Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Data Mining and Artificial Intelligence Research Group

Topic Modelling Of OpenML Dataset Descriptions

Master Thesis

Ivan Germanov

Supervisors:
prof. Joaquin Vanschoren
MSc Taniya Das
Your Third Committee Member, usually the external member

1.0

Eindhoven, October 2024

Abstract

THIS IS MY ABSTRACT

Acknowledgements

This thesis marks the end of my Master's degree in Computer Science at the Eindhoven University of Technology. I would like to express my deepest gratitude to my supervisor, prof. Joaquin Vanschoren, for his invaluable guidance and continuous support throughout the project.

I am also sincerely thankful to my daily supervisor, Taniya Das, for her timely feedback and suggestions, and for pointing me to the relevant literature, all of which were instrumental in shaping the outcome of this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Topic modeling is a rapidly growing field with applications in various contexts that include text corpora - social media posts [1, 2, 3], books [4], newspapers [5, 6, 7], legal documents [8, 9], research papers [10] and financial reports [11, 12], to name a few. Topic models can take a large corpus of documents as input and extract the latent topics present in the corpus [13]. A topic refers to a recurring pattern of words (terms) or phrases that commonly occur together in a set of documents [14]. For instance, in a collection of news articles, a topic T_1 may consist of the terms *election*, *candidate*, and *vote*, while another topic T_2 may consist of the terms *stock*, *market*, and *investment*.

Churchill and Singh [15] define a topic model to be a mathematical model that takes as input a set of documents D, and returns a set of topics T that represent the content of D in an accurate and coherent manner. The documents within the collection can subsequently be tagged with these identified topics. This process enables users to discern the importance of each topic both within individual documents and across the entire collection.

OpenML [16] is a platform designed for machine learning researchers to share and manage data. It facilitates global collaboration by allowing users to present new datasets for analysis and share their findings, including code, models, predictions, and evaluations. OpenML ensures the clear definition of tasks and organizes all contributions online for easy accessibility, reuse, and discussion.

For each dataset, OpenML provides a dedicated page that contains detailed information such as a general dataset description, attribution details, and characteristics of the data, as well as statistics on the data distribution. Additionally, OpenML supports the use of tags on datasets, facilitating easier filtering and searchability.

1.1 Problem formulation and goal

In OpenML, datasets are currently categorized using manual tagging. However, many datasets lack semantic tags that are readable by humans. This situation presents an opportunity for a Master's thesis project aimed at developing an unsupervised, automated topic modelling system for tagging datasets. Given that most datasets come with descriptions, applying topic modeling to extract topics is an innovative approach to generate and assign relevant tags. Most topic modeling techniques are unsupervised, meaning they do not require labeled data for training. This characteristic makes them suitable for the task of tagging OpenML datasets, as the tags are not predefined.

Applying topic modeling to extract topics as tags could improve how users interact with the OpenML platform. Specifically, it could make the process of searching and filtering through the extensive collection of datasets more efficient, thus improving dataset discoverability. The addition of semantic tags based on the topics identified in the descriptions could also lead to better organization and management of datasets, thereby improving data governance on the platform.

Automating the process of tagging can save considerable time for researchers and data scientists who would otherwise have to tag datasets manually. This method ensures consistency in the tags

applied and enriches the datasets' metadata, making them more useful and accessible.

Furthermore, previous work by Das has shown the potential of using scripts to automate the tagging of datasets in OpenML [17]. Das's approach involved using dataset descriptions and a predefined list of tags to prompt GPT-3.5-turbo to assign relevant semantic tags to each dataset. This method demonstrated the feasibility of classifying datasets with a set of predefined tags, similar to the dataset tags in the Wolfram Data Repository [18].

The main goal of this research is to explore the potential of unsupervised topic modeling when applied to the dataset descriptions of OpenML. By extracting topics from the descriptions, we can use the terms in the topics as tags for the datasets.

1.2 Research questions

To refine the main goal of the research, we define the following research questions:

- **RQ1 — What are the specifications of the OpenML dataset descriptions, and what impact may they have on model performance?** Explore the dataset descriptions in OpenML and analyze their characteristics to understand the challenges and opportunities for extracting topics. Evaluate whether additional preprocessing steps are necessary to improve the quality of the descriptions to be used as input for the topic model.
- **RQ2 — What are the different approaches to topic modeling that can be applied to the OpenML dataset descriptions, and what are the tradeoffs involved in their use?** Investigate existing topic modeling techniques and architectures and assess their suitability for extracting topics from the dataset descriptions, while considering tradeoffs such as computational cost, scalability, and model complexity. Additionally, explore strategies to ensure that the generated tags can be continuously refined and improved as new topic modeling approaches and advancements become available.
- **RQ3 — What are suitable automated evaluation metrics for assessing the quality of the topics and terms extracted by the topic model?** Define and implement automated evaluation metrics to quantitatively measure the quality of the topics and terms extracted by the topic model. Compare the performance of different topic models (benchmarks) using these metrics.
- **RQ4 — What are suitable human evaluation methods for assessing the quality of the topics and terms extracted by the topic model?** Define and implement human-centered evaluation strategies to assess the usefulness and coherence of the topics and terms from a human perspective. This evaluation will consider how well the extracted topics align with a human's understanding and expectations.

1.3 Contributions

This research aims to contribute to the OpenML platform by providing a novel approach to tagging datasets. By automating the tagging process, the research can improve the efficiency of dataset management and organization on the platform. This contribution can benefit researchers and data scientists who use OpenML by making it easier to search for and filter datasets based on their content.

Furthermore, the research aims to explore the potential of topic modeling when applied to dataset descriptions, an area that to our knowledge has not been studied. By doing so, it seeks to contribute new insights to the field of topic modeling, which has been applied in various contexts but not extensively in the categorization of datasets based on their descriptions.

Another contribution of this research is the development of a new topic model that not only extracts topics from the specific context of OpenML dataset descriptions, but that is also generalizable to other text corpora. We also provide a set of automated and human evaluation metrics to assess the quality of the topics and terms extracted by the model.

Chapter 2

Preliminaries

The goal of this chapter is to introduce the reader with the preliminary knowledge which is needed for understanding this research. We present an in-depth literature review of the current state-of-the-art and most widely adopted topic modeling techniques, along with the evaluation metrics used to assess the quality of the extracted topics. The foundational models discussed include Latent Dirichlet Allocation (LDA), Non-negative Matrix Factorization (NMF), and Top2Vec.

For the reader that is primarily interested in the approach that will be central to this research, we encourage you to focus on the subsection about BERTopic. As BERTopic will serve as the primary model in this research, the others — LDA, NMF, Top2Vec — will be used as baseline models for comparison.

Readers interested in a comprehensive overview of the field are invited to explore the broader literature in appendix A, which provides a detailed account of topic modeling techniques, their applications and history.

2.1 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) [13] is a seminal and widely adopted generative probabilistic model that assumes documents are a mixture of topics, and topics are a mixture of words. It is based on the bag-of-words assumption, i.e. the order of words in a document does not matter.

Figure 2.1 illustrates the plate notation for LDA. Each plate can be viewed as a "loop", where the variable in the bottom right can be seen as the number of iterations of the loop. The figure shows that there are K topics whose Dirichlet distribution over words is controlled by the hyper-parameter β. The plate below shows that there are M documents, each of which has N words. The gray circle with w represents the observed word, while the other circles represent latent variables. z refers to the topic of w, θ refers to the Dirichlet distribution of topics over documents, which is controlled by the hyper-parameter α.

The generative process for a corpus in the context of LDA is as follows:

1. For each document $i = 1, \ldots, M$:
   - Sample θ from a Dirichlet distribution $\theta_i \sim \text{Dir}(\alpha)$.
2. For each topic $k = 1, \ldots, K$:
   - Sample ϕ from another Dirichlet distribution $\phi_k \sim \text{Dir}(\beta)$.
3. For each word $j = 1, \ldots, N$ in document i:
   - Sample a topic $z_{ij} \sim \text{Multinomial}(\theta_i)$.
   - Sample a word $w_{ij} \sim \text{Multinomial}(\phi_{z_{ij}})$.

θ_{ik} represents the probability of the i-th document to contain words from the k-th topic. Similarly, ϕ_{kw} represents the probability of the k-th topic to contain the w-th word.

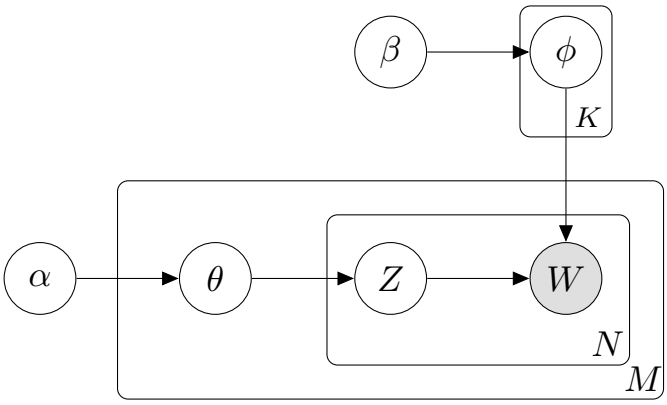

Figure 2.1: LDA plate notation

2.1.1 Dirichlet distribution

Take the example of a large digital library of academic papers. First, for each paper i, we sample its topic distribution θ_i from a Dirichlet distribution. This represents the mixture of topics covered by the document. Secondly, for each topic k, we sample a word distribution ϕ_k over each topic from a Dirichlet distribution. Then, for each word j in the document, we draw a topic z_{ij} from the topic distribution Multinomial(θ_i), followed by sampling a word w_{ij} from the word distribution Multinomial($\phi_{z_{ij}}$). This process models the generation of words in an academic paper based on latent topic structures and their corresponding word distributions.

The intuition behind the Dirichlet distribution is that the k-dimensional Dirichlet distribution θ is controlled by a k-dimensional vector of positive real numbers, α. The α parameter shapes how topics are distributed across documents. A uniform α suggests no prior preference for topic prevalence, leading to a balanced mix of topics within documents. Smaller α values push the model towards sparser topic representations, where documents are likely to be dominated by fewer topics. An asymmetric α allows for the modeling of prior beliefs about topic prevalence, making some topics more prominent than others.

Similarly, β controls the concentration of the word distribution for each topic, where the m-dimensional Dirichlet distribution ϕ is controlled by a m-dimensional vector of positive real numbers, β.

