MapIntel: Enhancing Competitive Intelligence Acquisition Through Embeddings and Visual Analytics

David Silva, Fernando Bação

Abstract

Briefly summarize your previous work, goals and objectives, what you have accomplished, and future work. (100 words max) If you have a question, please use the help menu ("?") on the top bar to search for help or ask us a question.

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). According to Brod (1999), "Companies with competitive intelligence programs have better knowledge of their markets, better cross-functional relationships between their business units and a greater ability to develop proactive competitive strategies." CI has a fundamental role in helping businesses remain competitive, influencing a wide range of decision-making areas, and leading to substantial improvements such as the increase of revenue, new products or services, cost savings, time savings, profit increases, and achievement of financial goals (Calof et al., 2017).

Competitive Intelligence analysts are responsible for developing the CI task through a combination of gathering data, processing it, and communicating information. 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. This shift has produced a large volume of data about products, customers, competitors, and any aspect of the business environment that can be used to foresee opportunities and risks. Given the vastness and diversity of this data, it has become a necessity to design tools that can aid analysts in the CI gathering and analysis process. Therefore, the goal is to enhance the analyst's task by providing a tool to explore, organize and visualize the environmental data present in the array of existing sources.

Traditionally, the most important sources of CI have been, respectively, news providers, corporate websites, and trade publications (Marin and Poulter, 2004). With the advent of the internet, new sources, such as social networks Dey et al. (2011), have emerged, while existing ones have become enriched and easily accessible. 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 analyst's task and exacerbate the need for tools to support it.

Various studies have attempted to create systems for exploring and gathering intelligence from large collections of textual data (Dey et al., 2011; Esteva et al., 2020; Lafia et al., 2019, 2021). These studies have consistently applied Natural Language Processing (NLP) techniques for helping users comprehend large volumes of text without requiring to sift through every document. Dey et al. (2011) designs 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 pre-defined triggers.

Although the previously mentioned systems have successfully been used for dealing with large amounts of text, 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 actively seek information, 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.

Another unique characteristic of our work is the use of state-of-the-art NLP techniques. With the recent emergence of the Transformer architecture (Vaswani et al., 2017), significant improvements were made in several NLP subdomains. This new architecture is based solely on the attention mechanism, providing parallelization capabilities and significant improvements in training time. Also, the Transformer can more easily learn long-range relationships between terms in the input sequence than pre-existing architectures (Vaswani et al., 2017). 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 an off-the-shelf, powerful way to create state-of-the-art models for a wide range of tasks.

In this paper, we explore 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 name the proposed system as MapIntel from (Competitive) Intelligence Map. The code developed for this work can be accessed at github.com/NOVA-IMS-Innovation-and-Analytics-Lab/mapintel_project.

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 large 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, in this section, existing work in similar systems applied in CI as well as in other domains.

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 uses SOM together with Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to convey the relatedness of research themes in a multidisciplinary university library. Documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. That said, each document is embedded in a vector space of n dimensions, corresponding to the number of topics selected. 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 documents.

Arguably, the closest system to ours in terms of domain application is Dey et al. (2011). They formulate a system for acquiring competitive intelligence from different types of web resources, including social media, using a wide array of text mining techniques. They also show 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* gathers and extracts 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 pre-defined triggers such as 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. The labeling contributes to the organization of the collection and facilitates the information extraction process.

(Lafia et al., 2021) proposes a method for modeling and mapping topics from bibliometric data and builds a web application based on this method. The map produced allows users to read a body of research "at a distance" while providing multiple levels of detail of the documents' topics. They also incorporate

a time dimension, allowing users to understand the evolution of the topics over time. They apply 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 produce 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.

We base 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 propose 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 report that by using this approach they can 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) presents a different application of VSM for querying COVID-19 literature. They propose 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.

3 MapIntel

We propose MapIntel - Figure 1, a system that supports the exploration of 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 which objectives are respectively, to get documents and their metadata from a source to a database, to retrieve the most relevant results to a user query, and to produce an interactive interface for exploring the document collection.

