, funding, stimulus, raises	0.617

Table 5: Topic labels and respective coherence values. We used the five words with the highest c-TF-IDF score per topic to label them and extracted the coherence value C_v of these words.

The prototype of MapIntel presented can help AICEP analysts' in fabricating the on-demand reports by searching and exploring the corpus of news articles, finding useful information along the way that could benefit the standard market/industry report. While we believe that this prototype can make this task more efficient, we are still limited on some technical aspects of the visual interface such as the display of large amounts of documents, and on some usability capabilities such as detecting logical relationships between documents (e.g. developments of a given news article) or the exploration of documents at multiple levels of detail as in Lafia et al. (2021).

6 Conclusion

In this paper, we presented MapIntel, a new system for extracting knowledge from large corpora of text documents. MapIntel differentiates from previous systems in that it leverages Transformer-based document embeddings to provide efficient, natural language searching of documents. The use of Transformer-based embeddings allows the harness of the semantic attributes of the documents, which can then be explored in a two-dimensional map, produced using UMAP. Additionally, MapIntel also organizes the documents in topical cohorts, providing yet another framework for the interaction of the user with the corpus. The system, rooted in the concept of Information Encountering (Erdelez and Makri,

2020), offers browsing and searching capabilities to facilitate information acquisition and encourage serendipitous discoveries. MapIntel is aimed at supporting Competitive Intelligence analysts, however it can also be applied in different scenarios that require analysis of a large corpus of documents, making it a system suitable for this type of tasks.

We detailed the methodology proposed, having evaluated it through a well-defined experimental setup. Furthermore, we showed how the MapIntel system can be used in a real-world case and what benefits it can provide, we also developed a web application, applying the proposed methodology while enhancing the interaction with the underlying data through an interactive scatter plot. Finally, we developed and open-sourced the code base of the web application and our experiments, so the work can be easily reproducible and further research can be continued.

Future work will involve conducting a more comprehensive evaluation of the system using new corpora, extending the system to other application domains, and incorporating additional relevant functionalities and tools. Some different directions would be to experiment with recent language models, such as newer versions of OpenAI's GPT, namely in the vectorization of documents, in the creation of a chat-like interface, and in the generation of pre-formatted outputs, like the reports from our use case.

Acknowledgments

This work was supported by the *Fundação para a Ciência e Tecnologia of Ministério da Ciência e Tecnologia e Ensino Superior* (research grant under the DSAIPA/DS/0116/2019 project).

Notes

[1] opendistro.github.io/for-elasticsearch
[2] lucene.apache.org
[3] newsapi.org
[4] fastapi.tiangolo.com
[5] streamlit.io

Bibliography

Angelov, D. (2020). Top2Vec: Distributed Representations of Topics. *arXiv:2008.09470 [cs, stat]*.

Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., and Wang, T. (2018). MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv:1611.09268 [cs]*.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, pages 2546–2554, Red Hook, NY, USA. Curran Associates Inc.

Bianchi, F., Terragni, S., and Hovy, D. (2021). Pre-training is a Hot Topic: Contextualized Document Embeddings Improve Topic Coherence. *arXiv:2004.03974 [cs]*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Caillou, P., Renault, J., Fekete, J.-D., Letournel, A.-C., and Sebag, M. (2021). Cartolabe: A Web-Based Scalable Visualization of Large Document Collections. *IEEE Computer Graphics and Applications*, 41(2):76–88.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*.

Dey, L., Haque, S. M., Khurdiya, A., and Shroff, G. (2011). Acquiring competitive intelligence from social media. In *Proceedings of the 2011 Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*, MOCR_AND '11, pages 1–9, New York, NY, USA. Association for Computing Machinery.

Erdelez, S. and Makri, S. (2020). Information encountering re-encountered: A conceptual re-examination of serendipity in the context of information acquisition. *Journal of Documentation*, 76(3):731–751.

Esteva, A., Kale, A., Paulus, R., Hashimoto, K., Yin, W., Radev, D., and Socher, R. (2020). CO-Search: COVID-19 Information Retrieval with Semantic Search, Question Answering, and Abstractive Summarization. *arXiv:2006.09595 [cs]*.

Grootendorst, M. (2020). Bertopic: Leveraging bert and c-tf-idf to create easily interpretable topics.