2.1.2 Learning LDA

The problem of learning an LDA model is referred to as an "inference" problem. That is, given the observed variable, w, and the hyper-parameters α and β, how do we estimate the posterior of the latent variables:

$$p(\theta, z, \phi|w, \alpha, \beta) = \frac{p(\theta, z, \phi, w|\alpha, \beta)}{p(w|\alpha, \beta)}$$

Blei et al. [13] discover that the integral for computing in the denominator is infeasible to compute exactly:

$$p(w|\alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_i \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \prod_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n} \right) d\theta$$

Therefore, approximate inference must be applied. Common approaches are Gibbs sampling and variational inference. Without delving into too much detail, Gibbs sampling allows us to avoid directly computing the intractable integral. The basic idea is that we want to sample from $p(w \mid \alpha, \beta)$ to estimate the distribution, but we cannot directly do so. Instead, Gibbs sampling

allows us to iteratively compute the posterior of one of the latent variables while fixing all the other variables. This way, we can obtain the posterior distribution $p(\theta, z, \phi \mid w, \alpha, \beta)$.

For each iteration, we alternatively sample θ, z, ϕ with all the other variables fixed. Because the samples from the early iterations are not stable, we discard the first B iterations of samples. The algorithm is shown in the following pseudo code:

For i from 1 to MaxIter:

- Sample $\theta_i \sim p(\theta \mid z = z_{i-1}, \phi = \phi_{i-1}, w, \alpha, \beta)$
- Sample $z_i \sim p(z \mid \theta = \theta_i, \phi = \phi_{i-1}, w, \alpha, \beta)$
- Sample $\phi_i \sim p(\phi \mid \theta = \theta_i, z = z_i, w, \alpha, \beta)$

The algorithm begins with initial, possibly random, values for the variables θ, z, and ϕ, and proceeds through a series of iterations up to a predefined maximum number, MaxIter. At each iteration i, the value of θ_i is sampled from its conditional distribution given the current values of z and ϕ, denoted z_{i-1} and ϕ_{i-1} to reflect their values from the previous iteration, alongside any observed data or parameters w, α, and β. Following this, z_i is updated based on the new θ_i and the previous ϕ_{i-1}, and finally, ϕ_i is sampled using the latest values of θ_i and z_i. This sequential updating of variables leverages the simpler conditional distributions to approximate the complex joint distribution. As the number of iterations increases, the algorithm converges, meaning the samples generated become representative of the target distribution.

2.2 Non-negative Matrix Factorization

Non-negative Matrix Factorization (NMF) [19, 20, 21] is a technique where an original matrix, consisting of non-negative values, is decomposed into two distinct matrices. The fundamental principle of NMF is that the product of these two matrices approximates the original matrix. This decomposition is a form of dimensionality reduction. The large original matrix typically represents a set of documents, with each document being a vector of words. The two resultant matrices in NMF are the word-topic matrix and the topic-document matrix. The topic-word matrix shows the association between topics and words, while the topic-document matrix shows the relationship between topics and individual documents.

Figure 2.2 illustrates the decomposition of a matrix A into two matrices W and H. The matrix A is a non-negative matrix, and the matrices W and H are also non-negative. The product of W and H approximates A. A represents the word-document matrix, where each row corresponds to a word and each column corresponds to a document. W represents the word-topic matrix, where each row corresponds to a word and each column corresponds to a topic. H represents the topic-document matrix, where each row corresponds to a topic and each column corresponds to a document.

Non-negative Matrix Factorization is a group of algorithms whose objective is to minimize F - the function which measures the error between the original matrix and the product of the two matrices. The most common algorithms for NMF typically involve iterative update rules that aim to minimize F, such as the Frobenius norm or the Kullback-Leibler (KL) divergence.

2.2.1 Frobenius norm

The goal in NMF using the Frobenius norm is to minimize the objective function F, which is given by:

$$F = \|A - WH\|_F^2 = \sum_{i=1}^{n} \sum_{j=1}^{m} (A_{ij} - (WH)_{ij})^2$$

where $\|\cdot\|_F$ denotes the Frobenius norm. This objective function represents the sum of the squares of the element-wise differences between A and the product WH.

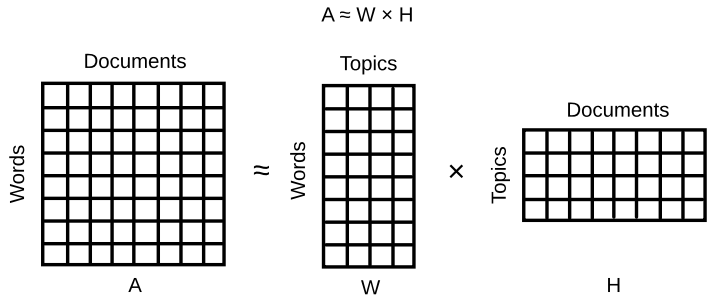

Figure 2.2: NMF decomposition

The typical algorithm to minimize the Frobenius norm in NMF is an iterative process that involves:

1. **Initialization:** Matrices W and H are initialized with non-negative values. This can be done randomly or based on some informed heuristic.

2. **Iterative Update:** The matrices W and H are updated iteratively to reduce F. The updates are performed using multiplicative rules that inherently maintain the non-negativity of W and H. The update rules are as follows:

$$W_{ai} \leftarrow W_{ai} \cdot \frac{(AH^\top)_{ai}}{(WHH^\top)_{ai}}$$

$$H_{ib} \leftarrow H_{ib} \cdot \frac{(W^\top A)_{ib}}{(W^\top WH)_{ib}}$$

   where the indices a and b iterate over all rows and columns of W and H, respectively.

3. **Convergence:** The iteration continues until the change in F between successive iterations is less than a predetermined threshold, or a maximum number of iterations has been reached.

While the Frobenius norm-based NMF is not convex over both W and H together, it is convex over each one individually when the other is held constant. Thus, each iteration is guaranteed to not increase F, although the solution may converge to a local minimum rather than a global minimum.

2.2.2 Kullback-Leibler

Unlike the Frobenius norm which assesses the difference based on squared errors, the KL divergence is more suitable for data that is inherently probabilistic. The KL divergence for two matrices is defined as:

$$D(A||WH) = \sum_{i=1}^{n}\sum_{j=1}^{m}\left(A_{ij}\log\frac{A_{ij}}{(WH)_{ij}} - A_{ij} + (WH)_{ij}\right)$$

where $D(A||WH)$ represents the KL divergence between A and WH, with the objective to minimize this divergence in NMF.

The iterative update rules for the matrices W and H that minimize the KL divergence are as follows:

$$W_{ai} \leftarrow W_{ai} \cdot \frac{(A \oslash (WH)H^\top)_{ai}}{\mathbf{1}H^\top_{ai}}$$

$$H_{ib} \leftarrow H_{ib} \cdot \frac{(W^\top(A \oslash (WH)))_{ib}}{W^\top \mathbf{1}_{ib}}$$

Here, $\oslash$ denotes element-wise division, and $\mathbf{1}$ is a matrix of ones that is used for normalization in the denominators.

Just like in the case of the Frobenius norm, the KL divergence-based NMF aims to iteratively update W and H until the decrease in $D(A||WH)$ is below a certain threshold, signaling convergence. However, it is important to note that this optimization problem is non-convex, and the solution found may represent a local minimum.

2.3 Top2Vec

A limitation of LDA and NMF is that they disregard semantic relationships between words, thus neglecting context. As a result, text embedding techniques which capture context have become popular as an NLP technique.

2.3.1 Embeddings

In Top2Vec [22], the first step is to embed the documents into dense vector representations (embeddings) to capture the semantic meaning of the text. Figure 2.3 [22] illustrates the embeddings, where the purple dots represent words and the green dots represent documents. Words are closest to documents that contain them, and documents are closest to words that are most representative of their content.

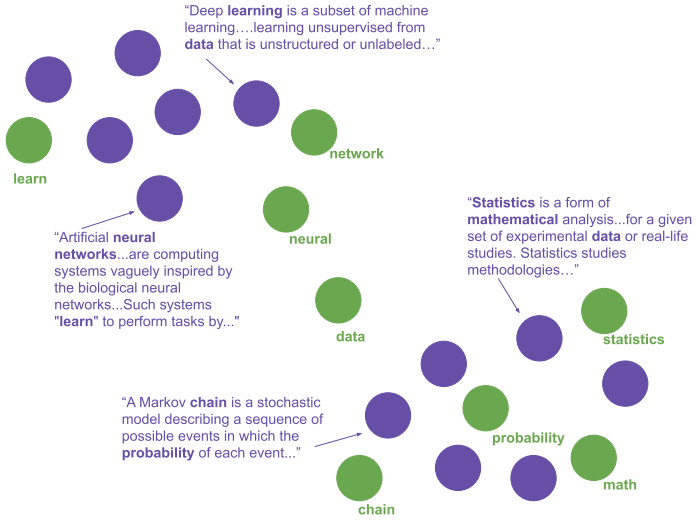

Figure 2.3: Example of embeddings

To learn the embeddings, Top2Vec utilizes Doc2Vec [23, 24], Universal Sentence Encoder [25], or Sentence-BERT (SBERT) [26].

The original paper uses Doc2Vec's Distributed Bag of Words (DBOW) model, and even though it is simpler than Doc2Vec's Distributed Memory (DM) model, it is more efficient and has been

shown to perform better in practice [27]. DBOW essentially uses the document vector to predict words within a context window in the document.

Doc2Vec's DBOW is similar to Word2Vec's Skip-gram model (appendix A.5), which uses the context word to predict surrounding words in the context window. The difference is that DBOW switches the context word for the document vector to predict the surrounding words in the context window.

The process of learning the embeddings in Top2Vec can be summarized as follows:

1. **Matrix Initialization**: The process initiates with the establishment of two matrices. The document matrix, denoted as $D_{c,d}$, encapsulates document vectors where c represents the corpus's document count and d the embedding dimensionality. Each row within $D_{c,d}$ represents a distinct document vector $\vec{d} \in \mathbb{R}^d$. Concurrently, the context word matrix $W'_{n,d}$, representing word vectors in analogous d-dimensional space for n vocabulary words, may originate from pre-training, random initialization, or parallel learning processes.

2. **Word Prediction Mechanism**: Contrary to relying on neighboring context words for prediction, the DBOW model employs the document vector for prediction. For every document d, each word's context vector $\vec{w_c}'$ within d (sourced from $W'_{n,d}$) aids in inferring the document's vector $\vec{d}$ in $D_{c,d}$. This inference employs a softmax function, $\text{softmax}(\vec{w_c}' \cdot D_{c,d})$, generating a corpus-wide probability distribution reflecting each document's likelihood of generating the word.

3. **Learning Process**: The learning process aims to optimize the document and word vectors to predict the document's constituent words. This optimization leverages backpropagation and stochastic gradient descent to modify both $D_{c,d}$ and $\vec{w_c}'$ from $W'_{n,d}$ to maximize the probability $P(\vec{d}|\vec{w})$ of correctly predicting the document given its words.