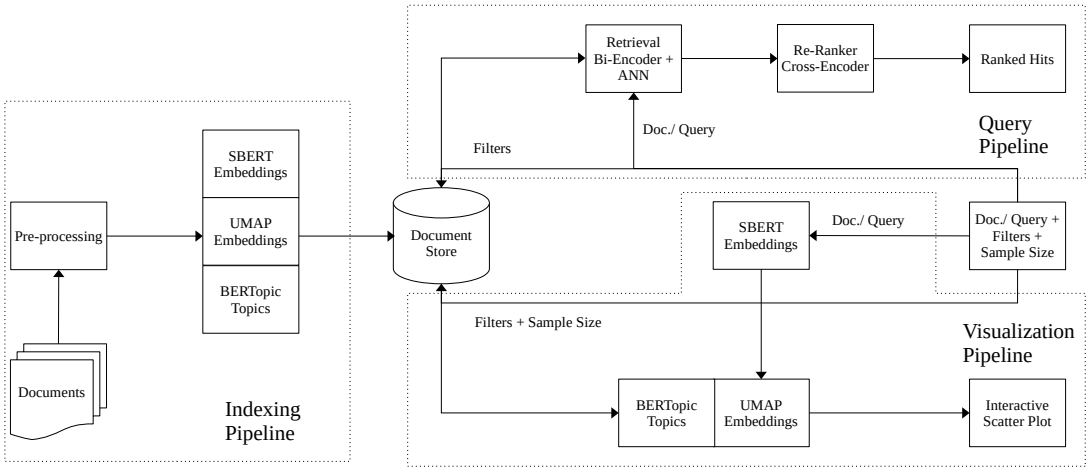


Figure 1: The MapIntel system architecture

3.1 Indexing

In this work we decided to focus on how NLP and particularly sentence embeddings could help in organizing, exploring, and retrieving text documents in the CI domain. Thus, we don't develop as much the precedent tasks of data collection and pre-processing as we believe they are independent of the system

and can be easily integrated with it. Nevertheless, it is important to point out that the quality of the system is extremely reliant on these steps, as if we feed it non-ideal data, we will get non-ideal results. MapIntel is nothing but a set of tools to facilitate the exploration and understanding of a corpus and it will not give useful insights if the data isn't useful itself.

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 what is the subject of the document, and it 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 use 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 produces vectors of 768 dimensions, which we then reduce to 2 dimensions using the Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2020) algorithm. UMAP constructs a topological representation of the high and low dimensional data and its goal is to minimize their cross-entropy, which measures the difference between the two representations, by adjusting the low-dimensional representation. This is another important component of MapIntel as it allows the organization and localization of the entire document collection in a 2-dimensional map, which can be used to explore and interact 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 apply 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 (5 dimensions by default) prior to the clustering stage. The main 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 for each cluster an importance value of each word, 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 load the documents, their metadata, their SBERT embeddings, their UMAP embeddings, and their topics into a database. We use 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 approach described 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.

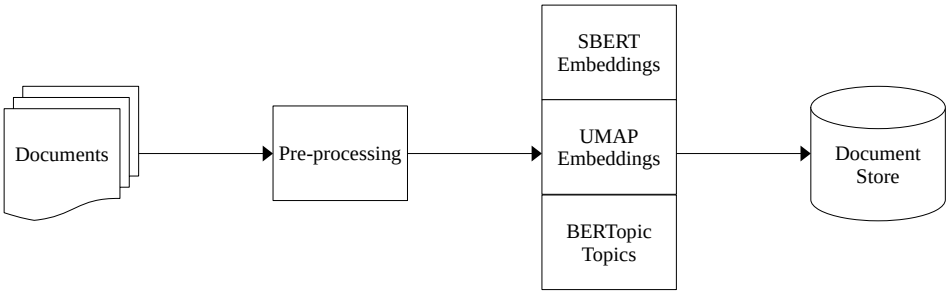


Figure 2: Indexing pipeline

[1] opendistro.github.io/for-elasticsearch
[2] lucene.apache.org

3.2 Query

Finding meaningful information within a large amount of data is a big 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 pre-defined 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 we obtain documents ordered by their relevancy to the query. We employ a Retrieve and Re-rank pipeline based on the work of Nogueira and Cho (2020); Kratzwald et al. (2019) composed by a "Retrieval Bi-Encoder + ANN" node that performs 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 documents. The Cross-Encoder isn't 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 combine both nodes by retrieving a large set of candidates from the entire collection using the Bi-Encoder, and by 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 makes use of 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.

