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

Topic Modeling Of OpenML Dataset Descriptions

Master Thesis

Ivan Germanov

Supervisors:
prof. Joaquin Vanschoren
MSc Taniya Das
prof. Alexander Nolte

1.1

Eindhoven, December 2024

Abstract

This thesis presents a novel approach to automatically generate tags for OpenML dataset descriptions using topic modeling techniques. Effective dataset tagging is important for organizing datasets on the OpenML platform [1], improving their discoverability, and enabling users to quickly identify relevant datasets for their needs. While OpenML datasets are currently categorized using both manual tagging and a semantic tagging approach that uses *GPT-3.5-turbo* to automatically assign predefined tags based on dataset descriptions, many datasets still lack semantic tags that are sufficiently informative and readable by humans. The current semantic tagging approach also has limitations as it does not leverage additional metadata beyond the dataset descriptions.

We develop a modular pipeline that extends the BERTopic model [2] with advanced language models and zeroshot text classification to automatically generate high-quality, human-readable tags. Through an exploratory data analysis, we first investigate OpenML dataset descriptions and augment them with additional metadata and contextual information. We then develop our pipeline, which combines state-of-the-art embedding models, specifically the *Salesforce/SFR-Embedding-2_R* [3] model found on the MTEB Benchmark [4], UMAP for dimensionality reduction [5], HDBSCAN for clustering [6, 7, 8], and fine-tuning with a large language model, *Llama-3-70b* [9] , combined with zeroshot text classification using the *MoritzLaurer/deberta-v3-large-zeroshot-v2.0* model [10]. The modular nature of our approach ensures it can evolve alongside advances in language models and embedding models. We also explore a cost-effective configuration that simplifies the pipeline and utilizes a smaller LLM (GPT-4o-mini [11]).

We evaluate our model using both automated metrics and human evaluation. Our automated evaluation reveals that the proposed model achieves a combined NPMI and diversity score of 0.779, outperforming all baseline models, including LDA [12], NMF [13, 14, 15], Top2Vec [16], and CTM [17, 18] on this metric, while achieving the highest diversity score of 0.808. A human evaluation study with 21 participants demonstrates that our model generates tags that are more relevant (mean score of 3.63) and provide better coverage (mean score of 4.46) compared to the baseline approach (2.39 and 2.65 respectively), while maintaining a good balance between specific and general tags (SD 1.40). In an intruder detection task, our model achieved a 95.2% success rate, compared to 42.1% for the baseline, and 100% for human-generated tags. These findings are further validated through a large-scale automated evaluation using *GPT-4-mini* across the entire OpenML dataset, where the LLM successfully identified 88.3% of the intruders and rating the tags for relevance (mean 4.11, SD 0.98), generality (mean 3.29, SD 0.87), and coverage (mean 3.72).

Our approach provides OpenML with an automated, scalable solution for dataset tagging that produces high-quality tags approaching human-level performance. To the best of our knowledge, this is the first approach to combine a modular topic modeling pipeline like BERTopic with advanced techniques such as prompt-based fine-tuning with LLMs and zeroshot classification for automated dataset tagging. The techniques we developed could be applied to similar problems in other domains where automated categorization of content is needed. While this work demonstrates significant improvements in automated dataset tagging, we also acknowledge the limitations of using LLMs, particularly regarding computational cost and potential biases. Our method of combining traditional topic modeling with modern language models and zeroshot text classification represents a novel contribution to the field of topic modeling itself.

Keywords: topic modeling, machine learning, natural language processing, dataset tagging, OpenML, BERTopic

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 [19, 20, 21], books [22], newspapers [23, 24, 25], legal documents [26, 27], research papers [28] and financial reports [29, 30], to name a few. Topic models can take a large corpus of documents as input and extract the latent topics present in the corpus [12]. A topic refers to a recurring pattern of words (terms) or phrases that commonly occur together in a set of documents [31]. 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 [32] 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.

The concept of organizing and tagging document collections has been implemented in various platforms, including OpenML [1], a collaborative platform for machine learning researchers. OpenML facilitates global collaboration by allowing users to present 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 structured and uniformly formatted metadata 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 both manual tagging and a semantic tagging approach that uses *GPT-3.5-turbo* to automatically assign predefined tags based on dataset descriptions. 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 modeling 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 [33]. 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 [34].

The main goal of this research is to explore the potential of unsupervised topic modeling when applied to the dataset descriptions combined with other available metadata on the OpenML platform. By extracting topics from the descriptions, we can use the terms in the topics as tags for the datasets. A tag is a descriptive keyword or phrase that characterizes the dataset's content or properties. For example, a dataset containing medical images might be tagged with terms like *Healthcare*, *Medicine*, or *Image Classification*.

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 how do they impact 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 advantages and disadvantages involved in their use?** Investigate existing topic modeling techniques and assess their suitability for extracting topics from the dataset descriptions, while considering advantages and disadvantages such as quality of extracted topics and terms, computational cost, and scalability. Explore strategies to ensure that the generated tags can be continuously refined and improved as new topic modeling approaches and advancements become available.

- **RQ3 — How to objectively optimize the quality of the topics and terms?** 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. Use optimization techniques to improve the quality of the topics and terms extracted by the model.

- **RQ4 — How to evaluate the tags from a human perspective?** Define and implement human-centered evaluation strategies to assess the relevance of the tags from a human perspective. This evaluation will consider how well the extracted topics and terms (tags) align with human understanding.

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

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.

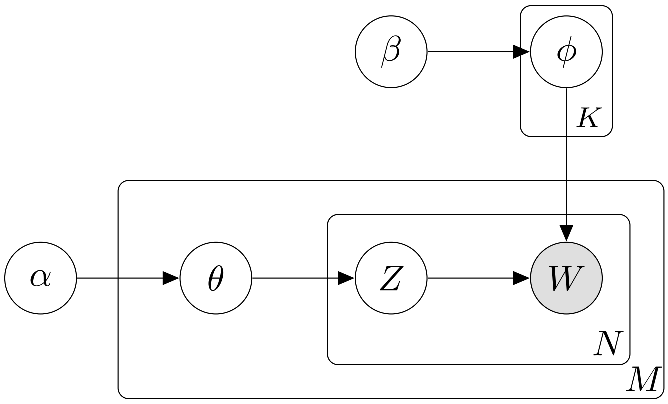


Figure 2.1: LDA plate notation

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. [12] 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) [13, 14, 15] 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.

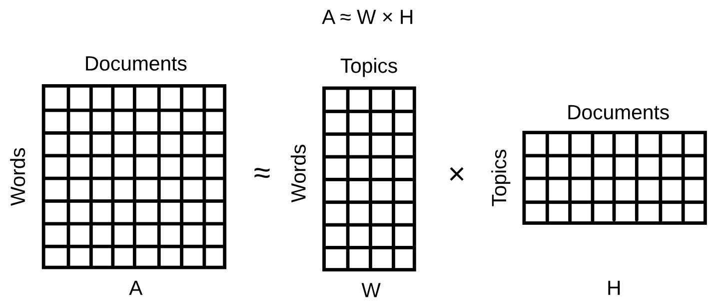


Figure 2.2: NMF decomposition

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 [16], the first step is to embed the documents into dense vector representations (embeddings) to capture the semantic meaning of the text. Figure 2.3 [16] 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.

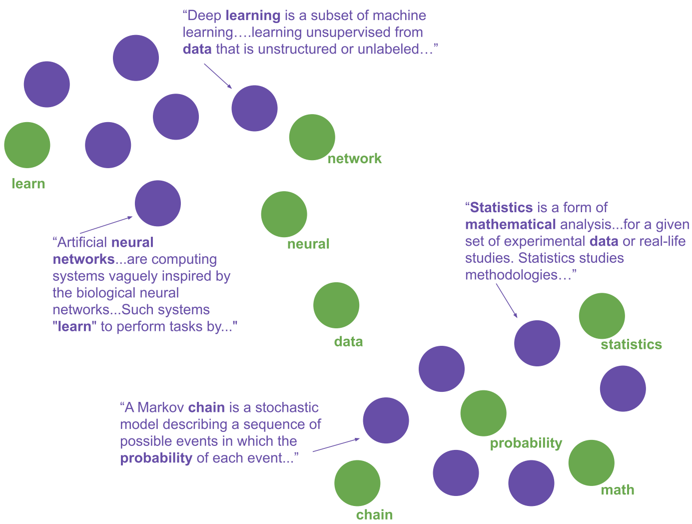


Figure 2.3: Example of embeddings

To learn the embeddings, Top2Vec utilizes Doc2Vec [35, 36], Universal Sentence Encoder [37], or Sentence-BERT (SBERT) [38].

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 [39]. 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, 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) [5] 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) [6, 7, 8]. 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 [2] 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 submodels. 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 [40]) 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 [5] or PCA [41] for dimensionality reduction, and GPT or KeyBERT [42] 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 [38]. 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 [43]. 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 [44]. 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

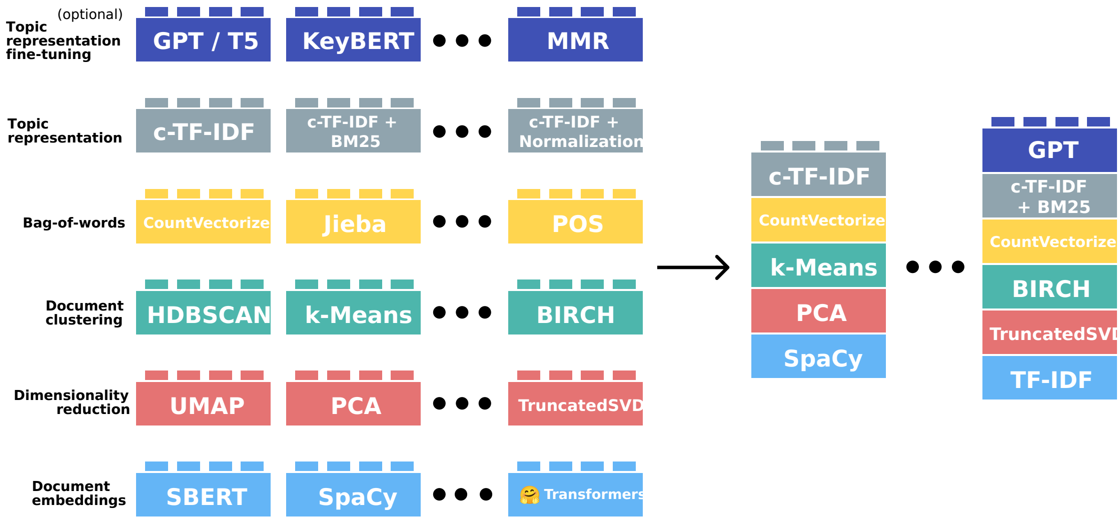


Figure 2.4: BERTopic modularity and steps (from bottom to top)

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) [44, 45]. 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 [38].

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.

GPT [46, 47, 48], BERT [45], T5 [49] and other similar models based on transformers are called large language models (LLMs). They are pre-trained on large corpora of text data and fine-tuned for specific tasks. LLMs have been shown to achieve state-of-the-art performance on a wide range of NLP tasks, including text classification, question answering, and language translation. However, the computational cost of training and fine-tuning LLMs is high, and they require large amounts of data to achieve good performance. In addition, LLMs are often criticized for their lack of interpretability, as the internal workings of the model are complex and difficult to understand.

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

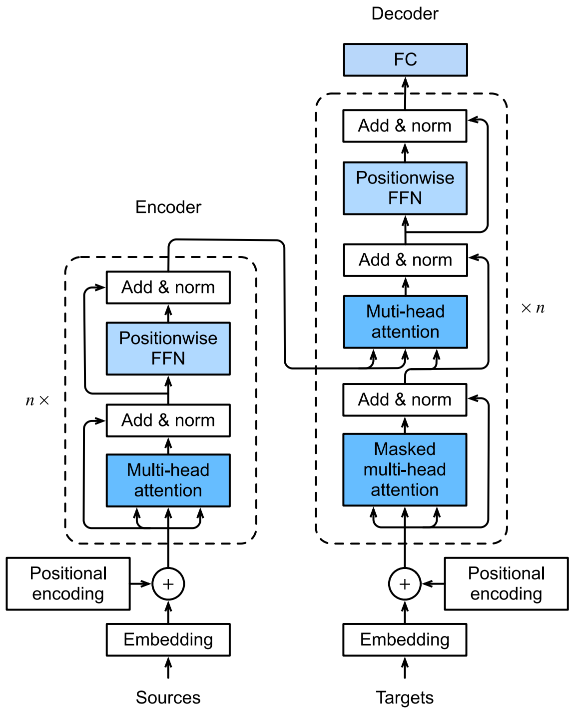


Figure 2.5: Transformer architecture

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) [4] 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 [50], and BERT. By leveraging MTEB, the user can select the most suitable embedding model for their specific use case.

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 [51, 52]. 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* [53, 54], a simpler strategy involves reducing the dimensionality of embeddings. Although PCA [41] and t-SNE [55] 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 [5].