4. **Embeddings**: Through optimization, embeddings emerge where documents gravitate towards the vectors of words they comprise, effectively "attracted" by these words. Consequently, semantically similar documents (sharing similar words) cluster, whereas dissimilar documents (sharing fewer words) diverge.

2.3.2 Number of topics

In the embeddings, a dense area of documents can be interpreted as an area of highly similar documents. First, Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [28] is used to reduce the dimensionality of the document vectors. That is because the high-dimensional document vectors lead to the *curse of dimensionality*, where the document vector sparsity makes it difficult to find dense clusters. Then, in order find the dense areas of documents in the embeddings, density-based clustering is used on the document vectors, specifically Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [29, 30, 31]. HDBSCAN assigns a label to each dense cluster of document vectors and assigns a noise label to all document vectors that are not in a dense cluster.

2.3.3 Topic vectors

Given labels for each cluster of dense documents in the embeddings, topic vectors can be calculated. The authors lay out multiple methods for calculating topic vectors, but discover that they perform similarly. The method that is used in the original paper is to calculate the centroid of the document vectors in each cluster. The centroid is the average of all the document vectors in the cluster. The centroid is calculated for each set of document vectors that belong do a dense cluster, generating a topic vector for each set. The number of dense areas found is the number of prominent topics identified in the corpus.

In the embeddings, every point represents a topic that is best described semantically by its nearest word vectors. Therefore, the word vectors that are closest to a topic vector are those that

are most representative of it semantically. The distance of each word vector to the topic vector will indicate how semantically similar the word is to the topic. The words closest to the topic vector can be seen as the words that are most similar to all documents into the dense area, as the topic vector is the centroid of that area. These words can be used to summarize the common topic of the documents in the dense area.

2.4 BERTopic

Top2Vec simplifies the process of generating topics by clustering embeddings of words and documents. Inspired from Top2Vec, BERTopic [32] is a state-of-the-art topic model that builds on top of the embeddings approach.

BERTopic generates representations of topics through a six-step process:

1. Initially, it transforms each document into an embedding using a pre-trained language model.
2. Before the clustering process, the dimensionality of these embeddings is reduced.
3. Following this, the embeddings are clustered.
4. Subsequently, a bag-of-words representation is generated for each cluster, containing the frequency of every word.
5. Next, topic representations are derived from these clusters using a specialized class-based version of TF-IDF.
6. The final step optionally fine-tunes these topic representations. Each of these steps is modular, allowing for the use of different techniques at each stage.

While these steps are the default, BERTopic offers a degree of modularity. Each step in the process is relatively independent from the others. For instance, the bag-of-words step does not depend on the specific embedding model used for document embeddings, which provides flexibility in how the bag-of-words representation is calculated.

As a result, BERTopic is highly modular, maintaining its ability to generate topics across different sub-models. This means that BERTopic effectively allows for the construction of customized topic models, appropriate for the specific use case. Additionally, the modularity of BERTopic ensures that it can be easily adapted to incorporate new advancements in the field of NLP, since each component can be changed independently from the others. Figure 2.4 (inspired by [33]) illustrates the six steps of BERTopic, presented from bottom to top. It highlights the possibility of employing various techniques at each step of the process. For example, one could choose between SBERT or spaCy for document embedding, UMAP [28] or PCA [34] for dimensionality reduction, and GPT or KeyBERT [35] for the fine-tuning phase.

2.4.1 Document embeddings

In BERTopic, documents are transformed into embeddings to create vector space representations for semantic comparison. It is based on the idea that documents sharing the same topic will have similar semantics. For this embedding step, by default BERTopic utilizes the SBERT framework [26]. SBERT is a modification of the popular Bidirectional Encoder Representations from Transformer (BERT) model (appendix A.6), and enables the conversion of sentences and paragraphs into dense vector representations by employing pre-trained language models. This achieves top performance on several sentence embedding tasks [36]. The embeddings are mainly used for clustering documents with semantic similarities rather than directly for topic generation.

The base Transformer architecture is presented in Figure 2.5 [37]. It uses a unidirectional multi-head self-attention mechanism to capture dependencies between words in a sentence. Unidirectional means that the attention mechanism only looks at previous words in the sentence, which can be

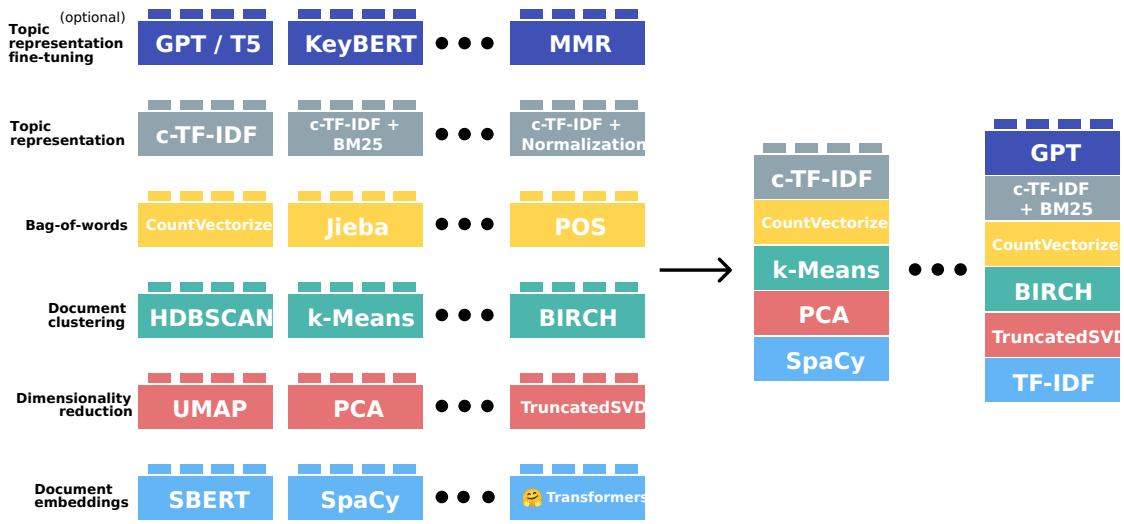

Figure 2.4: BERTopic modularity and steps (from bottom to top)

a limitation. The model consists of a stacked encoder-decoder architecture, where the encoder processes the input sequence, and the decoder generates the output sequence. The encoder is composed of a stack of identical layers, each containing two sub-layers: a multi-head self-attention mechanism and a feed-forward neural network. The decoder is also composed of a stack of identical layers, each containing three sub-layers: a multi-head self-attention mechanism, a multi-head attention mechanism over the encoder's output, and a feed-forward neural network. The self-attention mechanism allows the model to weigh the importance of each word in the sentence when generating the embeddings.

BERT, in contrast, is a bidirectional model that captures dependencies between words in both directions (Figure 2.7, Figure 2.8, Figure 2.6) [37, 38]. It achieves this by leveraging a *masked language model (MLM)* objective, where the model is trained to predict masked words within a sentence. Pre-trained on a large corpus of text, BERT's embeddings can be fine-tuned for various downstream tasks, such as sentence classification or question answering. However, tasks relevant to topic modeling, such as semantic similarity, are computationally expensive due to BERT's cross-encoder architecture. In a cross-encoder, pairs of sentences are passed through the transformer network together, and the target value is predicted. For example, in a collection of $n = 10,000$ sentences, finding the most similar pair using BERT would require $n \times (n-1)/2 = 49,995,000$ inference computations [26].

SBERT addresses this issue by incorporating a Siamese network architecture with a bi-encoder setup, where two identical BERT models share the same weights (Figure 2.9). In SBERT, each sentence is processed independently, and the embeddings are computed separately. As a result, for $n = 10,000$ sentences, only 10,000 inference computations are needed. Once the embeddings are generated, similarity can be easily calculated, as it is a computationally efficient operation.

BERTopic can use any embedding technique or architecture, provided the language model used for generating document embeddings is fine-tuned for semantic similarity. Hence, the quality of BERTopic's clustering improves as more advanced language models are developed, allowing BERTopic to evolve alongside advancements in embedding techniques. This proves particularly useful in the context of topic modeling, since the quality of the embeddings directly influences the quality of the topics generated.

The Massive Text Embedding Benchmark (MTEB) [39] is a benchmark that evaluates the performance of various embedding models on a wide range of tasks. It provides a comprehensive comparison of the performance of different embedding models, including SBERT, RoBERTa [40], and BERT. By leveraging MTEB, the user can select the most suitable embedding model for their specific use case.

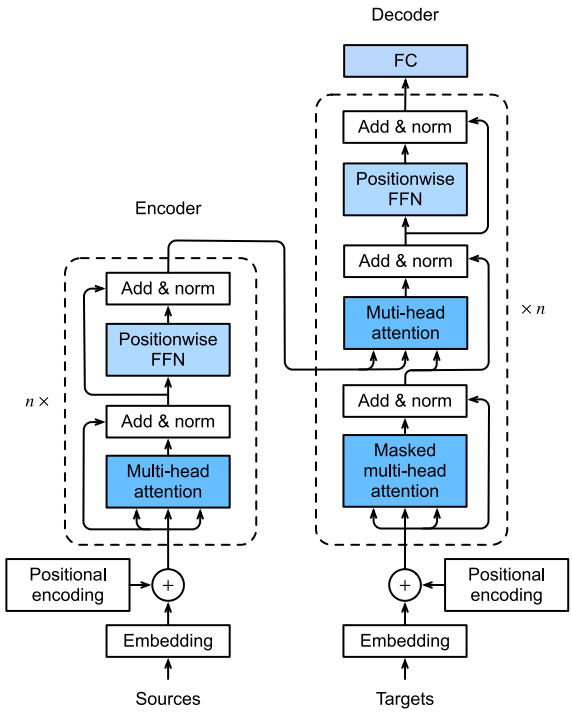

Figure 2.5: Transformer architecture

2.4.2 Dimensionality reduction

As the dimensionality of data (embeddings) increases, the distance to the nearest data point tends to become similar to the distance to the farthest data point [41, 42]. This phenomenon implies that in high-dimensional spaces, the notion of spatial locality becomes unclear, and distances between points become increasingly small. While several clustering methods have been developed to address this *curse of dimensionality* [43, 44], a simpler strategy involves reducing the dimensionality of embeddings. Although PCA [34] and t-SNE [45] are popular dimensionality reduction techniques, UMAP has been found to better preserve the local and global characteristics of high-dimensional data in its lower-dimensional representations [28].

UMAP is a non-linear dimensionality reduction technique that is rooted in manifold learning and is mathematically grounded in *topological data analysis*. More specifically, UMAP seeks to approximate a high-dimensional manifold by constructing a weighted k-nearest neighbor (k-NN) graph and then optimizing a low-dimensional layout of this graph.