Grootendorst, M. (2022). Bertopic: Neural topic modeling with a class-based tf-idf procedure.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57.

Humeau, S., Shuster, K., Lachaux, M.-A., and Weston, J. (2019). Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*.

Kaski, S., Honkela, T., Lagus, K., and Kohonen, T. (1998). WEBSOM–self-organizing maps of document collections. *Neurocomputing*, 21(1-3):101–117.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural networks*, 37:52–65.

Kratzwald, B., Eigenmann, A., and Feuerriegel, S. (2019). RankQA: Neural Question Answering with Answer Re-Ranking. *arXiv:1906.03008 [cs]*.

Lafia, S., Kuhn, W., Caylor, K., and Hemphill, L. (2021). Mapping research topics at multiple levels of detail. *Patterns*, 2(3):100210.

Lafia, S., Last, C., and Kuhn, W. (2019). Enabling the Discovery of Thematically Related Research Objects with Systematic Spatializations. In *14th International Conference on Spatial Information Theory (COSIT 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196. PMLR.

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

Madureira, L., Popovič, A., and Castelli, M. (2021). Competitive intelligence: A unified view and modular definition. *Technological Forecasting and Social Change*, 173:121086.

Malkov, Y. A. and Yashunin, D. A. (2018). Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *arXiv:1603.09320 [cs]*.

McInnes, L., Healy, J., and Astels, S. (2017). hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205.

McInnes, L., Healy, J., and Melville, J. (2020). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv:1802.03426 [cs, stat]*.

Nogueira, R. and Cho, K. (2020). Passage Re-ranking with BERT. *arXiv:1901.04085 [cs]*.

Ozaki, Y., Tanigaki, Y., Watanabe, S., and Onishi, M. (2020). Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, pages 533–541, New York, NY, USA. Association for Computing Machinery.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Louppe, G., Prettenhofer, P., Weiss, R., Weiss, R. J., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.*

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs].*

Röder, M., Both, A., and Hinneburg, A. (2015). Exploring the Space of Topic Coherence Measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 399–408, New York, NY, USA. Association for Computing Machinery.

Sampathila, N., Pavithra, and Martis, R. J. (2020). Computational approach for content-based image retrieval of K-similar images from brain MR image database. *Expert Systems*, n/a(n/a):e12652.

Schütze, H., Manning, C. D., and Raghavan, P. (2008). *Introduction to Information Retrieval*, volume 39. Cambridge University Press Cambridge.

Thakur, N., Reimers, N., Daxenberger, J., and Gurevych, I. (2021). Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *arXiv:1706.03762 [cs].*

Vinh, N. X., Epps, J., and Bailey, J. (2009). Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1073–1080, New York, NY, USA. Association for Computing Machinery.

A Zoom on UMAP plot

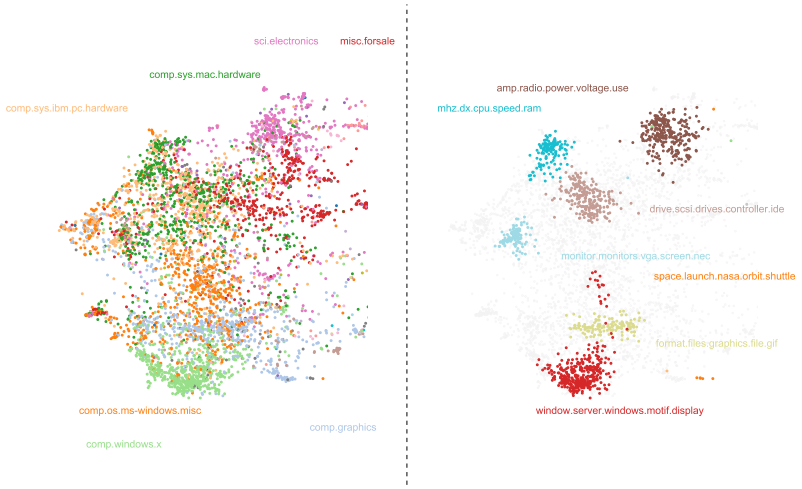

Fig. 9: Zoom on technological region of UMAP. Comparison between UMAP planes of **train data** with original (left) and topic labels (right).