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 more 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

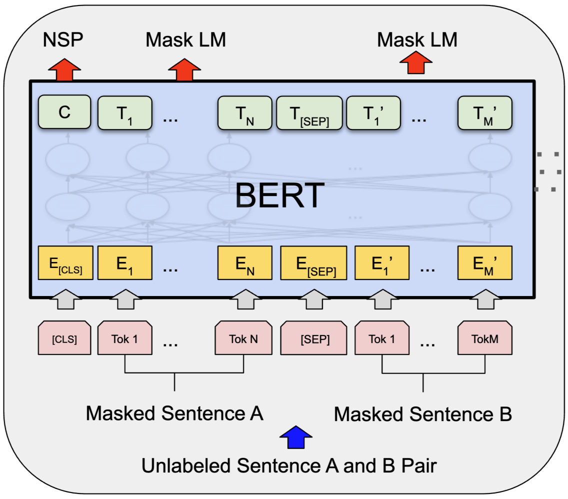


Figure 2.6: BERT architecture

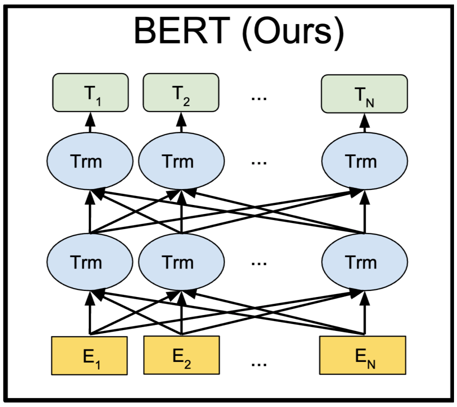


Figure 2.7: BERT bidirectional architecture

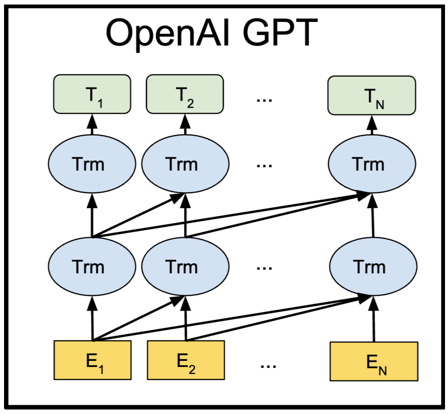


Figure 2.8: OpenAI GPT unidirectional architecture

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 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

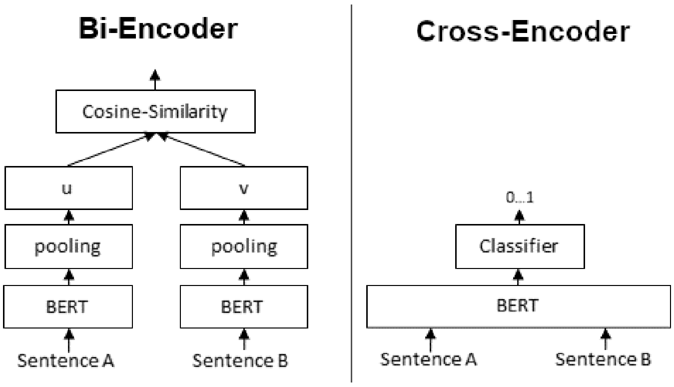


Figure 2.9: Transformer architecture

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*, 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) [6, 7, 8, 56].

HDBSCAN is built on top of DBSCAN [57] 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. [58] 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 [6], 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.

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

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 [59] method combines term frequency and inverse document frequency to calculate a weight $W_{t,d}$ for term t in document d as follows:

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

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.

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

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. However, 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 [46, 47, 48], KeyBERT [42], spaCy [60], 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 candidate topics, comprising a set of terms and representative documents. These can serve as a foundation for 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 Zeroshot text classification

Zeroshot text classification refers to the ability of a model to classify text into predefined categories without having been explicitly trained on those particular categories. This method leverages large pre-trained models that have a broad understanding of language, enabling them to generalize to new tasks without needing specific labeled data for fine-tuning.

In the context of Natural Language Inference (NLI), this can be achieved by reframing classification tasks as entailment tasks. NLI systems are trained to determine whether a given *hypothesis* (a statement) is entailed (i.e., is true or supported) by a *premise* (another statement). For zeroshot text classification, we can treat the task as determining which hypothesis (category) is entailed by a given text.

The zeroshot text classification process can be broken down as follows:

- **Input Text**: This is the text we want to classify. For example, *"The economy is improving rapidly."*
- **Hypothesis Generation**: Each potential label or class is verbalized as a hypothesis. For instance, when classifying topics, a few hypotheses might be:
  - *"This text is about politics."*
  - *"This text is about economics."*

  - *"This text is about sports."*

- **Model Evaluation**: A pre-trained NLI model is used to evaluate how well each hypothesis is entailed by the input text. In zeroshot text classification, the model calculates the probability that each hypothesis is true, given the input text.

- **Prediction**: The hypothesis with the highest probability is selected as the predicted label for the input text. Alternatively, the model can return the probabilities for each hypothesis, making it possible to assign multiple labels to the input text.

For example, in the case of the input text *"The economy is improving rapidly"*, the model would likely assign the highest probability to the hypothesis *"This text is about economics."*

In their recent work, Laurer et al. [61] propose an efficient approach to zeroshot text classification using NLI as a universal task. They demonstrate that NLI can be leveraged to generalize across a wide range of classification tasks, such as topic classification, sentiment analysis, and stance detection. By training a universal classifier on a combination of NLI datasets and non-NLI classification datasets, they achieve significant improvements in zeroshot performance, with a 9.4% increase compared to NLI-only models. Their approach capitalizes on the ability to verbalize classification labels as hypotheses, allowing the model to determine which hypothesis is most consistent with the input text. This method not only improves classification flexibility but also reduces the need for task-specific fine-tuning, making it applicable to diverse applications.

2.5 Evaluation metrics

According to Abdelrazek et al. [31], topic models, which are applicable across a variety of domains, can undergo evaluation through two distinct approaches: extrinsic and intrinsic. Extrinsic evaluation assesses performance based on the specific domain of application, whereas intrinsic evaluation focuses on the qualities of the generated topics themselves, independent of any domain. This makes intrinsic evaluation more universally applicable. The various models are distinguished by their simplicity, computational efficiency, and underlying assumptions, which influence their performance across different corpora and applications. However, there is a lack of agreement on the criteria for evaluating topic models, and multiple methods exist for evaluating the same quality.

Abdelrazek et al. [31] highlight a range of criteria for evaluating topic models, including quality, interpretability, stability, diversity, efficiency, and flexibility, as illustrated in Figure 2.10. We will focus on quality, interpretability, and diversity, given their relevance to our specific use case.

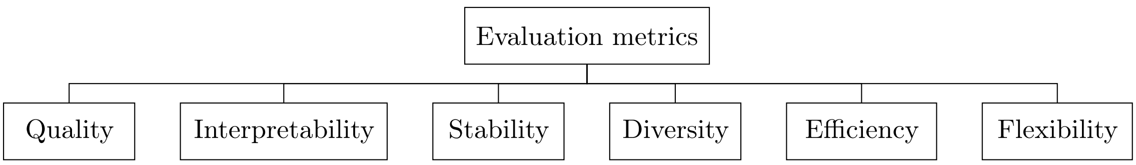


Figure 2.10: Topic models evaluation criteria

2.5.1 Quality and Perplexity

Perplexity measures a model's ability to reproduce the documents in a corpus using the learned topics. It evaluates the model's predictive ability rather than its ability to uncover the latent structure, indicating how effectively the model explains the data. A lower perplexity suggests a model is more effective in explaining the observed documents, as it implies a higher information gain from predicting the outcome of the random variable.

However, using perplexity as an evaluation metric for our use case has several drawbacks. Firstly, perplexity needs to be normalized for the size of the corpus vocabulary, as it varies with different corpus and topic sizes. This is a consideration especially since BERTopic may not consistently extract the same number of topics without specific instructions to limit topic quantity. Besides that, when using custom fine-tuning as a final step in BERTopic, the final extracted terms may not exactly match terms from the original corpus. For instance, when using LLMs, the term *hockey* may exist, but a term like *winter sports* may be created which is not present in the original corpus.

Additionally, perplexity has not been found to be correlated with human judgment [62]. Furthermore, non-generative models like NMF do not have a defined perplexity score because they do not provide probabilities of word sequences.

2.5.2 Interpretability and Topic coherence

A topic is defined as a discrete distribution over words, with the expectation that this word set is interpretable by humans. For interpretability, the words generated should collectively convey a single semantic meaning. Topic coherence metrics evaluate how related the top-k words of a topic are to one another.

Newman et al. [63] measure coherence by examining the lexical similarity between word pairs, employing various similarity measures and identifying mutual information as the most reliably performing measure. The pointwise mutual information, *PMI*, and the normalized pointwise mutual information, *NPMI* [64], between a pair of words (w_i, w_j) are calculated as follows [65]:

$$\text{PMI}(w_i, w_j) = \log \frac{P(w_i, w_j)}{P(w_i)P(w_j)}$$

$$\text{NPMI}(w_i, w_j) = \frac{\log \frac{P(w_i, w_j)}{P(w_i)P(w_j)}}{-\log P(w_i, w_j)} = \frac{\text{PMI}(w_i, w_j)}{-\log P(w_i, w_j)}$$

This formula quantifies the difference between the probability of w_i and w_j occurring together compared to the probabilities of them appearing independently within the corpus. Here, $p(w_i, w_j)$ represents the joint probability of both words occurring together, while $p(w_i)$ and $p(w_j)$ are the individual probabilities w_i and w_j occurring in the corpus.

Topic coherence suffers from the same disadvantages as perplexity, as it cannot account for terms that are not present in the original corpus. In fact, it is important to mention that most modern topic models cannot be adequately evaluated using only perplexity or coherence [66, 67].

Additionally, a known trade-off exists between coherence and perplexity [62], where optimizing for lower perplexity often results in decreased coherence. Nonetheless, as opposed to perplexity, coherence has been found to correlate with human judgment [65, 66, 67, 68, 69, 70].

Many authors contend that the field of topic modeling has largely converged on coherence as the primary evaluation metric [65, 66, 67, 70]. This is because coherence is more closely aligned with human judgment, making it a more reliable measure for evaluating topic models. However, these authors also acknowledge that coherence is not without limitations and advocate for more human evaluations, as well as the development of new metrics that better capture the complexities of topic models, particularly modern neural topic models.

2.5.3 Topic diversity

Topic diversity refers to the semantic diversity among the generated topics. A method to assess diversity, as proposed by Dieng et al. [71], considers it as the proportion of unique words within the top 25 words across all topics:

$$TD = \frac{|\bigcup_{i=1}^{T} \{w_{i,1}, w_{i,2}, ..., w_{i,k}\}|}{kT}$$

Where TD represents the Topic Diversity score; T is the total number of topics; k is the number of top words considered for each topic; $w_{i,j}$ represents the j-th word in the i-th topic; $\bigcup$ denotes the union of sets; and $|\cdot|$ denotes the cardinality of a set (i.e., the number of elements in the set).

So, in general, diversity metrics aim to quantify the variation among the top-k words within a topic. A high score in topic diversity suggests that a topic model successfully generates diverse topics, whereas a low score may indicate the presence of redundant topics, showing the model's inability to clearly differentiate the themes within the corpus. It is important to note that the choice of the number of topics in a model significantly influences topic diversity. Choosing too many topics might lead to similar topics with overlapping words, while too few topics can result in overly broad topics that lack interpretability.

2.5.4 Silhouette score

Silhouette score [72] is a metric used to evaluate the quality of clusters in unsupervised learning. It measures how similar an object is to its own cluster compared to other clusters. The silhouette score ranges from -1 to 1, where a score closer to 1 indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. Conversely, a score closer to -1 suggests that the object is poorly matched to its own cluster and well matched to neighboring clusters. A score of 0 indicates that the object is on the boundary between two clusters.

2.5.5 Classification evaluation metrics

The evaluation metrics discussed previously pertain to topic modeling as an unsupervised learning task. In our case, we treat the topic modeling of OpenML datasets as an unsupervised problem. However, if ground truth labels exist, the evaluation of topic models can be approached as a classification problem. Beyond well-known metrics such as accuracy, precision, recall, and F1 score, coverage and purity would also be considered [32].

Coverage examines the extent to which the concepts within the document collection are captured by the model. It can be divided into topic coverage and document coverage. Topic coverage measures how good the model is at identifying the topics in a document corpus. The most popular measure for topic coverage is topic recall, which denotes the proportion of ground truth topics identified by the topic model. Conversely, document coverage focuses on how well documents are represented by the topics. Topic model accuracy is a frequently used measure, which is the proportion of documents accurately labeled by the model. For evaluating both topic recall and accuracy, ground truth topics are required.

When ground truth topics are missing, alternative metrics like purity are used. Purity measures the model's accuracy under the assumption that documents are always assigned to the dominant topic. This metric aims to penalize models that assign a large number of low probability topics to documents, in contrast to models that assign a high probability to a single topic from the document corpus.

2.5.6 OCTIS

OCTIS (Optimizing and Comparing Topic models Is Simple) [73] is an open-source framework for the training, analysis, and comparison of topic models across various datasets and evaluation metrics.

It allows for the optimization of model hyper-parameters for experimental comparison. OCTIS introduces a pipeline for topic modeling (Figure 2.11), which includes dataset preprocessing, training topic models, evaluation metrics, hyperparameter optimization, and visualization.

OCTIS offers a range of evaluation metrics for assessing topic models, such as coherence, diversity, and classification metrics.

The discovery of good hyper-parameter settings relies on a Bayesian Optimization (BO) approach [74, 75, 76], where the objective can be any of the available evaluation metrics. Given the potential variability in performance outcomes due to noise, the objective function is defined as the median performance across multiple runs of the model under the same hyperparameter settings for the chosen evaluation metric.

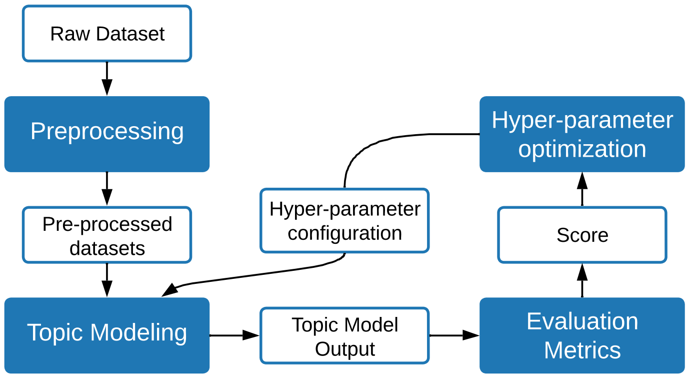


Figure 2.11: Workflow of the OCTIS framework

BO is a sequential, model-based optimization technique for noisy black-box functions that are costly and complex to evaluate directly, such as topic models. Its main idea involves using all previously evaluated hyperparameter settings to approximate the performance metric's value, and then selecting new, likely better hyperparameter settings for the next run. The approximation is done by a probabilistic *surrogate model*, which has a prior belief of the objective function based on observed hyperparameter settings. The selection of the next hyperparameter settings is driven by optimizing an *acquisition function*, which uses the uncertainty within the posterior distribution to guide the exploration of the parameter space.

OCTIS is useful for our use case, as it provides a unified framework for training the proposed BERTopic model alongside the baseline models, facilitating their comparison across a variety of evaluation metrics. In fact, in the original BERTopic paper, Grootendorst [2] employed OCTIS to evaluate the model's performance.

Additionally, we can utilize OCTIS' hyperparameter optimization capabilities (BO) to find good hyperparameter settings for the proposed BERTopic model.

2.6 Chapter conclusion

This chapter provides an overview of topic modeling approaches, from traditional statistical methods to modern embedding-based techniques, along with evaluation metrics for assessing their performance.