The theory behind UMAP is mathematically complex, but the algorithm can be summarized as follows:

1. **Constructing the k-NN Graph**:

The initial phase of UMAP involves constructing a k-nearest neighbor graph from the high-dimensional data. In this step, for each data point, the algorithm identifies its nearest neighbors based on a distance metric (such as Euclidean distance). The choice of k determines the number of neighbors to consider, and this influences the local structure captured by the graph. Importantly, UMAP assumes that the data points are uniformly distributed on the underlying manifold, and it computes a local distance metric for each point using *Riemannian geometry*. This local metric is based on the distance to the k-th nearest neighbor, creating a unique distance function for each point.

2. **Fuzzy Simplicial Complex Representation**:

Once the k-NN graph is built, UMAP constructs a fuzzy simplicial complex. This is a crucial step rooted in *topological data analysis*. Instead of a binary decision about whether or not a point

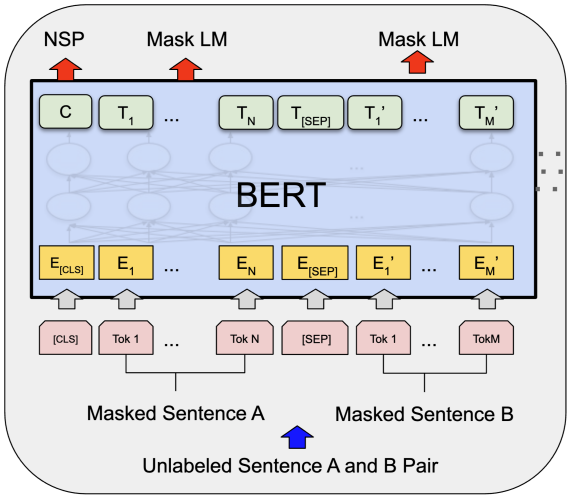

Figure 2.6: Transformer architecture

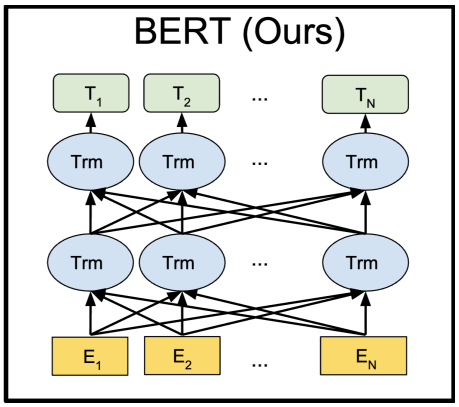

Figure 2.7: BERT Bidirectional Architecture

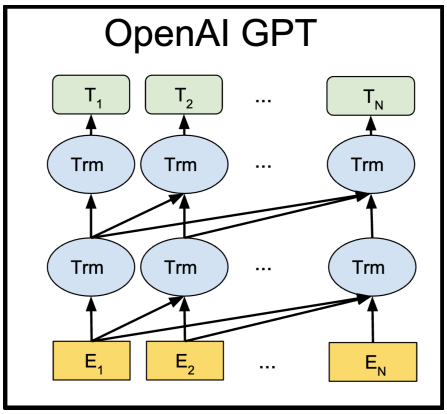

Figure 2.8: OpenAI GPT Unidirectional Architecture

belongs to a neighborhood, UMAP assigns fuzzy membership values to reflect the probability of points being connected. This fuzzy approach smooths out the local connectivity of the manifold. The fuzzy simplicial set is built by combining local fuzzy simplicial sets of each point, using a probabilistic union operation to merge the connections between points. This union operation involves combining the fuzzy memberships of edges (1-simplices) between points, which can be expressed mathematically as $w(e) = a+b-ab$, where a and b are the fuzzy membership probabilities for the edge in each direction, and $w(e)$ is the resulting combined weight.

3. **Low-Dimensional Embedding Optimization**:
After constructing the fuzzy topological representation, UMAP seeks to optimize a low-dimensional embedding that preserves the topological structure of the original data. This process can be viewed as a *graph layout problem*. UMAP minimizes the cross-entropy between the fuzzy topological structures of the high-dimensional data and the low-dimensional embedding. The cross-entropy loss is given by:

$$\sum_{e \in E} w_h(e) \log\left(\frac{w_h(e)}{w_l(e)}\right) + (1 - w_h(e)) \log\left(\frac{1 - w_h(e)}{1 - w_l(e)}\right)$$

Here, $w_h(e)$ and $w_l(e)$ represent the fuzzy membership weights of the edge e in the high-dimensional and low-dimensional spaces, respectively. The first term represents an *attractive force*,

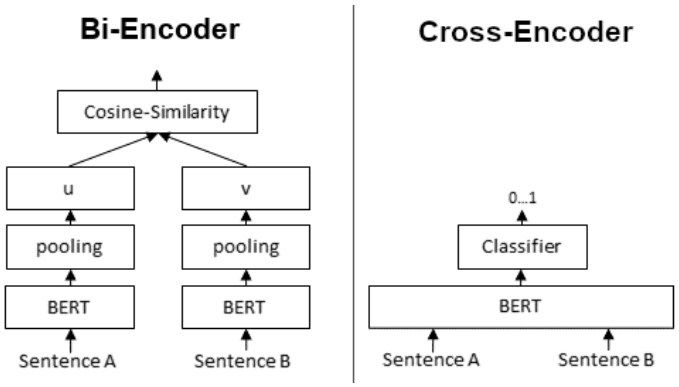

Figure 2.9: Transformer architecture

pulling points that are close in the high-dimensional space to be close in the low-dimensional space. The second term represents a *repulsive force*, pushing points that are distant in the high-dimensional space to remain distant in the low-dimensional space.

4. **Use of Negative Sampling**:
To further optimize the computational efficiency of the embedding process, UMAP uses the *negative sampling trick*, similar to methods employed in algorithms like *word2vec* and *LargeVis*. Instead of computing the cross-entropy over all possible pairs of points, UMAP samples a subset of non-neighboring points (negative samples) to approximate the repulsive forces. This reduces the computational complexity, making the algorithm scalable to large datasets.

5. **Stochastic Gradient Descent (SGD) Optimization**:
The optimization of the low-dimensional embedding is performed using *stochastic gradient descent (SGD)*. The final objective function is designed to be differentiable, allowing for efficient gradient-based optimization. UMAP uses a smooth approximation for the membership strength function in the low-dimensional space, typically of the form $\frac{1}{1+ax^{2b}}$, where a and b are curve parameters selected to best approximate the fuzzy memberships in the low-dimensional space. The optimization process iteratively adjusts the positions of points in the low-dimensional space until the cross-entropy loss is minimized.

6. **Initialization with Spectral Embedding**:
To speed up convergence, UMAP often initializes the low-dimensional embedding using *spectral embedding techniques*. This involves computing the eigenvectors of the graph Laplacian, which provides an initial guess for the positions of the points in the low-dimensional space. The spectral embedding serves as a good starting point for the SGD optimization.

By combining these steps, UMAP produces a low-dimensional embedding that preserves both the local and global structure of the high-dimensional data.

2.4.3 Document clustering

The reduced embeddings are clustered using a clustering model/algorithm. A popular choice is HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [29, 46, 30, 31]. HDBSCAN is built on top of DBSCAN [47] and is designed to identify clusters of various densities by transforming DBSCAN into a hierarchical clustering algorithm. It employs a soft-clustering approach, which allows for the treatment of noise as outliers. Traditional clustering assigns each point in a dataset to a cluster, which is a hard assignment without mixed memberships. Conversely, in soft clustering, points are not assigned a cluster label, but are instead assigned a vector of probabilities. This allows points to potentially be a mix of clusters. Soft clustering is particularly useful in topic modeling, as documents can belong to multiple topics.

Furthermore, Allaoui et al. [48] showed that the performance of well-known clustering algorithms, including k-Means and HDBSCAN, can be significantly improved by reducing the dimensionality of

high-dimensional embeddings with UMAP, in terms of both clustering accuracy and computational time.

HDBSCAN combines hierarchical clustering with density-based clustering. HDBSCAN's algorithm can be summarized as follows:

1. **Core Distance Calculation**:

The first step in HDBSCAN is similar to DBSCAN, but instead of using a fixed distance threshold (ϵ), HDBSCAN calculates the *core distance* for each point in the dataset. The core distance of a point x_p with respect to a parameter `min_samples` is defined as the distance to its `min_samples`-th nearest neighbor:

$$d_{\text{core}}(x_p) = d(x_p, \text{min\_samples-th nearest neighbor}).$$

This core distance acts as a local density estimate, and each point must have at least `min_samples` points within its core distance to be considered a core point.

2. **Mutual Reachability Distance**:

To account for varying densities in the dataset, HDBSCAN uses the *mutual reachability distance* between two points, x_p and x_q. This is defined as the maximum of their core distances and the direct distance between them:

$$d_{\text{mreach}}(x_p, x_q) = \max\left(d_{\text{core}}(x_p), d_{\text{core}}(x_q), d(x_p, x_q)\right).$$

This distance ensures that points in lower-density regions are only connected if they are sufficiently close to higher-density points.

3. **Constructing the Mutual Reachability Graph**:

Using the mutual reachability distance, HDBSCAN constructs a *complete graph* where each data point is a vertex, and the weight of each edge between points x_p and x_q is given by $d_{\text{mreach}}(x_p, x_q)$. This graph is then used to create a *minimum spanning tree* (MST), which forms the basis for the hierarchical structure of the clusters.

4. **Minimum Spanning Tree (MST) Construction**:

The next step is to construct the MST of the mutual reachability graph. This is done by connecting points in a way that minimizes the total edge weight while ensuring all points are connected. The MST provides a hierarchical structure where clusters can be formed by progressively cutting edges at increasing mutual reachability distances.

5. **Hierarchy of Clusters**:

As edges are removed from the MST in increasing order of mutual reachability distance, clusters start to form. Clusters in HDBSCAN are represented as connected components of the graph after removing edges. This process forms a *hierarchical* clustering structure, where each cluster can potentially be split into subclusters at different distance thresholds.

6. **Cluster Stability and Excess of Mass**:

To extract the most significant clusters from the hierarchy, HDBSCAN measures the *stability* of each cluster. Stability is a measure of how long a cluster persists as the mutual reachability distance increases. Formally, the stability of a cluster C_i is defined as the relative excess of mass [29], which measures the area under the curve of the density function as the cluster shrinks:

$$S(C_i) = \sum_{x_j \in C_i} \left(\frac{1}{\epsilon_{\min}(x_j, C_i)} - \frac{1}{\epsilon_{\max}(C_i)} \right),$$

where $\epsilon_{\min}(x_j, C_i)$ is the minimum mutual reachability distance at which point x_j is part of cluster C_i, and $\epsilon_{\max}(C_i)$ is the distance at which C_i either splits or disappears.