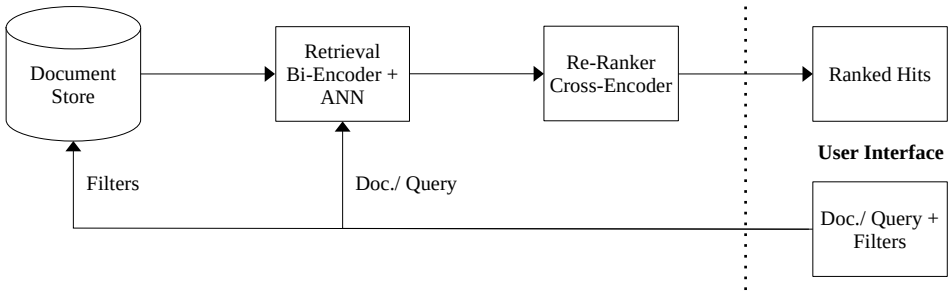


Figure 3: Query pipeline

3.3 Visualization

To facilitate the environment scanning task, we developed a visual interface that organizes and displays the documents, 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 2-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 2-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 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 doesn't affect the query results, as the search is always performed on the entire collection.

To produce the interactive scatter plot, the filters and sample size are used to select the documents to be displayed from the database. 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.

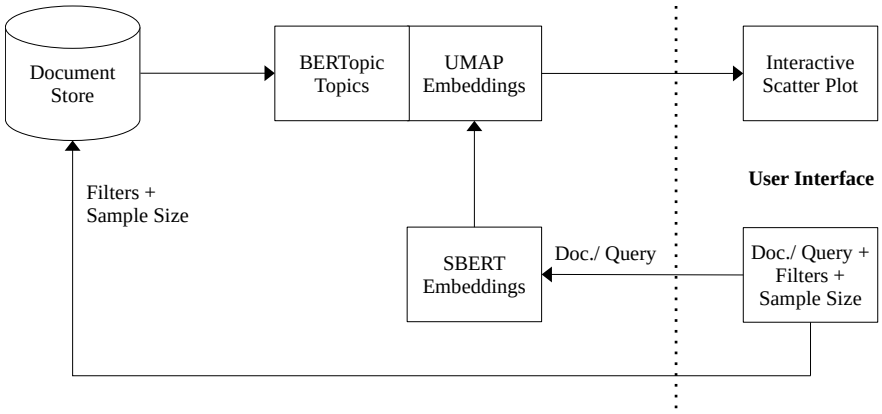


Figure 4: Visualization pipeline

4 Evaluation

Our methodology addresses the issues of information dispersion and overload impacting the CI analysts' task. 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 analyse 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 6 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 on 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 2 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. 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**: evaluate the identification of the topical nature of the documents by computing the NMI between the original labels and the assigned topics. 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 original labels and the main topics in the corpus.

- **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 with human ratings on understandability of topic descriptions, given as word sets, and it ranges between -1 and 1, where the former corresponds to incoherent topic descriptions, and the latter indicates a coherent topic description.

We don't set the hyperparameters of the models we compare a priori as we believe they can have a significant impact on the evaluation. For that reason, we perform hyperparameter tuning using a multi-objective approach to optimize the three metrics specified previously. We use 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 evaluate the sampled hyperparameters using a 5-fold cross-validation approach where the folds preserve the percentage of samples of each class. In total, 100 trials were evaluated.