While LDA [12] and NMF [13, 14, 15] are seminal and widely adopted models that have proven effective across various domains, they are probabilistic, treating words as discrete tokens without considering their semantic relationships. Traditional word embedding techniques like GloVe [77] and Word2Vec [78] attempted to address this by creating context-free embedding spaces, but they still assign a single, static embedding to each word, limiting their ability to capture the nuanced meanings words can have in different contexts [79].

Top2Vec [16] and BERTopic [2] advance beyond these earlier approaches by leveraging embedding techniques that better capture semantic relationships. These models utilize transformer-based architectures that can understand words in context. Unlike GloVe and Word2Vec, these models can generate contextual embeddings where the same word can have different representations depending on its usage.

BERTopic improves upon Top2Vec's foundation by introducing several innovations. The addition of c-TF-IDF provides a method for identifying distinctive terms within topics, while the optional fine-tuning step allows for refinement of topic representations using language models.

These innovations have led to superior performance on standard metrics, with BERTopic achieving higher NPMI (coherence) and diversity scores compared to other approaches [2]. BERTopic's modular architecture, comprising separate components for embeddings, dimensionality reduction, clustering, c-TF-IDF, and topic representation, makes it possible to evolve the model as new embedding models, clustering algorithms, and dimensionality reduction techniques are introduced.

BERTopic's modular design allows for selecting state-of-the-art components at each processing stage. For embeddings, Salesforce SFR-Embedding-2_R [3] is currently one of the best models on the MTEB benchmark [4]. In the dimensionality reduction step, UMAP has been shown to better preserve both local and global data characteristics compared to alternatives like PCA and t-SNE [5]. For clustering, HDBSCAN demonstrates several advantages over conventional algorithms: unlike k-means, it doesn't assume spherical clusters or require pre-specifying the number of topics; and compared to DBSCAN, it adaptively identifies clusters across different density scales, making it more robust for heterogeneous document collections [6, 58].

Although Top2Vec and BERTopic utilize transformer-based embedding models that are expensive to train, they leverage pre-trained versions of these models, eliminating the training overhead and making their practical application comparable in computational cost to traditional approaches.

These reasons make BERTopic particularly well-suited for generating tags for OpenML dataset descriptions. The modular nature of the model also ensures it can evolve alongside advances in language models and embedding techniques, making it a sustainable long-term solution.

It is also important to highlight the state of evaluation metrics in the field. Topic modeling evaluation has largely converged on using NPMI coherence and topic diversity as primary automated metrics [65, 66, 67, 70]. These metrics are widely adopted in the literature, where researchers typically report coherence and diversity scores alongside human evaluation results.

The widespread adoption of NPMI and diversity metrics influenced our choice to use them as optimization and evaluation metrics. Furthermore, these metrics provide complementary perspectives on topic quality: while coherence measures how semantically related words within a topic are, diversity ensures that topics remain distinct from one another. This creates balance, as increasing coherence often leads to less diverse topics (since semantically related words are often similar), while maximizing diversity might reduce coherence. This balance helps prevent the model from repeatedly selecting the same high-frequency terms across different topics.

However, we acknowledge the limitations of these automated metrics. Several studies have found that coherence and diversity scores show only small to moderate correlations with human judgment [66, 67]. The relationship between these metrics and actual topic quality, as perceived by humans, is not straightforward. Some researchers have questioned whether automated topic model evaluation is fundamentally flawed [67], pointing out that coherence metrics may not capture the nuanced ways humans interpret and understand topics.

While other evaluation metrics exist, such as perplexity and various classification-based metrics, they are less commonly adopted in the field and their relationship with human judgment is even less studied. Additionally, many of these metrics are not applicable to modern neural topic models [66], which often generate terms not present in the original corpus through fine-tuning steps.

For implementing these metrics and comparing different topic models, we choose to use the OCTIS framework [73]. OCTIS provides a standardized environment for training and evaluating topic models, ensuring fair comparisons between different approaches. Its support for Bayesian optimization of hyperparameters is particularly valuable for our work with BERTopic, as it allows us to explore the hyperparameter space while optimizing for both coherence and diversity.

Given the limitations of automated metrics, we acknowledge that they should not be the sole basis for evaluating topic model performance. While we use these metrics for optimization and baseline comparisons to align with field practices, we place greater emphasis on human evaluation, which will be discussed in later chapters. This approach allows us to more directly assess whether the generated topics and tags are truly meaningful to users, which is ultimately the most important criterion for a practical topic modeling system.

This chapter primarily addresses **RQ2** and **RQ3**. For **RQ2**, we conduct an examination of topic modeling approaches, from traditional statistical methods such as LDA and NMF to modern embedding-based techniques like Top2Vec and BERTopic. We analyze their advantages and disadvantages, particularly focusing on BERTopic's modularity and ability to evolve with advances in the field. For **RQ3**, we investigate evaluation metrics, explaining why NPMI coherence and diversity have become standard metrics in the field while acknowledging their limitations. We also introduce OCTIS as our framework for implementing these metrics and optimizing model performance.

Chapter 3

Methodology

In this chapter, we present our proposed methodology, including the methods and techniques used to address our research questions. Our methodology consists of four main components: (1) data exploration, preprocessing, and augmentation to prepare the OpenML dataset descriptions, (2) a novel tag generation pipeline that extends BERTopic with large language models and zeroshot classification, (3) automated evaluation and optimization using topic coherence, diversity metrics, and baseline comparisons, and (4) human evaluation through a user study complemented by large-scale automated evaluation. We provide a detailed description of each component, along with their limitations and how they address our research questions.

3.1 Data exploration, preprocessing and augmentation

3.1.1 Exploratory data analysis

First, an exploratory data analysis (EDA) will be conducted to understand the characteristics of the OpenML dataset descriptions. Figure 3.1 shows an example of an OpenML dataset, which includes a dataset name (`COVID-19-Rio-de-Janeiro-(City)`), a dataset description (`Context - World Health Organization (WHO)...`), tags (`Computer Systems, Machine Learning`), and columns (`Date, Hour, Neighborhood, Cases, Deaths, Recovered`).

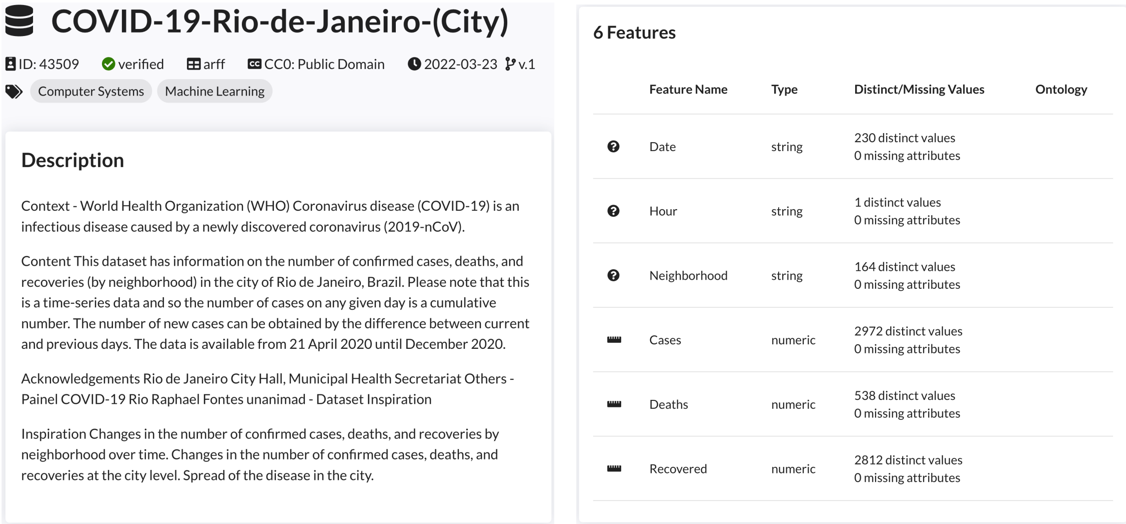
This analysis will be unstructured, discovering patterns in the data in an exploratory manner, using statistical and visual methods. The goal of the EDA is to gain insights into the dataset descriptions, such as their length, vocabulary, distribution of words, etc. This information will help us understand the nature of the data and identify any preprocessing steps that may be necessary to improve the quality of the descriptions.

3.1.2 Data preprocessing

After the EDA, the dataset descriptions will undergo preprocessing to prepare them for the topic modeling process. For traditional topic models like LDA and NMF, which rely on sparse representations of text (e.g., bag-of-words or TF-IDF) and do not inherently understand semantic relationships between words, several preprocessing steps are necessary:

- Stemming and lemmatization: These reduce words to their root or base form (e.g., *running* to *run*), ensuring that different forms of the same word are treated as identical. This is important for models that treat words as discrete tokens, as it helps capture true word frequencies.

- Stop-word removal: Common words like *the*, *and*, *is* are removed as they carry little semantic meaning and can dominate frequency-based representations.



(a) Dataset information (b) Dataset columns

Figure 3.1: OpenML dataset example

- Tokenization: Text is split into individual tokens (words or subwords), creating the discrete units that these models operate on.

Modern embedding-based models like Top2Vec and BERTopic typically do not require these preprocessing steps, as they use embedding models that can capture semantic meaning and handle word variations naturally.

3.1.3 Data augmentation

In addition to the original dataset descriptions, we will explore the possibility of augmenting the data with additional information. This could include metadata such as dataset name, tags, features (column names), scraping from the original dataset if available. This additional information can provide context and background to the descriptions, which may help improve the quality of the topics extracted by the topic model.

3.2 Tag generation

3.2.1 Pipeline

In this subsection, we introduce our model, which is, in fact, a pipeline comprising multiple submodels and techniques. We shall henceforth refer to this pipeline as our *proposed model*.

Steps 1-3 involve data preprocessing to prepare the data for the proposed model. We shall refer to steps 5-8 as the *Base BERTopic model*. It includes dimensionality reduction, clustering, bag-of-words and c-TF-IDF. Finally, steps 9 and 10 represent our improvements to the base model, introducing an approach that, to the best of our knowledge, is novel within the literature. Additionally, we provide an explanation of which steps are computationally efficient and which are more expensive.

To explain the pipeline illustrated in Figure 3.2 in more detail, we provide a step-by-step description of the process:

1. **Original Descriptions**: The input to the pipeline is a set of original dataset descriptions. These come from the OpenML dataset and are used as the basis for generating tags.

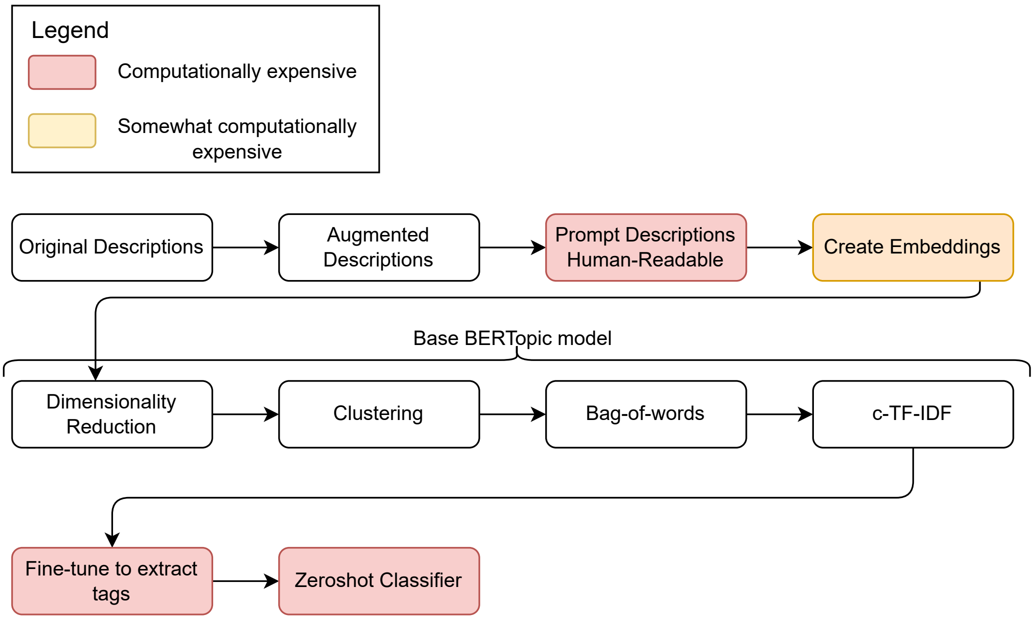


Figure 3.2: Tag generation pipeline (*proposed model*)

2. **Augmented Descriptions**: The OpenML dataset descriptions come with metadata such as dataset name, tags, features (column names) and some of them link to the original dataset, which can be used to scrape (extract) additional information. These are used to augment the dataset descriptions.

3. **Prompt Descriptions Human-Readable**: Augmented descriptions are rewritten to be more human-readable via an LLM and prompt engineering. This is because the original descriptions are often in a technical format that is not easily interpretable by humans. Since LLMs are trained on large text corpora, they work best with human-readable (natural language) text.

   Additionally, we design the prompt to extract *keyword tags* from each individual description. These tags are directly mentioned within the description and are typically highly specific to it. For instance, if a description includes the terms "US elections," "voting," and "candidates," these would be considered keyword tags.

   Figures 3.3 and 3.4 show an example of an augmented description before and after the augmentation and rewriting process.

4. **Create Embeddings**: Embeddings for each description are created using a pre-trained embedding model.

5. **Dimensionality Reduction**: The dimensionality of the embeddings is reduced to cure the curse of dimensionality.

6. **Clustering**: The reduced embeddings are clustered to group similar descriptions together. The output of this step is clusters, which represent our topics. Each cluster contains a set of descriptions (which we now call *representative documents*) that are similar to each other.

7. **Bag-of-words**: The descriptions in each cluster are converted to a bag-of-words representation, ignoring common words such as "the", "and", etc.

8. **c-TF-IDF**: The bag-of-words representation is used to calculate the c-TF-IDF score for each word in each cluster. This score is used to rank the words in each cluster.

9. **Fine-tune To Extract Tags**: For each topic, we have representative documents (from the clustering step) and representative words (from the c-TF-IDF step). We prompt engineer a question that asks an LLM to generate tags for each cluster. The generated tags are categorized as *regular tags* and *overarching tags*.

   Regular tags refer to tags that frequently appear among the representative documents and words. *Overarching tags* capture the broader, more general theme of the cluster. For example, if a cluster pertains to *US elections* and the representative documents contain the word *election* while the representative words include *candidate*, a possible regular tag could be *election candidate*, whereas the overarching tag might be *politics*.