7. **Extracting the Optimal Flat Partition**:

Once the hierarchy of clusters is formed, HDBSCAN uses an optimization algorithm to extract the most significant flat partition of clusters. This selection is based on maximizing the sum of the stabilities of the clusters while ensuring that no nested clusters are selected simultaneously. The optimization problem can be formulated as:

$$\max \sum_{i=2}^{\kappa} \delta_i S(C_i), \quad \text{subject to } \sum_{j \in I_h} \delta_j = 1, \forall h \in L,$$

where $\delta_i \in \{0, 1\}$ indicates whether cluster C_i is included, L is the set of leaf clusters, and I_h is the set of all clusters on the path from leaf cluster C_h to the root.

8. **Outlier Detection**:
Points that do not belong to any cluster are labeled as *noise* or *outliers*. These points either do not meet the density requirements to be considered part of a cluster or are isolated early in the hierarchical process. HDBSCAN is robust in handling noise, as it naturally separates outliers from dense regions of data.

9. **Soft Clustering with Probabilities**:
HDBSCAN also supports a *soft clustering* mode, where instead of assigning a hard cluster label to each point, the algorithm calculates a *membership probability* for each point to belong to a cluster. This is particularly useful in applications like topic modeling, where a document may belong to multiple topics. The membership probability $p(x_j \in C_i)$ for a point x_j to cluster C_i is derived from the distance and density estimates, allowing for more flexible cluster assignments.

2.4.4 Bag-of-words

Before creating topic representations in BERTopic, it is necessary to select a technique that supports the algorithm's modular nature, and does not make assumptions about the data. When using HDBSCAN, we assume that clusters may vary in density and shape, indicating that techniques based on centroid models may not be suitable. The desired method should ideally make minimal assumptions about the cluster structures.

The process begins by combining all documents within a cluster into a single document, which then represents the entire cluster. Subsequently, the frequency of each word within this single document is counted, resulting in a bag-of-words representation that reflects the word frequencies at the cluster level rather than the individual document level. The adoption of a bag-of-words approach ensures that no assumptions are made about the density and shape of the clusters.

There are various approaches to constructing a bag-of-words representation, facilitated by the use of `CountVectorizer`. `CountVectorizer` allows us to control the number of tokens in each topic representation. For instance, single words like *game* or *team* may appear in a topic, but it can also be useful to include multi-word phrases, such as *hockey league*, which consist of two tokens. Additionally, some topics might include stop words, such as *he* or *the*, which are generally undesirable in topic representations as they provide little meaningful information. Removing these stop words is typically preferred to improve the quality of the topics.

To illustrate how topic representations can be constructed using a bag-of-words approach, let's consider an example where we have clustered documents related to sports. After applying HDBSCAN and merging the documents within each cluster into a single document, we can create a topic representation by counting the frequency of words and phrases within the merged document.

Table 2.1 shows a bag-of-words representation for a cluster related to hockey. The table includes both single words and multi-word phrases, and stop words have been removed to improve the quality of the representation.

2.4.5 Topic representation

From the generated bag-of-words representation, our goal is to identify what distinguishes one cluster from another. Specifically, we want to determine which words are characteristic of a particular cluster (e.g., cluster 1) but less common in other clusters. To achieve this, we need to modify the traditional TF-IDF such that it operates at the cluster level, treating clusters as topics instead of individual documents.

The classic TF-IDF [49] method combines term frequency and inverse document frequency to calculate a weight $W_{t,d}$ for term t in document d as follows:

$$W_{t,d} = tf_{t,d} \cdot \log\left(\frac{N}{df_t}\right)$$

Word/Phrase	Frequency
hockey	45
league	32
game	28
team	27
ice	25
hockey league	20
playoff	18
goals	15
players	14
national team	12
championship	10
tournament	9
world cup	8
goalie	7
penalty	6

Table 2.1: Example Bag-of-Words Representation for a Hockey-Related Cluster

Here, term frequency $tf_{t,d}$ represents the frequency of term t in document d, and inverse document frequency measures t's importance across documents, calculated by the logarithm of the ratio of the total number of documents N to the number of documents containing t.

BERTopic extends the TF-IDF concept to clusters of documents by introducing class-based TF-IDF (c-TF-IDF). In this approach, documents within a cluster are concatenated into a single document, and the TF-IDF formula is modified for cluster-level representation:

$$W_{t,c} = tf_{t,c} \cdot \log\left(1 + \frac{A}{tf_t}\right)$$

In this formula, term frequency $tf_{t,c}$ now models the frequency of term t within a cluster c, treated as a single document. The inverse document frequency is substituted with an inverse class frequency, which assesses the term's importance to a cluster. This is calculated by the logarithm of the average number of words per cluster A divided by the term's frequency tf_t across all clusters, with 1 added inside the logarithm to ensure positive values. This adaptation of TF-IDF to clusters allows us to model the importance of words in clusters instead of individual documents. Furthermore, by iteratively merging c-TF-IDF representations of less prevalent topics with their closest topics, the total number of topics can be reduced to meet a predefined threshold.

To illustrate this, consider the hockey-related cluster. Table 2.2 shows an example of how c-TF-IDF weights are calculated for several terms. As we can see from the table, terms like *hockey*, *ice*, and *playoff* receive high c-TF-IDF weights because they are frequent in the hockey cluster but less common in other clusters. Conversely, terms such as *game* and *team*, which are more evenly distributed across clusters, receive lower c-TF-IDF scores.

2.4.6 (Optional) Topic representation fine-tuning

After generating the c-TF-IDF representations, we obtain a collection of words that describe a collection of documents. c-TF-IDF is a method for quickly producing accurate topic representations. Nonetheless, the field of NLP is rapidly advancing, with frequent new developments. To make use of these developments, BERTopic offers the option to refine c-TF-IDF topics further using GPT [50, 51, 52], KeyBERT [35], spaCy [53], and other techniques, many of which are integrated within the BERTopic library. Users can also implement their own fine-tuning methods, allowing for a high degree of customization.

In particular, the topics generated through c-TF-IDF can be viewed as canditate topics, comprising a set of terms and representative documents. These can serve as a foundation for

Word/Phrase	Term Frequency in Cluster (TF)	Total Frequency (TF across all clusters)	Average Words per Cluster (A)	c-TF-IDF Weight
hockey	45	60	100	$45 \cdot \log\left(1 + \frac{100}{60}\right) \approx 22.95$
league	32	50	100	$32 \cdot \log\left(1 + \frac{100}{50}\right) \approx 22.40$
game	28	100	100	$28 \cdot \log\left(1 + \frac{100}{100}\right) = 8.40$
team	27	90	100	$27 \cdot \log\left(1 + \frac{100}{90}\right) \approx 2.97$
ice	25	25	100	$25 \cdot \log\left(1 + \frac{100}{25}\right) \approx 34.75$
hockey league	20	30	100	$20 \cdot \log\left(1 + \frac{100}{30}\right) \approx 24.00$
playoff	18	20	100	$18 \cdot \log\left(1 + \frac{100}{20}\right) \approx 30.60$
goals	15	40	100	$15 \cdot \log\left(1 + \frac{100}{40}\right) \approx 13.80$
players	14	80	100	$14 \cdot \log\left(1 + \frac{100}{80}\right) \approx 3.08$
national team	12	15	100	$12 \cdot \log\left(1 + \frac{100}{15}\right) \approx 22.20$

Table 2.2: Example c-TF-IDF Weights for Words in the Hockey Cluster

further refinement of topic representations. The availability of representative documents for each topic can be useful, as it enables fine-tuning on a reduced number of documents, thereby reducing computational demands. This makes the use of architectures such as large language models more viable in production environments, often resulting in shorter processing times compared to the steps of dimensionality reduction and clustering.

2.4.7 Evaluation metrics

Chapter 3

Methodology

And the second real chapter.

Chapter 4

Results

Results

Chapter 5

Conclusions and Recommendations

Write your conclusions here.
maybe split recommendations into separate small chapter

5.1 Recommendations

1. Keep looking for improved sub-models — embedding model and fine-tuning model and zero-shot model especially. These models can be tested with the automated evaluation metrics and with the GPT evaluation quickly, quick iterations. 2. Look into training a custom zero-shot model with a better base architecture — this could be a separate Master's project on its own. Look into Moritz Laurer's work on zero-shot learning.

Bibliography

[1] Stephan A. Curiskis, Barry Drake, Thomas R. Osborn, and Paul J. Kennedy. An evaluation of document clustering and topic modelling in two online social networks: Twitter and Reddit. *Information Processing & Management*, 57(2):102034, March 2020. ISSN 0306-4573. doi: 10.1016/j.ipm.2019.04.002. 1

[2] Michael J. Paul and Mark Dredze. Discovering Health Topics in Social Media Using Topic Models. *PLOS ONE*, 9(8):e103408, August 2014. ISSN 1932-6203. doi: 10.1371/journal.pone.0103408. 1

[3] Marco Pennacchiotti and Siva Gurumurthy. Investigating topic models for social media user recommendation. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11, pages 101–102, New York, NY, USA, March 2011. Association for Computing Machinery. ISBN 978-1-4503-0637-9. doi: 10.1145/1963192.1963244. 1

[4] Krishna Raj P M and Jagadeesh Sai D. Sentiment analysis, opinion mining and topic modelling of epics and novels using machine learning techniques. *Materials Today: Proceedings*, 51: 576–584, January 2022. ISSN 2214-7853. doi: 10.1016/j.matpr.2021.06.001. 1

[5] Carina Jacobi, Wouter van Atteveldt, and Kasper Welbers. Quantitative analysis of large amounts of journalistic texts using topic modelling. In *Rethinking Research Methods in an Age of Digital Journalism*. Routledge, 2018. ISBN 978-1-315-11504-7. 1

[6] Quintus Van Galen Nicholson, Bob. In Search of America: Topic modelling nineteenth-century newspaper archives. In *Journalism History and Digital Archives*. Routledge, 2020. ISBN 978-1-00-309884-3. 1

[7] Jani Marjanen, Elaine Zosa, Simon Hengchen, Lidia Pivovarova, and Mikko Tolonen. Topic modelling discourse dynamics in historical newspapers, November 2020. 1

[8] Raquel Silveira, Carlos G O Fernandes, João A Monteiro Neto, Vasco Furtado, and Ernesto Pimentel Filho. Topic Modelling of Legal Documents via LEGAL-BERT. 1

[9] James O'Neill, Cecile Robin, Leona O'Brien, and Paul Buitelaar. An analysis of topic modelling for legislative texts. 2016. ISSN 1613-0073. 1

[10] Claus Boye Asmussen and Charles Møller. Smart literature review: A practical topic modelling approach to exploratory literature review. *Journal of Big Data*, 6(1):93, October 2019. ISSN 2196-1115. doi: 10.1186/s40537-019-0255-7. 1