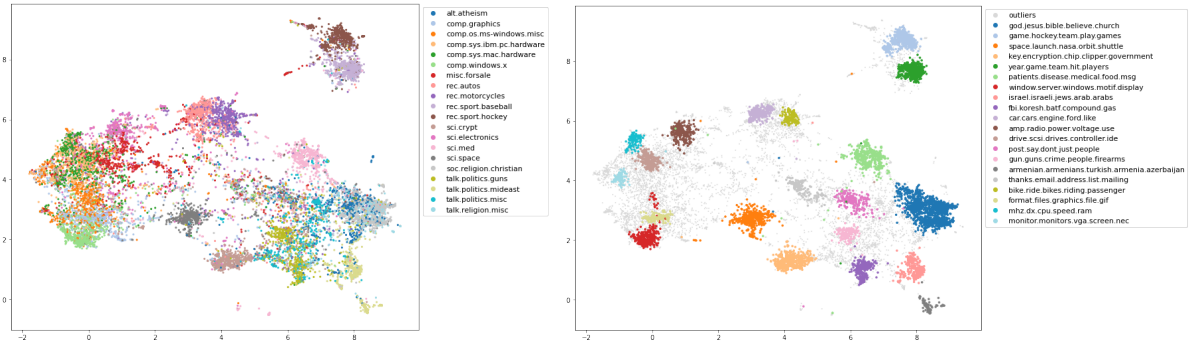
4.2 Results

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 MinMax average of the three objective metrics. 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.

Topic Model	**Embedding Model**	MinMaxAverage	NMI	TopicCoherenceC_v	kNNClassifierAccuracy
BERTopic	Doc2Vec	0.499	0.105 ± 0.010	0.721 ± 0.024	0.157 ± 0.013
BERTopic	SBERT	**0.942**	$\mathbf{0.363 \pm 0.033}$	$\mathbf{0.759 \pm 0.008}$	0.359 ± 0.012
CTM	Doc2Vec	0.558	0.230 ± 0.017	0.546 ± 0.016	0.235 ± 0.029
CTM	SBERT	0.704	0.329 ± 0.017	0.576 ± 0.020	0.277 ± 0.041
LDA	Doc2Vec	0.576	0.248 ± 0.028	0.529 ± 0.029	0.249 ± 0.014
LDA	SBERT	0.71	0.261 ± 0.034	0.531 ± 0.029	$\mathbf{0.369 \pm 0.051}$

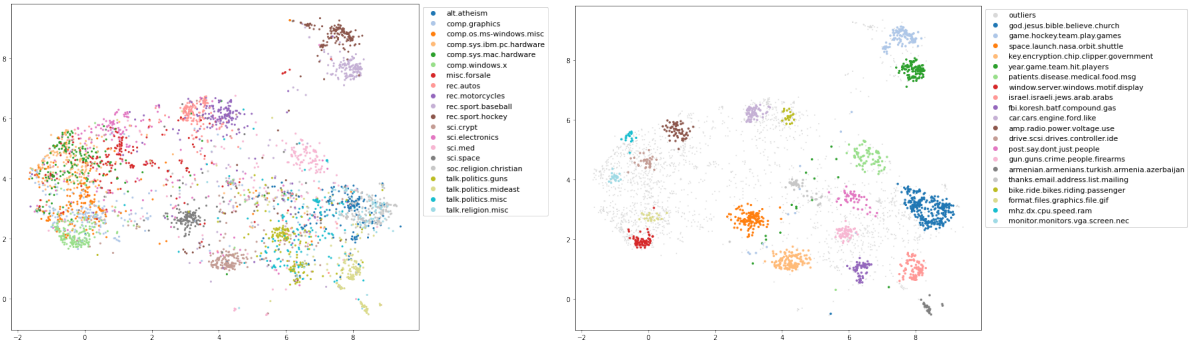
Table 1: Hyperparameter tuning best results per topic and embedding model

Additionally, we present the UMAP 2-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. 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. Additionally, it is possible to see that similarly semantic 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*.



(a) Original labels (b) Topic labels

Figure 5: Comparison between UMAP plots of **train data** with original and topic labels



(a) Original labels (b) Topic labels

Figure 6: Comparison between UMAP plots of **test data** with original and topic labels

An important characteristic of BERTopic is that it is able to identify noise, leading to a topic assignment where part 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 the percentage of documents classified into the aforementioned category is 51.4%.