   As context, we feed the LLM with the top k representative documents and the top m representative words for each topic. This results in tags that are common among the representative documents and representative words, and hence are representative of the topic.

   We prompt engineer a question that asks the LLM to generate tags for each cluster, and provide a few examples. This is known as few-shot prompting, which has been shown to improve the performance of LLMs on specific tasks compared to zero-shot or one-shot prompting [48, 80, 81, 82].

10. **Zeroshot Text Classifier**: Each description can in reality be contained in multiple clusters (topics) if we use a soft clustering approach such as HDBSCAN. In this step, we get the top n most likely clusters for each description. Then, for each description, we get the tags for each of the top n clusters in a set.

    The advantage provided by this step is that a description can be contained in multiple clusters, and we can get tags for each of these clusters. This is important because a description may contain multiple topics, and we want to ensure that we capture all of them. For instance, we may have a cluster about *politics* and a cluster about *sports*. A description about the Olympics may be contained in both of these clusters, and we want to ensure that we get tags for both of these clusters.

    We then feed this set of tags to a zeroshot text classification model, which returns confidences from 0 to 1 for whether each tag describes the description. This step is crucial for filtering out irrelevant tags for each description. For instance, if a description about diabetes and a description about cancer are both contained in the same topic (which may be, for example, *medical conditions*), we want to ensure that the tags for the cancer description are not assigned to the diabetes description. Furthermore, the description about diabetes may be contained in another topic that cancer is not in, such as *nutrition*. In this case, we want to ensure that the tags in *nutrition* are not assigned to the cancer description, but are assigned to the diabetes description.

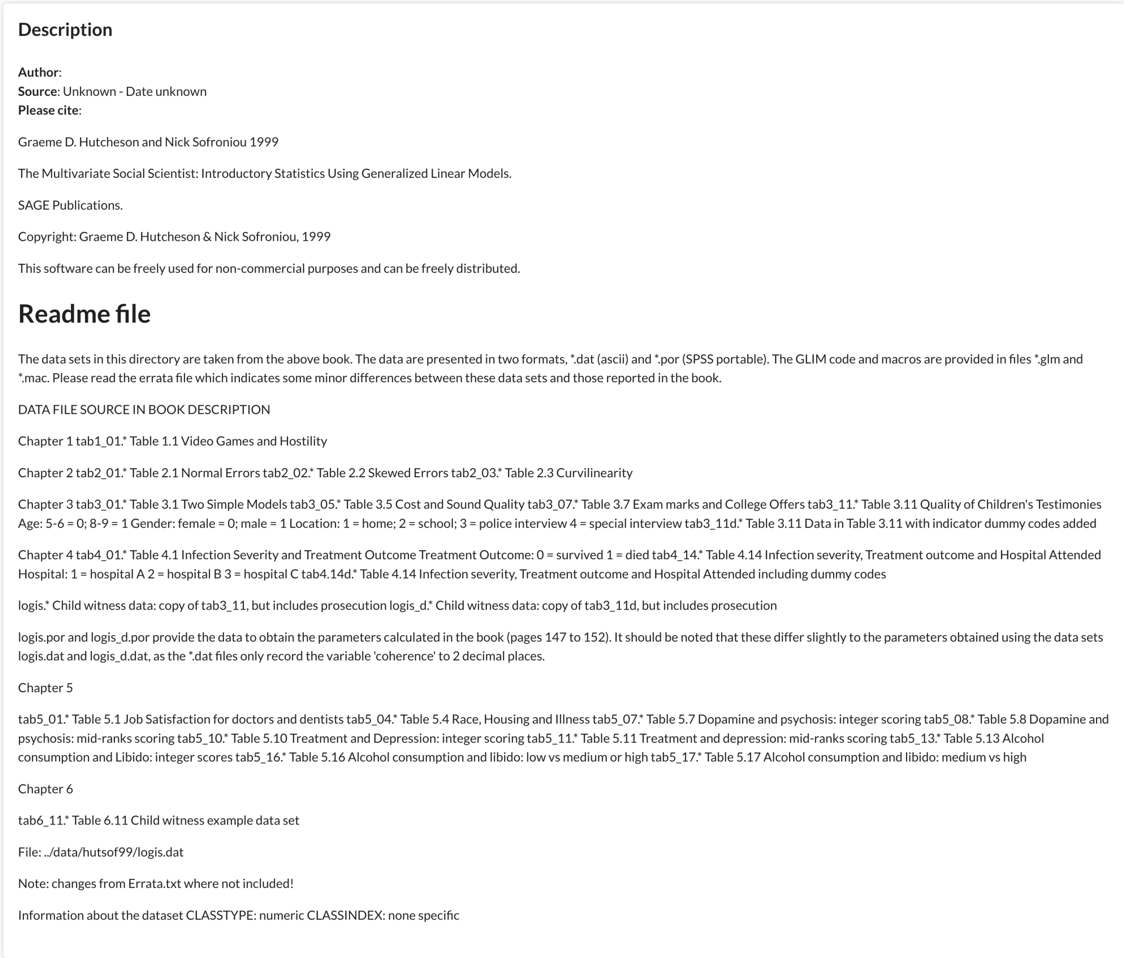
3.2.2 Rationale

In this subsection, we aim to provide a rationale for the proposed model.

Proposed model

We provided our rationale for steps 1 and 2 in section 4.1, and for step 3 in section 3.2.1. Steps 4-8 are part of the Base BERTopic model, a well-established topic modeling technique proven effective across various datasets. Hence, we will focus on steps 9 and 10, which are novel contributions to the existing literature.

Without steps 9 and 10, we would need to solely rely on the c-TF-IDF representation from step 8 to extract tags. For background on c-TF-IDF, the reader may refer to sections 2.4.4 and 2.4.5. This c-TF-IDF representation has several limitations:


Description

Author:
Source: Unknown - Date unknown
Please cite:

Graeme D. Hutcheson and Nick Sofroniou 1999

The Multivariate Social Scientist: Introductory Statistics Using Generalized Linear Models.

SAGE Publications.

Copyright: Graeme D. Hutcheson & Nick Sofroniou, 1999

This software can be freely used for non-commercial purposes and can be freely distributed.

Readme file

The data sets in this directory are taken from the above book. The data are presented in two formats, *.dat (ascii) and *.por (SPSS portable). The GLIM code and macros are provided in files *.glm and *.mac. Please read the errata file which indicates some minor differences between these data sets and those reported in the book.

DATA FILE SOURCE IN BOOK DESCRIPTION

Chapter 1 tab1_01.* Table 1.1 Video Games and Hostility

Chapter 2 tab2_01.* Table 2.1 Normal Errors tab2_02.* Table 2.2 Skewed Errors tab2_03.* Table 2.3 Curvilinearity

Chapter 3 tab3_01.* Table 3.1 Two Simple Models tab3_05.* Table 3.5 Cost and Sound Quality tab3_07.* Table 3.7 Exam marks and College Offers tab3_11.* Table 3.11 Quality of Children's Testimonies Age: 5-6 = 0; 8-9 = 1 Gender: female = 0; male = 1 Location: 1 = home; 2 = school; 3 = police interview 4 = special interview tab3_11d.* Table 3.11 Data in Table 3.11 with indicator dummy codes added

Chapter 4 tab4_01.* Table 4.1 Infection Severity and Treatment Outcome Treatment Outcome: 0 = survived 1 = died tab4_14.* Table 4.14 Infection severity, Treatment outcome and Hospital Attended Hospital: 1 = hospital A 2 = hospital B 3 = hospital C tab4.14d.* Table 4.14 Infection severity, Treatment outcome and Hospital Attended including dummy codes

logis.* Child witness data: copy of tab3_11, but includes prosecution logis_d.* Child witness data: copy of tab3_11d, but includes prosecution

logis.por and logis_d.por provide the data to obtain the parameters calculated in the book (pages 147 to 152). It should be noted that these differ slightly to the parameters obtained using the data sets logis.dat and logis_d.dat, as the *.dat files only record the variable 'coherence' to 2 decimal places.

Chapter 5

tab5_01.* Table 5.1 Job Satisfaction for doctors and dentists tab5_04.* Table 5.4 Race, Housing and Illness tab5_07.* Table 5.7 Dopamine and psychosis: integer scoring tab5_08.* Table 5.8 Dopamine and psychosis: mid-ranks scoring tab5_10.* Table 5.10 Treatment and Depression: integer scoring tab5_11.* Table 5.11 Treatment and depression: mid-ranks scoring tab5_13.* Table 5.13 Alcohol consumption and Libido: integer scores tab5_16.* Table 5.16 Alcohol consumption and libido: low vs medium or high tab5_17.* Table 5.17 Alcohol consumption and libido: medium vs high

Chapter 6

tab6_11.* Table 6.11 Child witness example data set

File: ../data/hutsof99/logis.dat

Note: changes from Errata.txt where not included!

Information about the dataset CLASSTYPE: numeric CLASSINDEX: none specific

Figure 3.3: Augmented dataset description example (before)

- It is based on the bag-of-words representation of the descriptions, which may not capture the full semantic meaning of the text. The c-TF-IDF calculation relies on word frequencies in the descriptions, which is a naive way of capturing word importance in the context of topics, as it does not consider the relationships between words.

- It depends on the top words in each cluster, which may not always be representative of the topics. For instance, if a cluster's top words are *election* and *candidate*, these words may not capture the broader theme of *politics*.

The LLM helps address these limitations by generating tags that better represent the topics. Trained on a large corpus of text, it can capture semantic relationships between words. By providing the LLM with the top representative documents and words for each cluster, we can prompt it to generate more representative topic tags. The LLM can identify broader themes and generate more descriptive tags, improving tag quality for users.

If we only use step 9 and not step 10, we have two significant limitations:

- We miss the opportunity to capture the multiple topics a description may belong to.

- We cannot filter out irrelevant tags for each description, as not all cluster tags may be relevant to every description within that cluster, as illustrated by the example in step 10 of section 3.2.1.

Description

Name: hutsof99_logis

Tags: StatLib, survival

The data sets in this directory are taken from the book "The Multivariate Social Scientist: Introductory Statistics Using Generalized Linear Models" published by SAGE Publications. The data are presented in two formats: ASCII and SPSS portable. GLIM code and macros are provided in separate files. Please refer to the errata file for minor differences between these data sets and those reported in the book.

Some data sets include additional information such as age groups, gender, location types, treatment outcomes, and hospital identifiers. Certain files contain dummy codes or alternative scoring methods.

The child witness data set includes information on prosecution. It should be noted that there are slight differences in parameters obtained using different file formats due to decimal place precision.

The data can be freely used for non commercial purposes and can be freely distributed. Users are requested to cite the book when using this data.

Keywords: multivariate statistics, generalized linear models, social science data, GLIM, SPSS, child testimonies, medical outcomes, psychological studies

Figure 3.4: Augmented dataset description example (after)

Conversely, if we only use step 10 and not step 9, we must rely on the c-TF-IDF representation to classify tags. As discussed, this representation has several limitations, and using it alone may result in lower quality and less representative tags. By combining steps 9 and 10, we leverage the strengths of both LLMs and zeroshot classifiers to generate tags that are higher quality, more representative, and more relevant to the descriptions.

Advantages over naive LLM approaches

A naive approach would be to simply feed each individual description to an LLM and ask it to extract tags. However, our proposed pipeline offers several advantages over this approach:

1. **Contextual Consistency**: By clustering similar descriptions together, our model develops a broader understanding of related topics. This allows it to generate more consistent and contextually relevant tags across similar descriptions. For example, if we have two similar descriptions about image classification using convolutional neural networks, a naive approach might tag one with *CNN* and another with *computer vision*, while our approach would recognize their similarity and consistently apply both relevant tags to both descriptions. A naive approach, analyzing descriptions in isolation, would generate inconsistent tags for similar descriptions due to lacking this broader context and varying interpretations of individual texts.

2. **Multi-level Tag Generation**: Our approach generates tags at multiple levels of specificity:

   - Keyword tags from individual descriptions (step 3)
   - Regular tags that are common across cluster members (step 9)
   - Overarching tags that capture broader themes (step 9)

   A naive LLM approach, working with single descriptions, would focus on explicitly mentioned concepts in individual descriptions and miss broader thematic connections.

3. **Cross-validation**: The combination of clustering and zeroshot classification provides a form of cross-validation for tag relevance. Tags are not just extracted from individual descriptions but are validated against similar descriptions within the same cluster and verified through the zeroshot classifier. For example, consider a cluster about weather forecasting where one of the descriptions briefly mentions *neural networks*. While the LLM might generate *neural*

networks as a tag for the cluster, the zeroshot classifier would likely assign it a low confidence score for other descriptions in the cluster, since it is not their main focus. This helps filter out irrelevant tags, thus improving tag quality.

4. **Computational Efficiency**: While our pipeline uses LLMs, it does so more efficiently by processing clusters rather than individual descriptions. In a naive approach, we would need n LLM calls for n descriptions, while our clustering approach requires only k calls, where k is the number of clusters ($k < n/2$, $k < n/3$, or even $k < n/5$, depending on the clustering parameters). This significantly reduces the number of LLM calls needed, particularly for large datasets with many similar descriptions. For large clusters, we can further optimize token usage by selecting only the most representative documents (e.g., using 50 instead of all 80 descriptions in a cluster) while still maintaining the cluster's semantic meaning through the remaining representative descriptions. Similarly, we can truncate individual description lengths while preserving the overall topic through the collective context of the cluster. In contrast, with a naive LLM approach, truncating or dropping parts of individual descriptions would result in lost context and lower quality tags, as each description is processed in isolation without the benefit of related documents providing additional context.

Alternative configurations

The pipeline described in Figure 3.2 is one possible configuration for extracting tags from the OpenML dataset descriptions, which we believe will yield high-quality tags based on our reasoning. However, this pipeline is more computationally expensive due to the use of large language models and zeroshot text classifiers. To address this issue, we can consider alternative configurations that are more computationally efficient but may sacrifice some quality in tag generation.

1. **Simpler Models**: We could use a smaller LLM for step 9 or a more lightweight zeroshot text classifier [83] for step 10 to reduce computational cost.

   Alternatively, the zeroshot classifier could be entirely replaced with a cheaper LLM. However, the confidence scores that the LLM outputs may not be as accurate as those from the zeroshot classifier, since the latter is specifically designed for text classification tasks. This could result in lower tag quality by assigning irrelevant tags to descriptions or missing relevant tags.