[11] Karim El Mokhtari, Mucahit Cevik, and Ayşe Başar. Using Topic Modelling to Improve Prediction of Financial Report Commentary Classes. In Cyril Goutte and Xiaodan Zhu, editors, *Advances in Artificial Intelligence*, Lecture Notes in Computer Science, pages 201–207, Cham, 2020. Springer International Publishing. ISBN 978-3-030-47358-7. doi: 10.1007/978-3-030-47358-7_19. 1

[12] Silvia García-Méndez, Francisco de Arriba-Pérez, Ana Barros-Vila, Francisco J. González-Castaño, and Enrique Costa-Montenegro. Automatic detection of relevant information, predictions and forecasts in financial news through topic modelling with Latent Dirichlet Allocation. *Applied Intelligence*, 53(16):19610–19628, August 2023. ISSN 1573-7497. doi: 10.1007/s10489-023-04452-4. 1

[13] David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. 1, 3, 4, 29

[14] Aly Abdelrazek, Yomna Eid, Eman Gawish, Walaa Medhat, and Ahmed Hassan Yousef. Topic modeling algorithms and applications: A survey. *Information Systems*, 112:102131, October 2022. doi: 10.1016/j.is.2022.102131. 1

[15] Rob Churchill and Lisa Singh. The Evolution of Topic Modeling. *ACM Computing Surveys*, 54(10s):1–35, January 2022. ISSN 0360-0300, 1557-7341. doi: 10.1145/3507900. 1, 30

[16] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, June 2014. ISSN 1931-0145, 1931-0153. doi: 10.1145/2641190.2641198. 1

[17] Taniya Das. Openml/scripts. https://github.com/openml/scripts/tree/main. 2

[18] Wolfram Data Repository: Computable Access to Curated Data. https://datarepository.wolframcloud.com/, . 2

[19] Farial Shahnaz, Michael W. Berry, V. Paul Pauca, and Robert J. Plemmons. Document clustering using nonnegative matrix factorization. *Information Processing & Management*, 42 (2):373–386, March 2006. ISSN 0306-4573. doi: 10.1016/j.ipm.2004.11.005. 5, 30

[20] Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 745–754, New York, NY, USA, October 2011. Association for Computing Machinery. ISBN 978-1-4503-0717-8. doi: 10.1145/2063576.2063686. 5, 30

[21] Xiaohui Yan, Jiafeng Guo, Shenghua Liu, Xueqi Cheng, and Yanfeng Wang. Learning Topics in Short Texts by Non-negative Matrix Factorization on Term Correlation Matrix. In *Proceedings of the 2013 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 749–757. Society for Industrial and Applied Mathematics, May 2013. ISBN 978-1-61197-262-7. doi: 10.1137/1.9781611972832.83. 5, 30

[22] Dimo Angelov. Top2Vec: Distributed representations of topics, August 2020. 7, 32

[23] Quoc V. Le and Tomas Mikolov. Distributed Representations of Sentences and Documents, May 2014. 7, 32

[24] Radim Řehůřek and Petr Sojka. *Software Framework for Topic Modelling with Large Corpora*. University of Malta, May 2010. ISBN 978-2-9517408-6-0. 7

[25] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Universal Sentence Encoder, April 2018. 7

[26] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, August 2019. 7, 9, 10, 33

[27] Jey Han Lau and Timothy Baldwin. An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation, July 2016. 8

[28] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, September 2020. 8, 9, 11

[29] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-Based Clustering Based on Hierarchical Density Estimates. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 160–172, Berlin, Heidelberg, 2013. Springer. ISBN 978-3-642-37456-2. doi: 10.1007/978-3-642-37456-2_14. 8, 13, 14

[30] Leland McInnes and John Healy. Accelerated Hierarchical Density Based Clustering. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 33–42, November 2017. doi: 10.1109/ICDMW.2017.12. 8, 13

[31] Leland McInnes, John Healy, and Steve Astels. Hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205, March 2017. ISSN 2475-9066. doi: 10.21105/joss.00205. 8, 13

[32] Maarten Grootendorst. BERTopic: Neural topic modeling with a class-based TF-IDF procedure, March 2022. 9, 33

[33] Maarten P. Grootendorst. The Algorithm - BERTopic. https://maartengr.github.io/BERTopic/algorithm/algorithm.html. 9

[34] Hervé Abdi and Lynne J. Williams. Principal component analysis. *WIREs Computational Statistics*, 2(4):433–459, July 2010. ISSN 1939-5108. doi: 10.1002/wics.101. 9, 11

[35] Maarten Grootendorst. MaartenGr/KeyBERT, February 2024. 9, 16

[36] Nils Reimers and Iryna Gurevych. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation, October 2020. 9

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 9, 10, 33

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. 10, 33

[39] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive Text Embedding Benchmark, March 2023. 10

[40] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach, July 2019. 10, 33

[41] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the Surprising Behavior of Distance Metrics in High Dimensional Space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, Lecture Notes in Computer Science, pages 420–434, Berlin, Heidelberg, 2001. Springer. ISBN 978-3-540-44503-6. doi: 10.1007/3-540-44503-X_27. 11

[42] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When Is "Nearest Neighbor" Meaningful? In Catriel Beeri and Peter Buneman, editors, *Database Theory — ICDT'99*, Lecture Notes in Computer Science, pages 217–235, Berlin, Heidelberg, 1999. Springer. ISBN 978-3-540-49257-3. doi: 10.1007/3-540-49257-7_15. 11

[43] Divya Pandove, Shivan Goel, and Rinkl Rani. Systematic Review of Clustering High-Dimensional and Large Datasets. *ACM Transactions on Knowledge Discovery from Data*, 12 (2):16:1–16:68, January 2018. ISSN 1556-4681. doi: 10.1145/3132088. 11

[44] Michael Steinbach, Levent Ertöz, and Vipin Kumar. The Challenges of Clustering High Dimensional Data. In Luc T. Wille, editor, *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition*, pages 273–309. Springer, Berlin, Heidelberg, 2004. ISBN 978-3-662-08968-2. doi: 10.1007/978-3-662-08968-2_16. 11

[45] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. | Journal of Machine Learning Research | EBSCOhost. https://openurl.ebsco.com/contentitem/gcd:36099312?sid=ebsco:plink:crawler&id=ebsco:gcd:36099312, November 2008. ISSN 1532-4435. 11

[46] Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection. *ACM Trans. Knowl. Discov. Data*, 10(1):5:1–5:51, July 2015. ISSN 1556-4681. doi: 10.1145/2733381. 13

[47] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. 13

[48] Mebarka Allaoui, Mohammed Lamine Kherfi, and Abdelhakim Cheriet. Considerably Improving Clustering Algorithms Using UMAP Dimensionality Reduction Technique: A Comparative Study. In Abderrahim El Moataz, Driss Mammass, Alamin Mansouri, and Fathallah Nouboud, editors, *Image and Signal Processing*, Lecture Notes in Computer Science, pages 317–325, Cham, 2020. Springer International Publishing. ISBN 978-3-030-51935-3. doi: 10.1007/978-3-030-51935-3_34. 13

[49] Thorsten Joachims. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 143–151, San Francisco, CA, USA, July 1997. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-486-5. 15

[50] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. . 16, 33

[51] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. . 16, 33

[52] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, July 2020. 16, 33

[53] Explosion/spaCy: Industrial-strength Natural Language Processing (NLP) in Python. https://github.com/explosion/spaCy/tree/master, . 16

[54] Scott Deerwester, Susan T Dumais, George W Furnas, LANDAUEŔT Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Indexing by latent semantic analysis*, 41(6): 391–407, 1990. ISSN 0002-8231. 29

[55] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57, Berkeley California USA, August 1999. ACM. ISBN 978-1-58113-096-6. doi: 10.1145/312624.312649. 29

[56] Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun, and Tom Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning*, 39(2): 103–134, May 2000. ISSN 1573-0565. doi: 10.1023/A:1007692713085. 29

[57] Pierre Simon Laplace (marquís de). *Théorie analytique des probabilités*. Courcier, 1814. 29

[58] Hamed Jelodar, Yongli Wang, Chi Yuan, Xia Feng, Xiahui Jiang, Yanchao Li, and Liang Zhao. Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications*, 78(11):15169–15211, June 2019. ISSN 1573-7721. doi: 10.1007/s11042-018-6894-4. 29

[59] Yee Teh, Michael Jordan, Matthew Beal, and David Blei. Sharing Clusters among Related Groups: Hierarchical Dirichlet Processes. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. 29

[60] John Lafferty and David Blei. Correlated Topic Models. In *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2005. 29

[61] David M. Blei and John D. Lafferty. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 113–120, New York, NY, USA, June 2006. Association for Computing Machinery. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143859. 29

[62] Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006. 29

[63] Ramesh M. Nallapati, Susan Ditmore, John D. Lafferty, and Kin Ung. Multiscale topic tomography. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 520–529, New York, NY, USA, August 2007. Association for Computing Machinery. ISBN 978-1-59593-609-7. doi: 10.1145/1281192.1281249. 29

[64] Chong Wang, David Blei, and David Heckerman. Continuous Time Dynamic Topic Models. https://arxiv.org/abs/1206.3298v2, June 2012. 29

[65] Tomoharu Iwata, Shinji Watanabe, Takeshi Yamada, and Naonori Ueda. *Topic Tracking Model for Analyzing Consumer Purchase Behavior*. January 2009. 29

[66] Arindam Banerjee and Sugato Basu. Topic Models over Text Streams: A Study of Batch and Online Unsupervised Learning. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*, Proceedings, pages 431–436. Society for Industrial and Applied Mathematics, April 2007. ISBN 978-0-89871-630-6. doi: 10.1137/1.9781611972771.40. 29

[67] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 569–577, New York, NY, USA, August 2008. Association for Computing Machinery. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401960. 29

[68] Limin Yao, David Mimno, and Andrew McCallum. Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 937–946, New York, NY, USA, June 2009. Association for Computing Machinery. ISBN 978-1-60558-495-9. doi: 10.1145/1557019.1557121. 29

[69] Matthew Hoffman, Francis Bach, and David Blei. Online Learning for Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. 29

[70] Jon Mcauliffe and David Blei. Supervised Topic Models. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. 30

[71] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic model for short texts. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, pages 1445–1456, New York, NY, USA, May 2013. Association for Computing Machinery. ISBN 978-1-4503-2035-1. doi: 10.1145/2488388.2488514. 30

[72] Xiaojun Quan, Chunyu Kit, Yong Ge, and Sinno Jialin Pan. Short and Sparse Text Topic Modeling via Self-Aggregation. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pages 2270–2276. AAAI Press/International Joint Conferences on Artificial Intelligence, July 2015. 30