5 Case Study

This use case is based on a real-world example, related with 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 allows 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 a step-by-step process of how the CI analyst can use MapIntel to screen the news and select those that are relevant to include in the reports. This process cannot be seen strictly 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 starts with a set of keywords that the analyst selects based on his/her knowledge of the market or industry. These keywords are going to be used as seeds in the process of exploring the corpora. Each seed will be projected onto the two-dimensional surface produced by MapIntel, defining a set of neighbors in this semantic space, thus generating a number of candidate documents relevant for 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 2 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 general 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 2: 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 a period of around 5 months.

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 2 so it can be used downstream by the Query pipeline 3 and the Visualization pipeline 4.

[3] newsapi.org

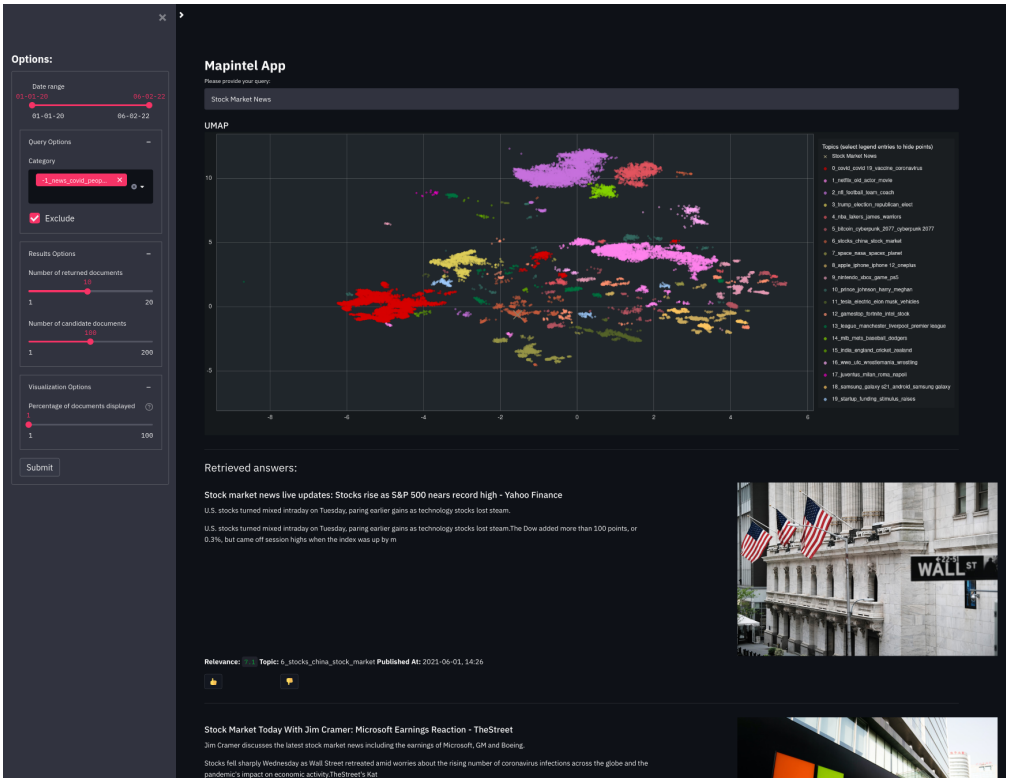


Figure 7: MapIntel webapp interface

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 API requests.

We highlight the user interface of the app in Figure 7. It is composed of 4 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 3 some query examples (without any filters) a CI analyst could make and their respective results. We can see that the results are semantically relevant for the query provided.

[4]fastapi.tiangolo.com
[5]streamlit.io

Query	Top 3 Results (Title)	Score	Result Topic
"Stock Market News"	Stock Market Today With Jim Cramer: Microsoft Earnings Reaction - TheStreet	6.90	7
	Stock Market Today With Jim Cramer: Home Depot Falls - TheStreet	6.27	-1
	Stock Market Today With Jim Cramer: Apple Event Preview - TheStreet	6.12	-1
"Indian Elections"	India elections: Modi party defeated in West Bengal battleground - BBC News	5.36	11
	At 103, India's first voter casts vote in Himachal panchayat polls	4.77	-1
	Vote count in five Indian states under way as pandemic rages	4.44	-1
"Oil Prices"	Oil prices near 2-year highs above $70 as investors expect OPEC+ to confirm its supply policies	7.63	-1
	Oil Prices Rally Towards $70 As Demand Outlook Improves - OilPrice.com	7.62	-1
	Oil prices to reach $72 by summer: Goldman Sachs - Fox Business	7.45	7

Table 3: Example queries top 3 results and respective scores

An additional feature of the app is the ability to browse documents through the scatter plot. This visualization shows the 2-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.