2. **Pipeline Modification**: Depending on computational constraints, we could modify the pipeline to use only the most essential steps. For instance:

   - Removing the zeroshot classifier and relying solely on the LLM
   - Using the zeroshot classifier without the LLM

   However, we expect that eliminating either component would significantly reduce tag quality based on our reasoning in the previous sections, so we do not recommend this approach.

3. **Skip Description Rewriting**: We could omit step 3, which involves rewriting augmented descriptions to be more human-readable. While this step enhances embedding quality and, consequently, tag quality, it is computationally expensive. If computational constraints become an issue, removing this step offers a reasonable trade-off.

4. **Combined Generation and Filtering**: Instead of using a separate zeroshot classifier, we could modify the fine-tuning step (step 9) to simultaneously generate and filter tags. This approach would generate tags for a cluster and assign them to each description in one step. However, this would skip getting the top n most likely clusters for each description, which could result in missing relevant tags for descriptions that belong to multiple clusters, thus reducing tag quality.

We recommend maintaining the complete pipeline for initial evaluation to ensure high-quality tags, and only exploring these alternative configurations if computational constraints become problematic.

3.3 Automated evaluation metrics and baselines

In order to evaluate the quality of the extracted topics, we will use a combination of automated evaluation metrics and comparison with baselines. These metrics and baselines will help us assess the performance of our model and compare it to existing topic modeling techniques. The following sections provide an overview of the metrics and baselines we will use in our evaluation.

3.3.1 Metrics

From the range of possible evaluation metrics for topic models — including quality, interpretability, stability, diversity, efficiency, and flexibility — we focus on three metrics that are most relevant to our use case and are supported by literature, as explained in section 2.5. Most of the remaining metrics appear infrequently in the literature and will not be included in our evaluation. Among our selected metrics, all except the silhouette score are widely adopted in topic modeling research. However, as discussed in section 2.5, these commonly used metrics have significant limitations [66, 67], especially when it comes to newer topic modeling approaches such as neural models. Therefore, their results should be interpreted with caution.

Topic coherence

As explained in section 2.5.2, topic coherence is a widely used metric for evaluating the quality of topics generated by topic models. It measures the semantic similarity between words in a topic and is based on the assumption that coherent topics contain words that are related to each other. As explained in section 2.5, topic coherence has been shown to correlate with human judgment, so we will use it to assess the cohesion of the topics extracted by our model.

Topic diversity

Topic diversity (section 2.5.3) is another important metric for evaluating topic models. It measures the extent to which topics are distinct from each other and do not overlap in terms of the words they contain. A diverse set of topics ensures that the model captures a wide range of themes and concepts present in the data. We will use topic diversity to evaluate the diversity of topics generated by our model.

Silhouette score

The silhouette score (section 2.5.4) is a metric used to evaluate the quality of clusters in unsupervised learning. It measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). A high silhouette score indicates that the clusters are well-separated and that the objects within each cluster are similar to each other. We will use the silhouette score to evaluate the quality of the clusters generated by our model.

3.3.2 Baselines

In addition to the proposed model, we will compare the performance of the Base BERTopic model against several *baseline models*. These baselines represent established or commonly used topic modeling techniques and will serve as a point of reference for evaluating the proposed model.

It is important to note that only the Base BERTopic model can and will be evaluated using the automated evaluation metrics, not the whole pipeline. There are two reasons for this limitation. First, the proposed model is a pipeline that includes several submodels and techniques, making it difficult to evaluate the pipeline as a whole using automated metrics due to resource constraints. Second, coherence and diversity metrics are not directly applicable to the proposed model, as it generates tags rather than topics. Therefore, automated evaluation metrics will be used only up to the Base BERTopic model (step 8 in Figure 3.2).

Additionally, BERTopic, on its own, automatically determines the final number of clusters (topics) based on the data, which is a significant advantage over traditional topic modeling techniques that require the number of topics to be specified in advance. However, when comparing the Base BERTopic model to baselines, we will use the same number of topics for all models to ensure a fair comparison. We do this by taking the clusters generated by BERTopic, and reducing them to the same number of topics as the baselines by merging clusters based on their similarity until the desired number of topics is reached. It is important to note that this process may result in a loss of information, as the original clusters may contain more nuanced distinctions than the merged topics, thus potentially disadvantaging BERTopic in the comparison.

The baselines we will consider include Latent Dirichlet Allocation (LDA), Non-negative Matrix Factorization (NMF), and Top2Vec (described in sections 2.1 to 2.3), and Contextualized Topic Models (CTM) [17, 18] which are a less popular, but still effective topic modeling approach. These models are widely used in the field of topic modeling and will provide a benchmark for evaluating the performance of our model.

3.3.3 Automated evaluation pipeline

Figure 3.5 shows a flowchart of the automated evaluation pipeline, which contains the following sequential steps:

1. **Data Fetching**: This is the first stage where dataset descriptions are downloaded from OpenML.
2. **Data Preparation**: After fetching the data, the next step involves preparation by preprocessing and augmenting it. Data preparation ensures that the input to the topic model is of high quality, which is important for the success of the subsequent modeling steps.
   The next step involves removing inadequate data points such as duplicates. Stop words are removed for models that require it (e.g., LDA, NMF). Additionally, the process includes stemming and lemmatization to normalize words to their base forms.
3. **Topic Model**: In this step, the Base BERTopic model is applied to the prepared data.
4. **Benchmark Models**: Concurrently with the Base BERTopic model, benchmark models are run. These models represent established or baseline approaches [2, 12, 13, 14, 15, 16, 17, 18] to topic modeling against which the performance of the proposed topic model is compared.
5. **Topic Labels**: The output from both the Base BERTopic model and the benchmark models are sets of topics, represented by a cluster of words that are characteristic of a particular topic.
6. **Evaluation**: Finally, the performances of the Base BERTopic model and benchmark models are evaluated. This includes comparing the topic coherence, diversity and silhouette score.

3.3.4 Hyperparameter tuning

As mentioned in section 2.5.6, we will use Bayesian optimization to tune the hyperparameters of our Base BERTopic model. This process will involve selecting the most relevant hyperparameters to optimize, defining the search space for each hyperparameter, and running the optimization algorithm to find the best combination of hyperparameters. The goal of hyperparameter tuning is to improve the performance of our model by finding the settings for the hyperparameters which are close to optimal.

As a metric to optimize, we will define a custom weighted metric which includes the topic coherence score (NPMI) and topic diversity score. We do not include the silhouette score in the weighted metric as it is positively correlated with the topic coherence score, since both metrics indirectly measure cluster quality, and we do not want to overweight the coherence score. Conversely,

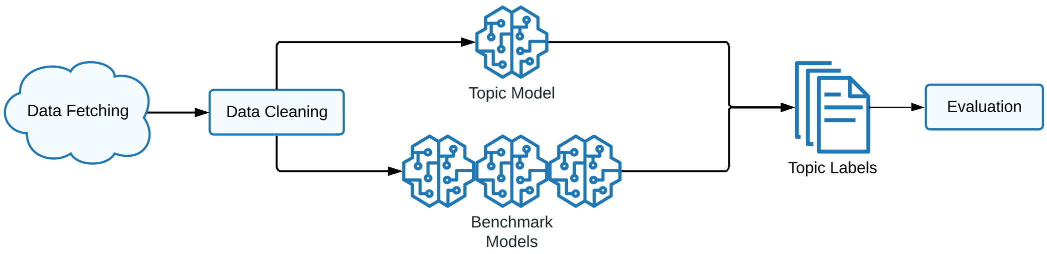


Figure 3.5: Data pipeline

the diversity score is negatively correlated with the coherence score (i.e., more diverse topics are less coherent, since they contain fewer semantically related words), so we want to ensure that both metrics contribute meaningfully to the final score. The weighted metric is defined as:

Let C represent the topic coherence score normalized between $[-1, 1]$, and D represent the topic diversity score normalized between $[0, 1]$. The normalized scores are given by:

$$C_{\text{norm}} = \frac{C+1}{2}, \quad D_{\text{norm}} = D$$

We apply a log transformation to both scores. The log-transformed scores are defined as:

$$\log(C_{\text{norm}} + \epsilon), \quad \log(D_{\text{norm}} + \epsilon)$$

where $\epsilon = 10^{-10}$ is a small constant to avoid undefined values at zero.

Finally, the weighted metric M is computed as:

$$M = w_C \cdot \log(C_{\text{norm}} + \epsilon) + w_D \cdot \log(D_{\text{norm}} + \epsilon)$$

where w_C and w_D are the weights for coherence and diversity, respectively.

The log transformation is applied to both the coherence and diversity scores for several reasons. First, it helps to handle skewed distributions by compressing large values and expanding smaller ones, which ensures that extreme values do not dominate the final metric. This makes the scores more comparable. Additionally, the log transformation avoids issues with zero values by introducing a small constant ϵ, preventing undefined values.

The weights w_C and w_D are introduced to provide flexibility in balancing the relative importance of coherence and diversity. Depending on the evaluation goals, one may prioritize more coherent topics or more diverse topics. The weights ensure that both metrics contribute meaningfully to the final score, preventing one from overshadowing the other. This allows for fine-tuning of the metric based on the specific requirements of the task or dataset being modeled.

By maximizing the weighted metric, we aim to find the hyperparameters that produce the most coherent and diverse topics. The hyperparameters we will tune include the number of topics, the parameters of the dimensionality reduction algorithm, the clustering algorithm, and other hyperparameters which will be explicitly defined in section 3.3.

3.3.5 Limitations

It is important to note that the automated evaluation metrics and baselines have their limitations. Automated metrics such as topic coherence and diversity provide a quantitative measure of topic quality but may not fully capture the nuances of human judgment. These metrics are based on statistical properties of the topics and may not always align with human perceptions, even though they are widely used in the literature. We already mentioned the limitations of these metrics in section 2.5.2.

3.4 Human evaluation

After designing our pipeline, we will perform a human evaluation to assess the quality of the tags produced by our model. As previously discussed, automated evaluation metrics offer only a limited perspective on the quality of the generated tags. Human evaluation is crucial for providing a more comprehensive assessment. To this end, we will conduct a user study in which participants will evaluate the quality of the tags generated by our model.

3.4.1 Evaluation design

Research questions

Our human evaluation aims to address the following research questions:

- Q1. Is the model good at generating individual tags relevant to the themes in the documents (*relevance*)?
- Q2. Is the model good at producing a good distribution between specific tags and general tags per document (*generality*)?
- Q3. Is the model good at covering the range of themes in the document (*coverage*)?
- Q4. Is the model good at capturing common tags between documents (*shared coverage*)?
- Q5. Does the model exhibit *robustness* by consistently producing high-quality tags across different documents and domains?

Questions Q1-Q4 will be directly addressed through our human evaluation tasks. Q5 is a broader question that our study will provide insights into, although a comprehensive answer to this question may require additional research beyond the scope of this evaluation.

Robustness in this context refers to the model's ability to consistently generate relevant, specific, general, and comprehensive tags across a wide range of documents and domains. While our study will provide initial insights into robustness, fully addressing this question would require evaluation across a larger and more diverse set of OpenML datasets.

Participants

We will begin by recruiting participants for the user study. Given that participants will be selected based on accessibility, ths will constitute a convenience sample. Colleagues, friends, and acquaintances will be invited to participate. Despite the use of a convenience sample, we will aim to recruit individuals whose backgrounds align with those of OpenML's target users. Specifically, we will seek participants with expertise in data science, computer science, or, at a minimum, individuals with a bachelor's degree and a high proficiency in English.

Materials

The study will utilize selected OpenML dataset descriptions as texts. Three separate surveys will be created, each containing the same set of texts but with different sets of tags:

1. A survey with tags generated by the *proposed model* (section 3.2.1).
2. A survey with tags generated by the *baseline model.* As a baseline, we will use the tags generated by the Bayesian optimization-tuned Base BERTopic model (section 3.2.1). This will allow for a direct comparison of the proposed model's performance against an established approach to tag generation. We choose to use it as a baseline, since BERTopic has SOTA results in the literature, and we have optimized it using Bayesian optimization.
3. A survey with *human-generated tags.*

This design allows for a direct comparison of tag quality across the relative performance of the proposed model compared to both a baseline model and human-generated tags, where the baseline model represents an established approach to tag generation, and human-generated tags serve as a gold standard for comparison.

Evaluation criteria

Participants will evaluate tags based on four main criteria, each addressing a specific research question:

- *Relevance* (Q1): How well each tag represents the main themes of the document, rated on a 1-5 scale:
  - 1 - Not at all
  - 2 - Somewhat well
  - 3 - Moderately well
  - 4 - Very well
  - 5 - Extremely well

  Example: A document about Covid-19 with tags *Covid-19*, *Virus*, *Car Crash*. *Covid-19* would be rated as 5, *Virus* as 4, and *Car Crash* as 1.

- *Generality* (Q2): How general or specific the tag is to the particular document, rated on a 1-5 scale:
  - 1 - Very specific to this document
  - 2 - Somewhat specific
  - 3 - Balanced
  - 4 - Somewhat general
  - 5 - Very general, could apply to many documents
  - 6 - Not applicable (if the tag is not relevant to the document at all)

  Example: A document about Covid-19 with tags *Covid-19*, *Virus*, *Medicine*, *Science*. *Covid-19* would be rated as 2, *Virus* as 3, *Medicine* as 4, and *Science* as 5.

- *Coverage* (Q3): How well the set of tags covers the range of themes within the document, rated on a 1-5 scale. If a participant does not rate this as 5, they will be prompted to suggest additional tags that would improve coverage. Example: A document about Covid-19 Biotechnology Companies with tags *Covid-19*, *Virus*, *Medicine*, *Science* (but missing *Biotechnology*).

- *Shared Coverage* (Q4): How well the common tags represent shared themes between two documents, rated on a 1-5 scale. Example: For datasets about Covid-19 Biotechnology Companies and Covid-19 World Vaccination Progress, common tags might include *Covid-19*, *Virus*, *Medicine* (but not *Biotechnology* or *Vaccinations*).

The goal is to measure relevance and coverage (higher is better), and to assess the distribution of generality. For generality, a low or high score is unimportant. What is important is having a larger standard deviation and a balanced distribution, indicating a good mix of specific and general tags, since we want the specific tags to capture the details of the document, connecting it with other documents that are highly similar, and the general tags to connect it with documents on a broader level. Shared coverage aims to evaluate the model's ability to capture common themes across related documents, which is important for dataset discoverability.

Variables

Our evaluation design involves several types of variables.