[73] Jianhua Yin and Jianyong Wang. A dirichlet multinomial mixture model-based approach for short text clustering. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 233–242, New York, NY, USA, August 2014. Association for Computing Machinery. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623715. 30

[74] Chenliang Li, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. Topic Modeling for Short Texts with Auxiliary Word Embeddings. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 165–174, New York, NY, USA, July 2016. Association for Computing Machinery. ISBN 978-1-4503-4069-4. doi: 10.1145/2911451.2911499. 30

[75] Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving Topic Models with Latent Feature Word Representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, December 2015. ISSN 2307-387X. doi: 10.1162/tacl_a_00140. 30, 31

[76] Ximing Li, Yue Wang, Ang Zhang, Changchun Li, Jinjin Chi, and Jihong Ouyang. Filtering out the noise in short text topic modeling. *Information Sciences*, 456:83–96, August 2018. ISSN 0020-0255. doi: 10.1016/j.ins.2018.04.071. 30

[77] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on Twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, MDMKDD '10, pages 1–10, New York, NY, USA, July 2010. Association for Computing Machinery. ISBN 978-1-4503-0220-3. doi: 10.1145/1814245.1814249. 31

[78] Henrique F. de Arruda, Luciano da F. Costa, and Diego R. Amancio. Topic segmentation via community detection in complex networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 26(6):063120, June 2016. ISSN 1054-1500, 1089-7682. doi: 10.1063/1.4954215. 31

[79] Rob Churchill, Lisa Singh, and Christo Kirov. A Temporal Topic Model for Noisy Mediums. In Dinh Phung, Vincent S. Tseng, Geoffrey I. Webb, Bao Ho, Mohadeseh Ganji, and Lida Rashidi, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 42–53, Cham, 2018. Springer International Publishing. ISBN 978-3-319-93037-4. doi: 10.1007/978-3-319-93037-4_4. 31

[80] R. Churchill and L. Singh. Percolation-based topic modeling for tweets. *WISDOM 2020 : The 9th KDD Workshop on Issues of Sentiment Discovery and Opinion Mining*, August 2020. 31

[81] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey, May 2023. 31

[82] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A Neural Probabilistic Language Model. In *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2000. 31

[83] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013. 31

[84] Jipeng Qiang, Ping Chen, Tong Wang, and Xindong Wu. Topic Modeling over Short Texts by Incorporating Word Embeddings. In Jinho Kim, Kyuseok Shim, Longbing Cao, Jae-Gil Lee, Xuemin Lin, and Yang-Sae Moon, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 363–374, Cham, 2017. Springer International Publishing. ISBN 978-3-319-57529-2. doi: 10.1007/978-3-319-57529-2_29. 31

[85] Stefan Bunk and Ralf Krestel. WELDA: Enhancing Topic Models by Incorporating Local Word Context. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, JCDL '18, pages 293–302, New York, NY, USA, May 2018. Association for Computing Machinery. ISBN 978-1-4503-5178-2. doi: 10.1145/3197026.3197043. 31

[86] Ximing Li, Jiaojiao Zhang, and Jihong Ouyang. Dirichlet Multinomial Mixture with Variational Manifold Regularization: Topic Modeling over Short Texts. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7884–7891, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33017884. 32

[87] Yishu Miao, Lei Yu, and Phil Blunsom. Neural Variational Inference for Text Processing. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1727–1736. PMLR, June 2016. 32

[88] Christopher E. Moody. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec, May 2016. 32

[89] Yuan Zuo, Junjie Wu, Hui Zhang, Hao Lin, Fei Wang, Ke Xu, and Hui Xiong. Topic Modeling of Short Texts: A Pseudo-Document View. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 2105–2114, New York, NY, USA, August 2016. Association for Computing Machinery. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939880. 32

[90] Paulo Bicalho, Marcelo Pita, Gabriel Pedrosa, Anisio Lacerda, and Gisele L. Pappa. A general framework to expand short text for topic modeling. *Information Sciences*, 393:66–81, July 2017. ISSN 0020-0255. doi: 10.1016/j.ins.2017.02.007. 32

[91] Felipe Viegas, Sérgio Canuto, Christian Gomes, Washington Luiz, Thierson Rosa, Sabir Ribas, Leonardo Rocha, and Marcos André Gonçalves. CluWords: Exploiting Semantic Word Clustering Representation for Enhanced Topic Modeling. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pages 753–761, New York, NY, USA, January 2019. Association for Computing Machinery. ISBN 978-1-4503-5940-5. doi: 10.1145/3289600.3291032. 32

[92] Felipe Viegas, Washington Cunha, Christian Gomes, Antônio Pereira, Leonardo Rocha, and Marcos Goncalves. CluHTM - Semantic Hierarchical Topic Modeling based on CluWords. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8138–8150, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.724. 32

[93] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. Topic Modeling in Embedding Spaces. *Transactions of the Association for Computational Linguistics*, 8:439–453, July 2020. ISSN 2307-387X. doi: 10.1162/tacl_a_00325. 32

[94] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. The Dynamic Embedded Topic Model, October 2019. 32

[95] Laure Thompson and David Mimno. Topic Modeling with Contextualized Word Representation Clusters. https://arxiv.org/abs/2010.12626v1, October 2020. 33

[96] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. 33

Appendix A

Broad Literature Review

A.1 Early Foundations and Probabilistic Models

The origins of topic models can be traced back to the early 1990s with the development of Latent Semantic Indexing (LSI), introduced by Deerwester et al. [54]. LSI creates a word-document matrix from a given vocabulary and a collection of documents. This matrix records how frequently each word in the vocabulary appears in the documents. The key step in LSI is the use of singular value decomposition (SVD). SVD compresses the dimensionality of documents, while still maintaining the meaning of the words.

LSI served as a precursor to topic models. In 1999, Hofmann [55] introduced Probabilistic Latent Semantic Indexing (pLSI). In pLSI, the SVD component of LSI was replaced with a generative data model known as an *aspect model*. This change enabled the training of the model using an expectation maximization algorithm. Instead of deriving topics through SVD, Hofmann's approach allowed topics to emerge as probabilistic mixtures of words. These mixtures were based on the joint probabilities of words and documents. This probabilistic framework marked a departure from the earlier matrix factorization approach of LSI and laid the groundwork for more advanced topic modeling techniques.

In 2000, Nigam et al. [56] explored how to integrate unlabeled data into text classification, leading to the development of the Dirichlet Multinomial Mixture (DMM). They employed expectation maximization in conjunction with the Dirichlet distribution [57]. The Dirichlet distribution is a multivariate extension of the Beta distribution. Unlike the Beta distribution, which is defined by parameters α and β, the Dirichlet distribution is characterized by a parameter k. This k represents the number of dimensions in the Dirichlet distribution. These dimensions collectively form a normalized probability distribution, which is adjusted using the parameter α. The Dirichlet distribution is particularly suitable for topic modeling, as each topic can be represented as one of the k dimensions in the distribution.

A.2 Latent Dirichlet Allocation

The term "topic model" was coined by Blei et al. [13] in their seminal 2001 paper on Latent Dirichlet Allocation (LDA). This work adopted the use of a generative model in a similar way to pLSI, but bases its model on the Dirichlet distribution. LDA introduced the concept that documents can be associated with multiple topics, rather than a single topic, as was the case with previous models. Furthermore, a key advancement of LDA was its capacity to be applied to new, unseen documents. The influence of LDA in topic modeling has been substantial, leading to numerous works improving upon LDA, and on the creation of LDA variants adapted for various tasks [58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69].

Most topic models, including LDA, are unsupervised. However, for some tasks, supervision is required. To address this, a variant of LDA named Supervised Latent Dirichlet Allocation

(sLDA) was developed, which transforms LDA into a supervised model [70]. sLDA extends LDA by associating a response variable with each document. The aim of sLDA is to discover latent topics that are both descriptive of the documents and predictive of the response variable. It achieves this by using a generalized linear model to connect the response variable with the topic proportions in each document.

The evolution of topic models has been influenced by changes in the types of data being analyzed, as noted by Churchill and Singh [15]. In the 2000s, the primary data sources for topic models were scientific articles, books, and newspapers. However, the landscape has shifted in recent years, with an increasing focus on digital and social media content such as tweets, blog posts, and Reddit posts.

Despite the change in data types, LDA and its variants have remained prevalent in the field. These models have been recognized as best practices for various data types and continue to be relevant. However, the field of topic modeling has progressed to include newer models beyond LDA.

A.3 NMF

Non-negative Matrix Factorization (NMF) represents one of the newer advancements in topic modeling, as highlighted by Churchill and Singh [15]. NMF is a technique where an original matrix, consisting of non-negative values, is decomposed into two distinct matrices. The fundamental principle of NMF is that the product of these two matrices approximates the original matrix. This decomposition is a form of dimensionality reduction. The large original matrix typically represents a set of documents, with each document being a vector of words. The two resultant matrices in NMF are the topic-word matrix and the topic-document matrix. The topic-word matrix shows the association between topics and words, while the topic-document matrix shows the relationship between topics and individual documents.

The application of NMF in topic modeling was first demonstrated by Shahnaz et al. [19]. They showcased the potential of NMF as a tool for extracting topics from a collection of documents. Following this, [20] expanded the application of NMF to temporal topic models. Yan et al. [21] later augmented NMF by replacing the document-term matrix with a term correlation matrix to detect topics in short texts.

In their 2013 study, Yan et al. [71] presented the Biterm Topic Model (BTM). This model is specifically tailored for analyzing short texts, like tweets and social media updates. Instead of focusing on patterns within entire documents, BTM works by identifying and analyzing pairs of words, termed 'biterms'. These biterms are formed based on a distribution of topics and words.

In 2015, Quan et al. [72] introduced the Self-Aggregating Topic Model (SATM), aimed at improving the topic modeling of short texts. SATM comprises two steps. Initially, it runs LDA on short texts. Subsequently, it uses the topics generated in the first step to create longer pseudo-texts. A pseudo-text in this context refers to the concatenation of shorter documents into a single, longer document.

In 2014, Yin and Wang [73] augmented the Dirichlet Multinomial Mixture (DMM) by introducing the Gibbs Sampling DMM. This adaptation incorporated the Gibbs sampling algorithm, specifically aimed at more effective modeling of short texts. This innovative approach to the established DMM model paved the way for further advanced variations of DMM.

Following this advancement, Li et al. [74] proposed two models: the Generalized Polya Urn Dirichlet Multinomial Mixture (GPUDMM) and the Generalized Polya Urn Poisson-based Dirichlet Multinomial Mixture (GPUPDMM). These models, similar to the approach by Nguyen et al. [75], integrate word embeddings into the classic DMM model. In the GPU approach, when a word is selected, a copy of that word along with similar words are added back to the topic. This mechanism leads to clusters of similar words rising to the forefront of a topic, thus creating topics with more coherent and related sets of words. Regarding the DMM aspect of their models, Li et al. directly draw from Yin and Wang [73]'s GSDMM.