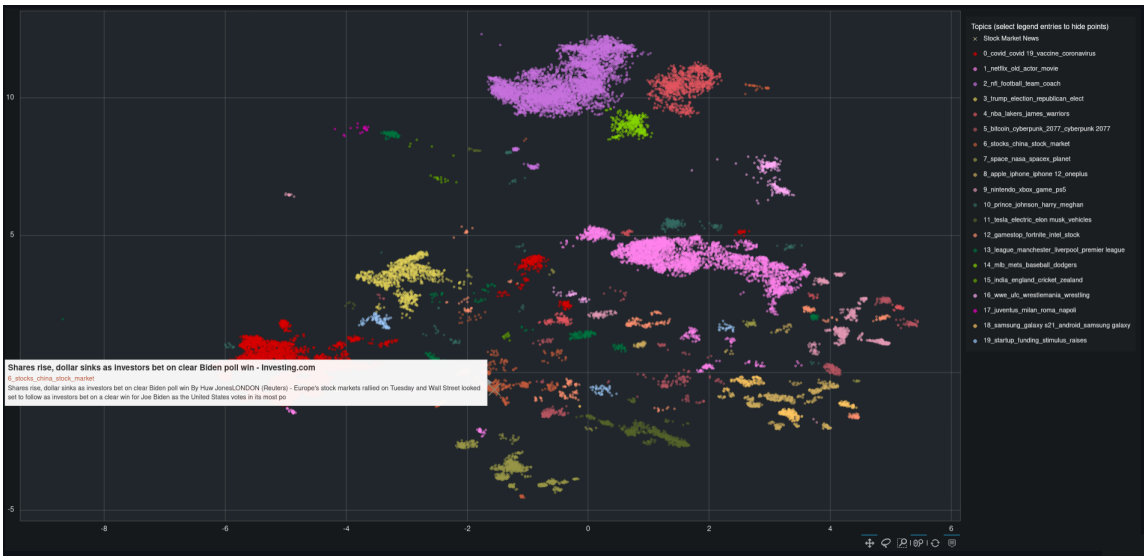


Figure 8: UMAP projection closeup

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 and the resulting topic labels can be seen in Table 4. The topic labels can be particularly useful to the analyst to understand the different subjects covered by the corpus at a glance. 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 4: Topic labels and respective coherence values

6 Conclusion and Perspectives

In this paper, we present MapIntel, a new system for extracting knowledge from large corpora of text documents. MapIntel differentiates from previous similar systems in that it leverages Transformer-based document embeddings to provide efficient, natural language searching of documents, a 2-dimensional map of the documents, and a topical organization of the corpus. The system is centered around the concept of Information Encountering Erdelez and Makri (2020), providing *browsing* and *searching* capabilities to acquire information and promote serendipity. MapIntel is aimed at supporting Competitive Intelligence analysts by providing a tool that facilitates the exploration and monitoring of the competitive environment from textual data.

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 we 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 continued.

Our next steps will be to perform a more extensive evaluation of the system with new corpora and develop a case study more closely with CI analysts from AICEP to better understand their needs. Some different directions would be to expand the system to different application domains to test its generality, include multilingual text documents to more easily monitor international events, and provide a way to interact with subsets of documents through manual selection in the interactive map.

References

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.

Brod, S. (1999). Competitive intelligence: Harvesting information to compete and market intelligently. *Camares Communications, New York, NY.*

Calof, J., Sewdass, N., and Arcos, R. (2017). Competitive Intelligence: A 10-year Global Development. Technical report, Competitive Intelligence Foundation.

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.

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].*

Marin, J. and Poulter, A. (2004). Dissemination of Competitive Intelligence. *Journal of Information Science*, 30(2):165–180.

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.

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]*.