Independent Variables (IVs): The primary independent variable in this study is the *Tag Set*. This refers to the different sets of tags provided to participants for rating across the three surveys (proposed model, baseline model, and human-generated tags). We will investigate how these different tag sets impact the dependent variables.

Dependent Variables (DVs): The dependent variables are the outcomes we are measuring, specifically the ratings participants provide for the tags. These include:

- Relevance ratings
- Generality ratings
- Coverage ratings
- Shared coverage ratings

Controlled Variables (CVs): To ensure the validity of our comparisons, we will control the following variables:

- *Texts*: The same dataset descriptions will be used across all surveys.
- *Survey Structure*: The instructions, layout, and format will be consistent across all surveys.
- *Rating Scale*: The same 1-5 scale will be used for all rating tasks.
- *Participants*: While not all participants will complete all surveys due to resource constraints, we will ensure that the participant pools for each survey are comparable in terms of expertise and background.

By manipulating the independent variable (tag set) while controlling for other factors, we aim to isolate the effect of different tag generation methods on the quality of tags as perceived by human evaluators.

Procedure

The evaluation will be conducted in two stages: *Individual Document Evaluation* and *Document Pair Evaluation*.

In the first stage, participants will perform two tasks.

The *Intruder Detection Task* will be the first task, which presents participants with a document and a set of tags, including one intruder tag, which they must identify. *Intruder Detection* is a common task in topic modeling studies, where human evaluators are asked to identify the term that does not belong to the topic [62, 65, 67, 69, 84, 85]. It does not directly address our research questions but is a standard task in topic modeling evaluation for assessing the coherence of topics.

Following the first task is the *Tag Quality Assessment Task*, where participants will rate each tag on its relevance and generality using the 1-5 scales, as well as rate the overall coverage of the tag set. Participants will be prompted to provide additional tags that would improve coverage if they rate the coverage as less than 5.

The second stage, *Document Pair Evaluation*, also consists of two tasks.

In the *Common Tags Identification Task*, participants will be presented with two related documents and their respective tag sets, and asked to identify the common tags between the two sets.

This will be followed by the *Common Tags Quality Assessment Task*, where participants will rate the shared coverage of the common tags using a 1-5 scale. In a similar manner to the first stage, participants will be prompted to suggest additional tags that would improve shared coverage if they rate it as less than 5.

These tasks are designed to address the specific research questions:

- The *Intruder Detection Task* and *Tag Quality Assessment Task* address Q1 (relevance), Q2 (generality), and Q3 (coverage).
- The *Common Tags Identification Task* and *Common Tags Quality Assessment Task* address Q4 (shared coverage).

Additionally, these tasks will help evaluate:

- *Robustness*: If the model consistently produces pairs of tag sets for related documents where the intersection is easily identifiable, demonstrating robustness in tag generation across documents.

It is important to note that while this human evaluation provides valuable insights, it cannot fully address questions of robustness across all OpenML datasets due to resource constraints. The evaluation of consistent performance across a large number of datasets remains a challenge for future work.

Data collection

Data will be collected through Google Forms. Participants will submit their responses for each task, including identified intruder tags, relevance and generality ratings for individual tags, coverage ratings for tag sets, identified common tags between document pairs, and shared coverage ratings for common tags.

Analysis plan

The analysis will encompass a range of statistical techniques to evaluate the performance of the proposed model. We will calculate true positive and false positive rates for intruder detection, and compute average relevance, generality, and coverage scores. The distribution of these scores will be analyzed to understand the spread and variability of the ratings.

Correlation analysis will be performed to explore relationships between relevance, generality, and coverage. We will also compare scores between regular tags, overarching tags, and keyword tags to identify any significant differences.

For the common tag identification task, we will calculate true positive, true negative, false positive, and false negative rates. Inter-rater reliability analysis will be conducted to assess the consistency of ratings across participants. Hypothesis testing will be employed to compare group means, medians or variances, and effect sizes will be calculated to quantify the magnitude of any differences found.

Ethical considerations

All participants will be provided with information about the study's purpose and procedures, and their informed consent about sharing their email will be obtained before participation. Data privacy will be ensured by anonymizing and securely storing participant responses.

The study poses minimal risk to participants as it involves only text evaluation tasks.

Hypotheses

We propose the following hypotheses for this study:

The *proposed model* will generate tags with higher relevance scores compared to the baseline model. We expect to see a more balanced distribution with a larger standard deviation on generality from the proposed model compared to the baseline model, indicating a good mix of specific and general tags. The coverage scores for the proposed model's tag sets are hypothesized to be significantly higher than those of the baseline model.

Furthermore, we anticipate that the shared coverage scores for the proposed model's common tags will be significantly higher than those of the baseline model. We hypothesize a positive correlation between tag relevance and coverage scores. Finally, we expect some inter-rater reliability for tag evaluations, indicating somewhat consistent judgments across participants.

We expect the proposed model to score lower than the human-generated tags on all evaluation criteria, as human-generated tags are considered the gold standard.

These hypotheses will guide our analysis and help us evaluate the effectiveness of the proposed model in generating high-quality, relevant, and comprehensive tags for dataset descriptions.

3.4.2 Large-scale automated evaluation

To complement our human evaluation and to assess the model's performance across a larger number of OpenML datasets, we propose using a large language model (LLM) as a proxy for human evaluators. This approach is inspired by recent literature demonstrating the effectiveness of LLMs in simulating human judgments for certain tasks [84], including intruder detection.

We will use an LLM to evaluate all available OpenML dataset descriptions, both with tags generated from the baseline model and tags generated by the proposed model. This automated evaluation will consist of two main tasks.

Automated Intruder Detection

For each OpenML dataset description, we will present the LLM with the description and a set of tags, including one intruder tag. The LLM will be tasked with identifying the intruder tag.

Automated Tag Quality Assessment

For each dataset, we will prompt the LLM to rate the relevance, generality, and coverage of the tags using the same 1-5 scale as in the human evaluation.

Rationale

This large-scale automated evaluation will allow us to assess the model's performance across a much larger and more diverse set of datasets, providing insights into the model's robustness. Additionally, it will enable us to generate a large amount of evaluation data quickly and cost-effectively.

However, it's important to note the limitations of this approach. LLMs, while sophisticated, are not perfect proxies for human judgment. The automated evaluation will not include the document pair tasks from the human evaluation, as the number of possible combinations of document pairs is prohibitively large ($O(n^2)$ for n documents), limiting our ability to assess shared coverage across datasets. Moreover, the results may be influenced by biases inherent in the LLM's training data.

The results from this automated evaluation will be analyzed in conjunction with the human evaluation results to provide a more comprehensive assessment of our model's performance. This dual approach of human and automated evaluation will offer a multifaceted understanding of our model's capabilities and limitations in generating tags.

3.4.3 Limitations

Several limitations of this study should be acknowledged. The use of a convenience sample may limit the generalizability of the results. The evaluation of tags involves subjective judgments, which may introduce some variability in the results. The study focuses on a specific set of documents and may not cover all possible domains or types of datasets.

Additionally, the study provides a snapshot of tag quality at the time of conducting the study. The study evaluates the quality of the tags generated by the proposed model with the chosen hyperparameters and submodels, meaning that the results may not be generalizable to other configurations of the model. For instance, changing the embedding model at Step 4 (section 3.2.1) may impact the quality of the tags downstream, or changing the LLM in the fine-tuning step (step 9) may affect the types of tags that are generated. Since it is not feasible to evaluate all possible configurations in a human study due to resource constraints, the results may not fully capture the model's performance across all possible configurations.

To add, resource constraints may limit the number of participants and documents evaluated. Meaning that the results may not be fully representative of the model's performance across all OpenML datasets.

Although the large-scale automated evaluation will address the robustness limitation, it is not a perfect substitute for human judgment. The results may be influenced by biases in the LLM's training data and may not fully capture the nuances of human evaluation.

3.5 Chapter conclusion

This chapter presents a methodology for generating and evaluating tags for OpenML dataset descriptions. Our approach combines data exploration, preprocessing and augmentation, a novel tag generation pipeline, and an evaluation strategy combining automated and human methods.

For data preparation, we outline an approach to understanding and improving OpenML dataset descriptions. We propose an unstructured exploratory data analysis to investigate dataset characteristics and patterns to detail any preprocessing and augmentation steps that may be necessary to improve the quality of the descriptions, and to identify potential limitations of the data. For traditional topic models like LDA and NMF, we detail necessary preprocessing steps including stemming, lemmatization, stop word removal, and tokenization. Modern embedding-based models like BERTopic typically do not require these steps. We also propose augmenting descriptions with metadata such as dataset names, existing tags, column names, and information scraped from source URLs where available, in order to provide more context for the subsequent steps in the pipeline.

The core of our methodology is a ten-step pipeline that extends BERTopic. The pipeline progresses from original descriptions through description augmentation and rewriting by an LLM, followed by the creation of embeddings, dimensionality reduction, and clustering. After generating bag-of-words representations and applying c-TF-IDF, we introduce two new steps: LLM-based fine-tuning for tag generation (step 9) and zeroshot classification for tag filtering (step 10). The LLM component generates both regular tags (specific to cluster content) and overarching tags (capturing broader themes), addressing limitations of c-TF-IDF by incorporating semantic understanding beyond word frequencies. The zeroshot classifier leverages HDBSCAN's soft clustering to handle cases where descriptions belong to multiple topics.

Our pipeline offers several advantages over naive LLM approaches. It achieves contextual consistency through clustering, enables multi-level tag generation (keyword, regular, and overarching tags), provides cross-validation for the LLM fine-tuning step by using a zeroshot classifier, and improves computational efficiency by processing clusters rather than individual descriptions, reducing the number of LLM calls and token counts.

The rationale for our two-step tag generation process (LLM fine-tuning followed by zeroshot classification) is supported by examining the limitations of using either component alone. Using only the LLM (step 9) would miss the opportunity to leverage soft clustering and filter irrelevant tags, while using only the zeroshot classifier (step 10) would rely on the limited c-TF-IDF representation for tag candidates. The combination allows us to generate higher quality tags while ensuring their relevance to specific descriptions.

We acknowledge that the use of LLMs and zeroshot classifiers introduces computational overhead. To address this, we propose alternative configurations including simpler models (e.g., smaller LLMs or lightweight zeroshot classifiers [83]), pipeline modifications (removing either the LLM or zeroshot classifier), skipping description rewriting, or combining generation and filtering steps. However, we expect these alternatives to result in lower tag quality and choose to maintain the complete pipeline in this work.

For evaluation, our primary automated metrics are NPMI coherence and topic diversity, following the field's convergence on these metrics [65, 66, 67, 70]. We maintain the same reasoning outlined in section 2.6.

For baseline comparisons, we selected models representing different approaches to topic modeling: LDA and NMF as traditional statistical approaches, Top2Vec as an embedding-based model, and CTM as another neural topic model. This selection allows us to compare our approach against both established and modern techniques. Using OCTIS [73] as our evaluation framework, we implement an automated pipeline that progresses from data fetching through preparation, model training, and evaluation. We employ Bayesian optimization through OCTIS to tune our model's hyperparameters, optimizing a custom weighted metric that combines coherence and diversity scores.

Our human evaluation study is designed to assess four aspects of tag quality: relevance (how well tags represent document themes), generality (balance between specific and general tags), coverage (comprehensiveness of tag sets), shared coverage (identification of common themes across documents), and robustness (consistency of tag quality across documents and domains). We will recruit participants with relevant expertise and conduct evaluations using three sets of tags: those generated by our proposed model, a baseline model, and human annotators. This design allows direct comparison against both an established approach and the gold standard of human-generated tags.

The evaluation procedure consists of two stages: Individual Document Evaluation and Document Pair Evaluation. In the first stage, participants perform an Intruder Detection Task [62, 65, 67, 69, 84, 85] and a Tag Quality Assessment Task. The second stage involves identifying and evaluating common tags between document pairs. We control for variables such as texts, survey structure, and rating scales while manipulating the tag set as our independent variable, and setting relevance, generality, coverage, and shared coverage as dependent variables.

To complement our human evaluation and to help assess robustness, we propose a large-scale automated evaluation using an LLM as a proxy for human judgment [84]. This approach will evaluate the entire OpenML dataset, performing automated versions of the intruder detection and tag quality assessment tasks. While this allows broader coverage than human evaluation, we acknowledge that LLMs are imperfect proxies for human judgment and may be influenced by training data biases.

Each evaluation approach has distinct limitations: automated metrics may not fully capture human perceptions of topic quality, human evaluation is constrained by participant availability and cognitive load, and large-scale automated evaluation may inherit biases from the underlying LLM. However, by combining these approaches, we seek to evaluate our model from multiple perspectives, providing a more complete understanding of its effectiveness in generating meaningful tags for OpenML datasets.

This methodology addresses our research questions. **RQ1** is addressed through our data exploration, preprocessing, and augmentation approach, which will help us understand the characteristics and limitations of OpenML dataset descriptions. **RQ2** is addressed through our pipeline design, which extends BERTopic and also provides alternative configurations to balance computational cost with tag quality. The pipeline's modularity ensures it can evolve alongside advances in the field. **RQ3** is answered through our automated evaluation strategy using NPMI coherence and diversity metrics, baseline comparisons, and hyperparameter optimization via OCTIS. Finally, **RQ4** is addressed through our human evaluation study and large-scale automated evaluation, which provide an assessment of tag quality from a human perspective.

Chapter 4

Results

In this chapter, we present the results following the methodology described in Chapter 3. We first present the results of the data exploration, preprocessing and augmentation. We then present the results of the automated evaluation, comparison with baseline models, and hyperparameter tuning. Then, we show the tag generation pipeline, including the base BERTopic model, its subcomponents and hyperparameters, the additional fine-tuning model, and zeroshot text classifier. Finally, we present the results of the human evaluation, followed by the results of the large-scale automated evaluation.