Building upon DMM, Li et al. [76] developed the Common Semantics Topic Model (CSTM). In this model, they introduced a concept known as 'common topics', designed to capture words

that are prevalent across all topics. CSTM then creates topics by combining words from a single specific topic with words from these common topics.

A.4 Graph-based Models

Graph-based models are another approach to topic modeling that followed LDA. In these models, words are represented as nodes in a graph, with their co-occurrences indicated by weighted edges. This method diverges from previous generative models by not assuming any underlying topic distribution, which facilitates the discovery of topics of varying sizes.

Cataldi et al. [77] were among the first to implement a graph-based model using a directed graph to detect emerging topics. Subsequently, de Arruda et al. [78] developed a topic model known as Topic Segmentation (TS), which was based on an undirected graph. In 2018, the Topic Flow Model (TFM) was introduced by Churchill et al. [79], applying graph-based methods to track the evolution of topics over time. Following this, Churchill and Singh [80] proposed the Percolation-based Topic Model (PTM), a graph-based model designed to detect topics within noisy datasets.

A.5 Word Embedding Models

The integration of natural language processing (NLP) techniques into topic models marks a recent advancement in the field, diverging from earlier statistical models like LDA. A key development in this area has been the use of pre-trained NLP models to augment the capabilities of unsupervised topic models.

According to Almeida and Xexéo [81], the most prominent form of NLP models employed in this context is word embedding spaces. The inception of word embeddings can be traced back to the early 2000s with the work of Bengio et al. [82], who proposed a neural model for learning distributed representations of words.

A seminal study in the field of NLP for creating word embeddings is Word2Vec by Mikolov et al. [83]. This study demonstrated the efficacy of word vectors in identifying semantically similar words. Word2Vec itself comprises two distinct architectures that facilitate the learning of high-quality word embeddings: Skip-gram and Continuous Bag of Words (CBOW).

Nguyen et al. [75] later enhanced the LDA and DMM models by incorporating word embeddings, resulting in two new models: Latent Feature LDA (LF-LDA) and Latent Feature DMM (LF-DMM). In these models, they added a word embedding component to the topic-word distribution of LDA and DMM. LF-LDA and LF-DMM maintain the original structure of LDA and DMM, but integrate a word embedding for each word in their distributions. When generating words for a document, the model can select words either from the topic's distribution or from the word embedding associated with that topic. This approach effectively enlarges the selection pool of words. The addition of the word embedding component improves the models' performance, especially when dealing with short texts.

Qiang et al. [84] developed the Embedding-based Topic Model (ETM), which leverages Word2Vec and introduces a new distance metric known as Word Mover's Distance (WMD). WMD calculates the minimum cumulative distance that words in one document need to travel to match the closest corresponding words in another document. After computing WMD, ETM combines short documents into longer pseudo-texts. Subsequently, LDA is applied to these pseudo-texts to determine topic assignments. The model then constructs an undirected graph to create a Markov Random Field. In this framework, similar words appearing in the same pseudo-text are more likely to be assigned to the same topic.

Bunk and Krestel [85] proposed another enhancement to LDA, termed Word Embedding LDA (WELDA). This approach involves integrating a pretrained word embedding model with a slightly modified version of LDA.

Li et al. [86] further adapted the Dirichlet Multinomial Mixture (DMM) model to better suit short texts, creating the Laplacian DMM (LapDMM). This model integrates variational manifold regularization to maintain the local neighborhood structure inherent in short texts. Before training LapDMM, a graph is constructed to measure the distances between documents, identifying their nearest neighbors. This graph's Laplacian matrix is then utilized as a constraint in the topic assignment process. This ensures that documents assigned to the same topic contain words that are located in similar neighborhoods within the graph. To calculate the distances between documents, the authors employ Word2Vec along with WMD.

In 2016, Miao et al. [87] introduced the Neural Variational Document Model (NVDM), which employs a neural network to perform a multinomial logistic regression. This process is used to generate a word embedding for each document.

In the same year, Moody [88] developed Ida2Vec, a model that integrates Word2Vec with the traditional LDA model. Ida2Vec generates vectors for both documents and words, enabling the measurement of similarity between documents as well as between documents and words or phrases. In this model, each topic is represented as a vector in the same space as the word and document vectors. The resultant topic vector can then be compared with word vectors to identify words most closely related to the topic.

Le and Mikolov later extended their Word2Vec model by introducing Doc2Vec [23]. Doc2Vec extends Word2Vec by introducing a novel framework that allows for the generation of vector representations not just for words, but for larger blocks of text such as sentences, paragraphs, or entire documents. While Word2Vec models (both Skip-gram and Continuous Bag of Words (CBOW)) are efficient at capturing the semantic similarity between words based on their context, they do not directly provide a method for aggregating these word vectors into meaningful representations for larger texts. Doc2Vec addresses this limitation through its architecture, enabling the capture of document-level context.

In 2020, Angelov [22] introduced Top2Vec. Top2Vec extends Word2Vec and Doc2Vec by leveraging their distributed representations of words and documents to model topics. It utilizes Doc2Vec to create semantic embeddings of documents and words, embedding them jointly in the same space, where the proximity between document and word vectors represents semantic similarity. This joint embedding allows for the identification of dense clusters of document vectors, assumed to represent topics. Top2Vec calculates topic vectors as centroids of these clusters and identifies topic words by finding the closest word vectors to each topic vector. This approach provides a more nuanced understanding of topics by exploiting the semantic relationships inherent in the distributed representations of words and documents.

In 2016, Zuo et al. [89] improved their original STM model by introducing the Pseudo-document-based Topic Model (PTM). PTM aggregates multiple short texts into a single pseudo-document. This approach results in a condensed word co-occurrence matrix, which in turn leads to a more accurate approximation of the topics.

In 2017, Bicalho et al. [90] introduced the Distributed representation-based expansion (DREx) technique. This method involves expanding a given document by incorporating the closest n-grams found in the embedding space that are similar to the n-grams present in the document.

Viegas et al. introduced two topic models, CluWords [91] and CluHTM [92], both of which utilize clusters of words and the Term Frequency-Inverse Document Frequency (TF-IDF) method to generate topics. TF-IDF is a widely used metric in text mining that indicates the relevance of a word to a document within a collection or corpus. The core concept in these models is a 'CluWord', which is essentially a cluster of words defined within an embedding space. The process begins by determining CluWords for each word in the vocabulary. Then, in each document, every word is replaced by its corresponding CluWord. Following this replacement, the TF-IDF values of the CluWords are calculated. CluHTM extends the concept of CluWords by combining it with NMF to facilitate hierarchical topic modeling.

Dieng et al. introduced two models: the Embedded Topic Model (ETM) [93] and the Dynamic Embedded Topic Model (D-ETM) [94]. In both models, topics and words are represented within an embedding space. Like LDA, ETM draws a topic for each document, but it diverges from LDA by using the logistic-normal distribution instead of the Dirichlet distribution. For each word

in a document, ETM assigns a topic, and then the observed word is drawn based on this topic assignment. This means that words are selected from their embeddings rather than based on their proximity to other words in the document. The D-ETM model extends this concept by adding a time-varying component to the framework. It runs the generative process at each time step, maintaining k topics at each step, but all of these topics are still projected onto the same embedding space.

A.6 Transformer-based models

The transformer model [37], introduced by Vaswani et al., marked another seminal step in the field of NLP. It is an architecture that significantly improves upon the efficiency and effectiveness of previous models for machine translation and other sequence-to-sequence tasks. Groundbreaking for its exclusive use of attention mechanisms, the transformer eliminates the need for recurrence and convolutions. It obviates the sequential data processing inherent in recurrent neural networks (RNNs) and the fixed receptive fields of convolutional neural networks (CNNs), enabling much greater parallelization of computation. This architecture sets new state-of-the-art benchmarks on translation tasks, demonstrating its superior ability to handle long-range dependencies within text. The transformer model has since influenced the development of numerous NLP models and frameworks, marking a pivotal shift in the approach to sequence modeling and machine learning. Its relevance to topic modeling is indirect but significant. Transformer models process text in a way that captures deeper semantic meanings, which can be leveraged for identifying coherent topics in large text corpora.

Following Vaswani et al. [37], OpenAI introduced the GPT-1, GPT-2 and GPT-3 models [50, 51, 52]. These models set state-of-the-art results, primarily through increases in the size of the training datasets together with an increased model size (parameter count).

In their 2019 work, Devlin et al. [38] introduced BERT (Bidirectional Encoder Representations from Transformers), a model leveraging the transformer architecture to pre-train deep bidirectional representations from unlabeled text. Unlike the original transformer by Vaswani et al. [37], which processes text in a unidirectional manner, BERT enhances understanding by evaluating both left and right contexts across all layers, achieving a more comprehensive grasp of language nuances.

There were a few seminal papers on BERT after Devlin et al.'s paper, namely RoBERTa [40] and Sentence-BERT (SBERT) [26].

In their research, the authors identified that the original BERT model had not been fully optimized in its training regimen. To address this, RoBERTa was developed, refining BERT's training by eliminating the next-sentence prediction objective, utilizing larger mini-batches and higher learning rates, and extending training over a substantially larger dataset. These strategic enhancements markedly boosted RoBERTa's performance, setting new benchmarks across a wide spectrum of NLP tasks.

In the former paper, the authors noticed that the original BERT model was undertrained, and so used RoBERTa to refine BERT's training process by removing the next-sentence prediction objective, using larger mini-batches and learning rates, and training on much more data. These optimizations lead to significantly improved performance across a variety of NLP benchmarks.

Sentence-BERT, proposed by Reimers and Gurevych [26], adapts BERT for efficient computation of sentence embeddings. By using a siamese network structure, SBERT generates embeddings that can be compared using cosine similarity, facilitating tasks like semantic textual similarity assessment and clustering with reduced computational resources and time.

In the context of topic modeling, Thompson and Mimno [95] later proposed using BERT [38] to produce topics. The authors use k-means to cluster tokens observed in the data set based on their contextual vectors drawn from BERT. This clustering task differs from previous models such as GloVe [96] and Word2Vec which are context-free embedding spaces with a single embedding representation for each word.

Grootendorst [32] later devised BERTopic, which is a state-of-the-art topic model that is based on the clustering idea. BERTopic employs pre-trained transformer models for creating document

embeddings, clusters these embeddings, and utilizes a class-based variation of TF-IDF, termed c-TF-IDF, for generating topic representations. This method ensures the generation of coherent topics and remains competitive in benchmarks against both traditional and modern clustering-based topic modeling approaches.