4.1 Data exploration, preprocessing and augmentation

Following an initial exploratory data analysis, we identified several methods for augmenting the dataset descriptions. The dataset descriptions are augmented with the following additional information:

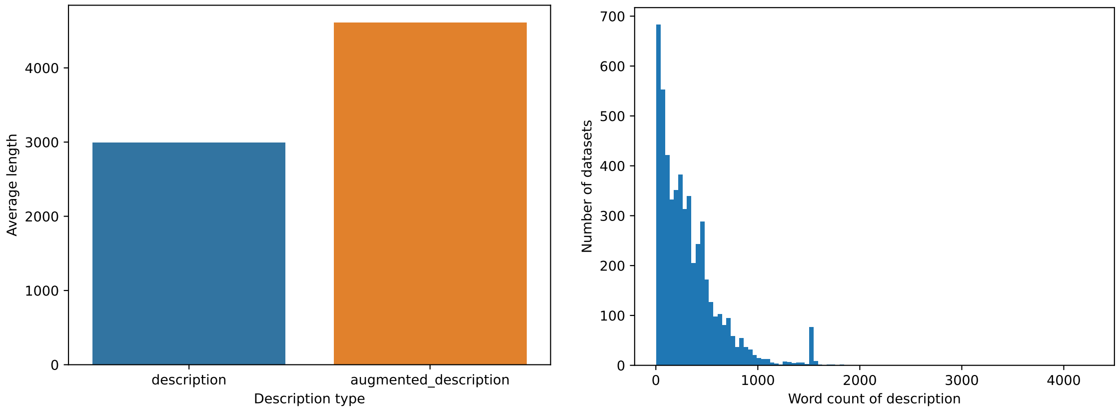
- **Name** — The name of the dataset.
- **Tags** — The generated tags that have already been created for the dataset.
- **Features** — The names of the dataset's features (columns).
- **Scraped text** — For some datasets, we scrape text from the original sources and append it to the dataset description.

Additionally, we remove all datasets that have a cosine similarity of 0.99 or higher with another dataset, as these datasets are likely to be duplicates or different versions of the same dataset. This results in the removal of $\sim$300 datasets from the original $\sim$5500 datasets.

After augmenting the dataset descriptions, we observe that the augmented descriptions are longer, and potentially more informative, as illustrated in Figure 4.1a. All subsequent analyses are performed on the augmented dataset descriptions.

Figure 4.1b presents a histogram depicting the number of words in dataset descriptions. The lengths range from 0 to 1500 words, with the majority of descriptions being under 500 words. From the histogram, we can infer that the dataset descriptions are relatively short, with the majority being comparable in length to two Twitter tweets. This is important to note, as descriptions that are too short may lack enough information to generate meaningful tags. However, this may not necessarily pose an issue, as prior studies have successfully applied topic modeling to tweets and other short texts [14, 19, 20, 86, 87, 88].

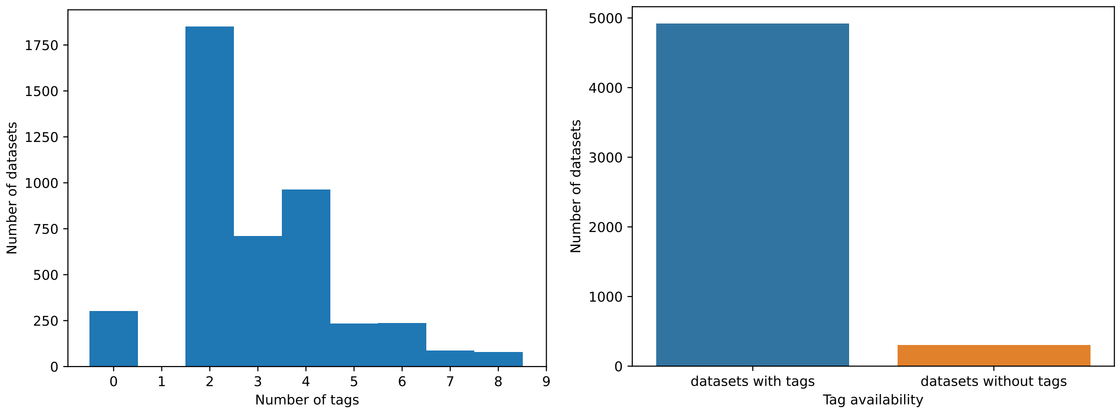
When we examine the availability of tags, we find that the majority of datasets already have tags associated with them (Figure 4.2b). This suggests that the tag generation model may be able to leverage the existing tags to improve its performance.



(a) Histogram of the length of dataset descriptions vs augmented dataset descriptions

(b) Histogram of the number of words of dataset descriptions

Figure 4.1: Analysis of dataset descriptions



(a) Histogram of the number of tags associated with datasets

(b) Histogram of the number of datasets with and without tags

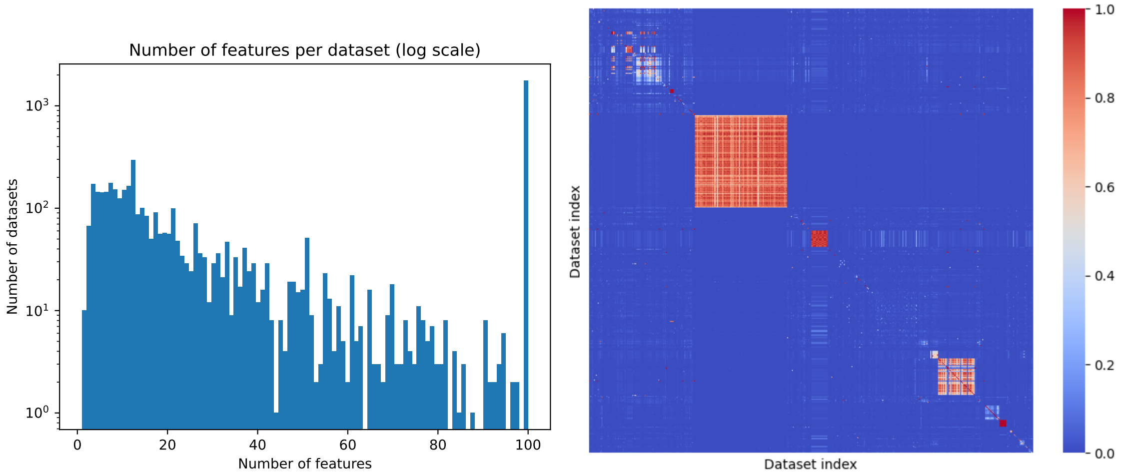
Figure 4.2: Analysis of tag distribution and availability in datasets

When we analyze the number of tags associated with each dataset, we find that most datasets have between 0 and 5 tags, with a few datasets having more than 5 tags (Figure 4.2a).

We then look at the number of features (columns) in the datasets. The distribution of the number of features is shown in Figure 4.3a. Most datasets have fewer than 100 features, with a large number of datasets having more than 100 features. This is because those datasets are likely to be high-dimensional datasets from domains such as genomics, text processing, or image analysis, where numerous variables or measurements are collected for each sample. Four tag generation pipeline, we use up to 50 features per dataset, as including all features may lead to a high-dimensional space that is difficult to model.

Additionally, we examine the cosine similarity between the augmented dataset descriptions to see whether there are datasets with similar descriptions and duplicates. The heatmap of the cosine similarity is shown in Figure 4.3b. The majority of dataset descriptions have a low cosine similarity, indicating that they are distinct from one another. However, there are some datasets with high cosine similarity, suggesting that they may be duplicates or different versions of the same dataset.

We find that OpenML datasets can have multiple versions. Our analysis of dataset versions



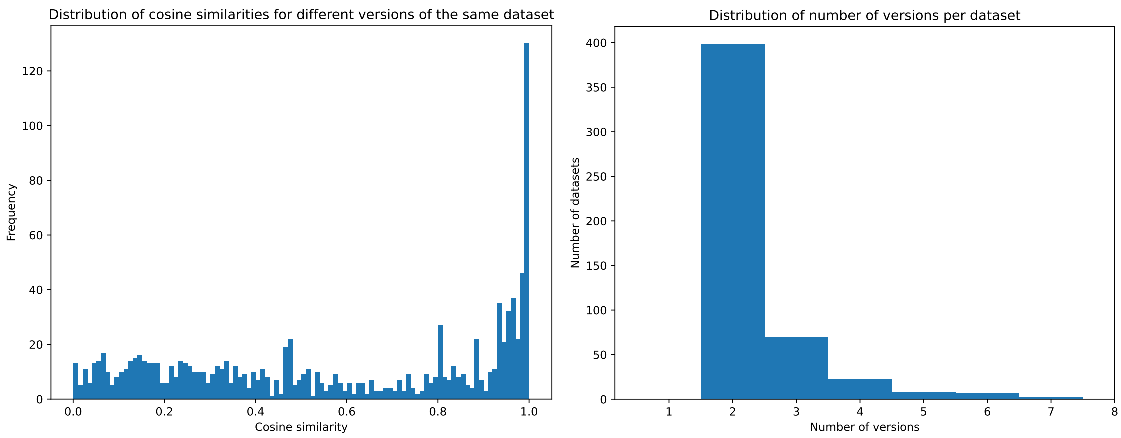
(a) Histogram of the number of features in datasets

(b) Heatmap of the cosine similarity between dataset descriptions

Figure 4.3

reveals that most datasets have only 2 versions, though several datasets have more than 2 (Figure 4.4b).

We also examine the similarity between different versions of the datasets. Our analysis shows that most datasets have a cosine similarity of 0.9 or higher within versions, suggesting that the versions are highly similar to one another (Figure 4.4a).

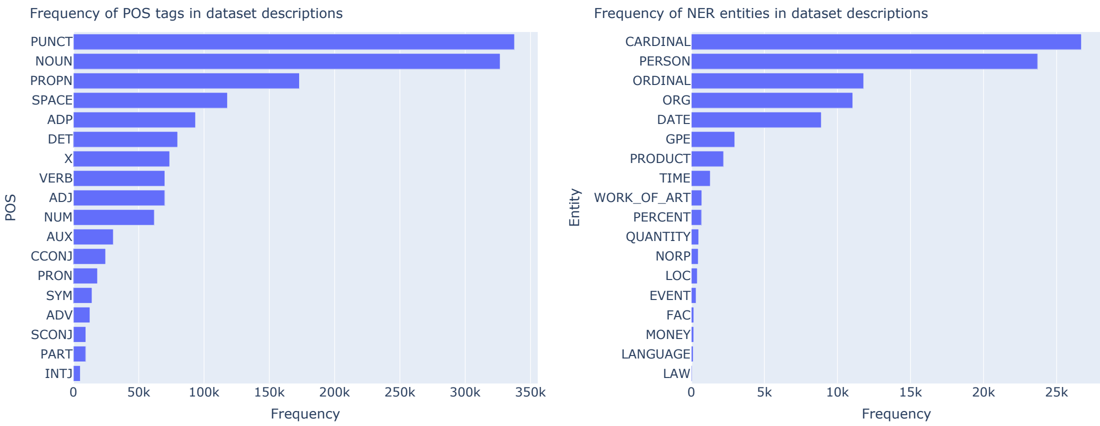


(a) Histogram of the cosine similarity of different versions of dataset descriptions

(b) Histogram of the number of versions of dataset descriptions

Figure 4.4: Analysis of dataset versions: similarity and distribution

We apply Named Entity Recognition (NER) and Part-of-Speech (POS) tagging to the dataset descriptions to identify the most common entities and parts of speech (Figures 4.5a and 4.5b). A significant number of words are tagged as *X*, indicating that the POS tagger is unable to determine the part of speech for these words. This is likely due to the presence of domain-specific terms not included in the POS tagger's vocabulary, as well as unrecognized or anomalous tokens and symbols. Additionally, we find that one of the most frequent named entities are *PERSON* and *ORG*, which can be attributed to the fact that many dataset descriptions reference the names of dataset authors or associated papers and their organizations. Both of these findings pose challenges for the performance of the tag generation model, as unrecognized tokens and author names do not contribute meaningful information for generating relevant tags.



(a) Bar chart of the most common parts of speech in dataset descriptions

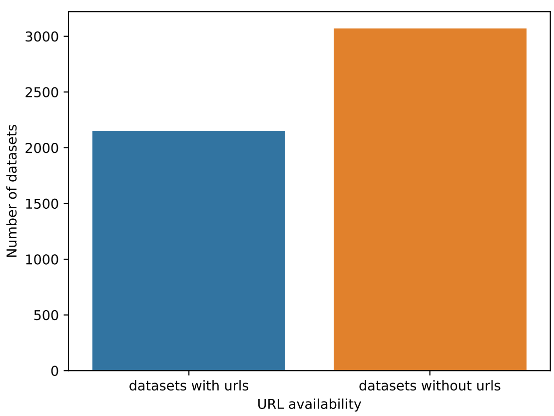
(b) Bar chart of the most common named entities in dataset descriptions

Figure 4.5: Analysis of linguistic features in dataset descriptions

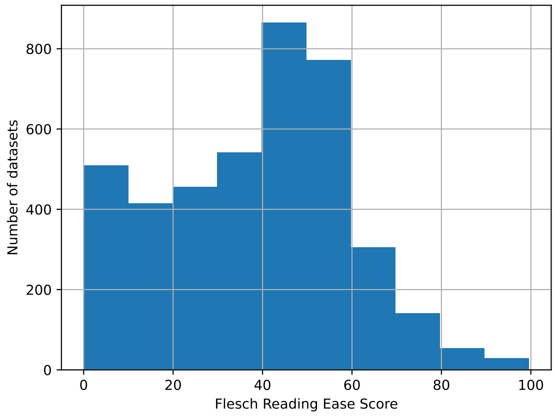
We also examine how many original datasets include URLs to original sources in their descriptions (Figure 4.6a). A significant number of datasets contain URLs, which could be used to scrape additional text for augmenting the descriptions. However, upon closer inspection, we find that the information in these URLs is often redundant with what is already provided in the dataset descriptions. Nonetheless, for some datasets, the URLs contain additional text, which is not in the original description, that could add additional information useful for generating tags. We scrape these texts and we augment the dataset descriptions with them.

We also explore the readability and complexity of the dataset descriptions by calculating the Flesch Reading Ease [89] (Figure 4.6b). The Flesch Reading Ease score ranges from 0 to 100, with higher scores indicating easier readability. We find that the dataset descriptions are somewhat difficult to read, with many falling in the 0-60 range. This suggests that the descriptions may contain complex language and jargon that could require specialized knowledge to understand. This may also affect the embedding model performance, as it may struggle to capture the semantic meaning of complex or domain-specific terms which the embedding model has not been trained on.

In this section, we described the most pertinent findings from our data exploration. For readers interested in a more detailed analysis, additional information can be found in the *eda.ipynb* notebook in the GitHub repository [90].



(a) Bar chart of the number of datasets with and without URLs to original sources



(b) Histogram of the Flesch Reading Ease scores for dataset descriptions

Figure 4.6

4.2 Automated evaluation metrics and baselines

Following section 3.3, we first explain the hyperparameter tuning (Bayesian optimization) process. Then, we explain the baseline models we use and what parameters we choose for them. We then showcase the results of the comparison between the baseline models and the hyperparameter-optimized BERTopic model.

4.2.1 Hyperparameter tuning

Using OCTIS, we perform hyperparameter tuning for the *Base BERTopic model*. As an objective metric, we use the weighted metric we defined in section 3.3.4. We set ranges for the hyperparameters, which are based on prior knowledge of good default values and the OpenML dataset characteristics. The hyperparameters and their respective search spaces are as follows:

- **min_topic_size**: This determines the minimum number of documents a topic should have. We set the range to [2, 3] to explore small topic sizes.
- **ctfidf_reduce_frequent_words**: This boolean hyperparameter determines whether frequent words should be reduced when constructing the c-TF-IDF matrix. The values considered are `True` or `False`.
- **umap_n_neighbors**: This hyperparameter controls the number of neighbors considered in the UMAP algorithm. We set the range to [2, 3]. We choose smaller values, since larger values result in more global views of the manifold, while smaller values result in more local data being preserved.
- **umap_n_components**: This controls the number of dimensions UMAP reduces the embeddings to. We have set the range to [2, 10].
- **umap_min_dist**: This defines the minimum distance between points in the UMAP embedding space. We are exploring a small range of [0.0, 0.01].
- **umap_metric**: The metric used to calculate distances between points in the UMAP algorithm. The two metrics we consider are 'cosine' and 'euclidean'.
- **hdbscan_min_cluster_size**: This defines the minimum cluster size for HDBSCAN. We set the range to [2, 3]. Similarly to `min_topic_size` and `umap_n_neighbors`, we explore small values to capture smaller clusters.

- **hdbscan_metric**: This defines the distance metric used by HDBSCAN. We restrict this to 'euclidean'.
- **hdbscan_cluster_selection_method**: This determines how clusters are selected in HDBSCAN. We use *eom* (excess of mass).
- **vectorizer_ngram_range**: This defines the n-gram range for the vectorizer. We consider both unigram (1,1) and bigram (1,2) settings.
- **vectorizer_stop_words**: We set the stop words for vectorization to be 'english'.
- **vectorizer_tokenizer**: This boolean hyperparameter controls whether a custom tokenizer is used. We set this to `False`.
- **outliers_strategy**: This defines how outliers should be handled. For this study, we set it to *none*, meaning that no specific outlier handling strategy is applied.
- **embedding_model**: This hyperparameter defines the embedding model used for the *Base BERTopic model*. Since they are slower to evaluate, we only consider the `Salesforce/SFR-Embedding-2_R` [3] model, as it is one of the best performing models on the MTEB leaderboard [4] at the time of writing.
- **representation_model**: This hyperparameter defines the representation model used for the BERTopic model. Since they are slower to evaluate, we only consider spaCy's Part-of-Speech model called `en_core_web_lg`.

This automated hyperparameter optimization process allows us to systematically explore the space of possible configurations and identify the best-performing model for our dataset.

As for the Bayesian optimization [74, 75, 76], we employ the `Optimizer` class from OCTIS. This method efficiently explores the hyperparameter space by constructing a probabilistic (surrogate) model to approximate the objective function. In our case, the surrogate model is a Random Forest (RF), which is updated iteratively to predict the performance of unobserved hyperparameter configurations based on previous evaluations.

The surrogate model relies on the Matern kernel with a smoothness parameter $\nu = 1.5$. This kernel helps balance the trade-off between exploration and exploitation by controlling the smoothness of the Gaussian process used in the optimization.

To guide the search for optimal hyperparameters, we use the Lower Confidence Bound (LCB) acquisition function. The LCB acquisition function is particularly effective in encouraging exploration of uncertain regions while still exploiting areas that show high potential for improvement.

Before the optimization begins, we initialize the surrogate model with a diverse set of hyperparameter configurations. These initial points are generated using Latin Hypercube Sampling (LHS), which ensures a broad coverage of the search space from the outset.

Additionally, we set the number of iterations for the optimization process to 125, and the number of model runs per iteration to 10. This configuration allows us to explore the hyperparameter space thoroughly while maintaining a reasonable computational cost. We set the model runs to 10 to account for the stochastic nature of some of the models, e.g. BERTopic, which may yield different results each time it is run primarily due to UMAP. Furthermore, having 10 runs per iteration allows us to identify statistically significant differences between the models.

Figure 4.7 shows the results of the Bayesian optimization process. The x-axis represents the number of iterations, while the y-axis represents the weighted metric score, *Median(model_runs)* and *Mean(model_runs)*, and other metrics.

We can see that the optimization process converges to a stable solution relatively quickly (looking at *Mean(model_runs)*). This indicates that the hyperparameter search space has been thoroughly explored, and a good configuration has been found. It is also worth noting that the starting state of the optimization process is relatively close to the final state, suggesting that the initial default hyperparameter configurations are already performing well, meaning that the optimization process does not need to make large adjustments to find a good solution. This indicates that the BERTopic model is robust to changes in hyperparameters and can achieve good performance across a wide range of settings on the OpenML dataset.

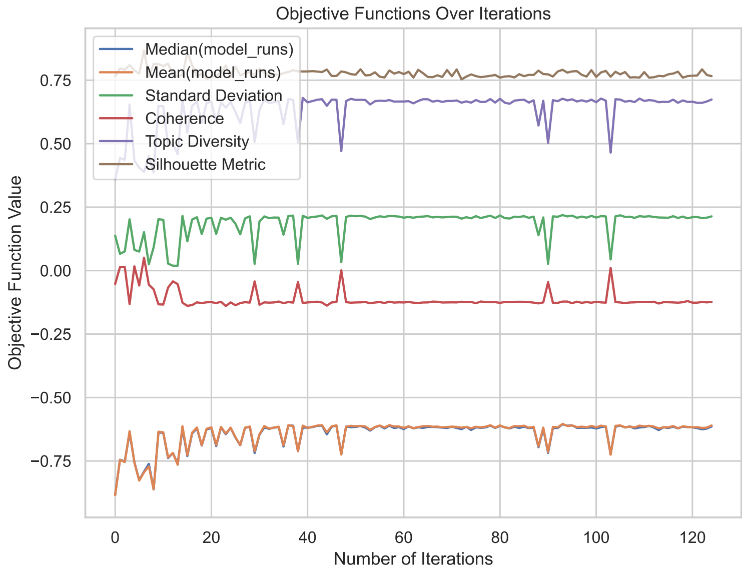


Figure 4.7: Results of the Bayesian optimization process

Table 4.1 shows the hyperparameters we explored and the optimal values found by the Bayesian optimization process.

Parameter	Search Space	Optimal Value
min_topic_size	[2, 3]	2
ctfidf_reduce_frequent_words	{True, False}	True
umap_n_neighbors	[2, 3]	3
umap_n_components	[2, 10]	5
umap_min_dist	[0.0, 0.01]	0.008
umap_metric	{cosine, euclidean}	euclidean
hdbscan_min_cluster_size	[2, 3]	3
hdbscan_metric	euclidean	euclidean
hdbscan_cluster_selection_method	eom	eom
vectorizer_ngram_range	{(1,1), (1,2)}	(1, 1)
vectorizer_stop_words	english	english
vectorizer_tokenizer	False	False
outliers_strategy	none	none

Table 4.1: BERTopic Hyperparameter Settings and Optimal Values

4.2.2 Baselines

Definition of Baselines

We select the hyperparameter-optimized BERTopic model as our primary model and compare its performance against several baseline models. The baseline models are as follows:

1. **BERTopic model with default parameters**: This model uses the default hyperparameters provided by the BERTopic library. We use this model to compare the performance of the hyperparameter-optimized model with the default settings.

2. **LDA**: We fit an LDA model to the dataset descriptions using the `LDA` class from OCTIS. LDA is a model which requires cleaning and preprocessing of the text data, such as removing stop words, stemming, and lemmatization. We use the default settings for the LDA model.

3. **NMF**: We fit an NMF model to the dataset descriptions using the `NMF` class from OCTIS. We use the default settings for the NMF model.

4. **CTM**: We fit a CTM model to the dataset descriptions using the `CTM` class from the Contextualized Topic Models library [10]. We preprocess the data and create the required embeddings with the `all-mpnet-base-v2` embedding model and use the default settings for the CTM model with the `contextual_size` parameter set to 768.

5. **Top2Vec**: We fit a custom implementation of Top2Vec based on the original model. This implementation uses the `all-mpnet-base-v2` embedding model from the Sentence Transformers library for document and word embeddings. The HDBSCAN clustering algorithm is used with custom arguments set in the `hdbscan_args` parameter. We use the `Top2VecNew` class from our custom implementation, which allows for more flexibility in topic number specification.

For each model, we train using a predefined set of topic numbers. This approach is necessary because some models require a fixed number of topics, while others, such as BERTopic, can either operate with a fixed number or determine the number of topics automatically. Specifically, we set the number of topics to 10, 20, 30, 40, 50, 100, and 200.

Additionally, for each specified number of topics, we run each model 10 times to account for their stochastic nature. This approach allows us to later assess whether there are statistically significant differences in the models' performance.

Results

In Figure 4.8b we present the NPMI scores for the hyperparameter-optimized BERTopic model and the baseline models. We remind the reader that we use the Base BERTopic model, not the proposed model, for this comparison. There are several BERTopic models with different hyperparameters:

- **BERTopic_optimized_POS_reduced_range**: The BERTopic model we optimized with Bayesian optimization above.

- **BERTopic_optimized_POS_full_range**: Similar to the previous model, also optimized with Bayesian optimization, but with a wider range of hyperparameters. We choose not to use this model as we are interested in exploring smaller topic sizes, but we provide it for comparison in the case where we do not restrict the hyperparameter search space.

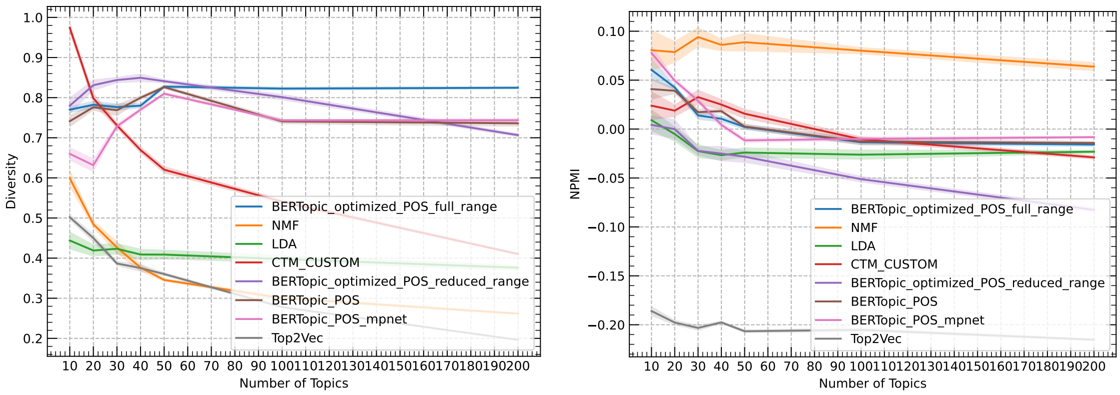
- **BERTopic_POS**: The BERTopic model with default hyperparameters (also with the spaCy Part-of-Speech model as the representation model). This is the default model provided by the BERTopic library, without any hyperparameter optimization. We include this model to compare the performance of the hyperparameter-optimized model with the default settings.

- **BERTopic_POS_mpnet**: Similar to the model above, but with the `all-mpnet-base-v2` embedding model instead of `Salesforce/SFR-Embedding-2_R` [3]. We choose this model to explore the impact of different embedding models on the BERTopic model, since `all-mpnet-base-v2` is a relatively small model compared to `Salesforce/SFR-Embedding-2_R` [3], which is one of the best performing models on the MTEB leaderboard [4].

We choose these four variations of the BERTopic model to explore the impact of different hyperparameters on the model's performance.

We observe that the NMF model outperforms all models across all topic numbers. However, when we look at Figure 4.8a, we see that the NMF model has very low diversity scores. This suggests that the NMF model may be picking the same terms in each topic, leading to high coherence but low diversity, as explained in the coherence-diversity trade-off in section 2.6. For NPMI, we then see that the BERTopic models perform the second best, with small differences between them. We see that **BERTopic_optimized_POS_reduced_range** performs slightly worse than the other BERTopic models, but this is likely due to the smaller topic sizes we explored, since this leads to a higher number of smaller clusters, leading to slightly lower coherence, but higher diversity. We see that LDA, CTM, and Top2Vec perform worse than the BERTopic models.

As for diversity, we see that the **BERTopic_optimized_POS_reduced_range** has the highest average diversity scores, followed by the other BERTopic models. This suggests that the BERTopic models are able to capture a wider range of terms in their topics compared to the other models. Surprisingly, the Top2Vec model has the lowest NPMI and diversity scores.



(a) Line chart of the diversity scores for the hyperparameter-optimized BERTopic model and the baseline models

(b) Line chart of the NPMI scores for the hyperparameter-optimized BERTopic model and the baseline models

Figure 4.8: Comparison of diversity and NPMI Scores for different topic models

In Table 4.2, we present the averaged NPMI and diversity scores for the models. We highlight the best scores in dark green and the subsequent best scores in lighter green. We see that the BERTopic models perform well in terms of diversity, with the **BERTopic_optimized_POS_reduced_range** model having the highest diversity score. If we only look at the BERTopic models, we can note a strong negative Pearson correlation between NPMI and diversity scores of -0.67. Surpisingly, **BERTopic_POS_mpnet** has the highest NPMI score (after NMF), even though it uses a smaller embedding model. However, we can see it has a lower diversity score compared to the other BERTopic models. This suggests that the higher coherence may be driven by the inclusion of more common terms in the topics, while the diversity score is driven by the inclusion of more unique terms, as explained in section 2.6. In any case, the BERTopic models outperform the baseline models in terms of combined NPMI and diversity scores.

Looking at the combined scores (NPMI + diversity), the top three performing models are **BERTopic_optimized_POS_full_range** with a score of 0.812, followed by **BERTopic_POS** with 0.783, and then **BERTopic_optimized_POS_reduced_range** with 0.779. This is expected, as for the first model, we explore a wider range of hyperparameters, which allows for more flexibility in the model's performance. For the second model, we use the default hyperparameters, which are already performing well. For the third model, we explore smaller topic sizes, which leads to a higher number of smaller clusters, leading to slightly lower coherence, but higher diversity.

Model	npmi	diversity
Top2Vec	-0.202	0.364
BERTopic_optimized_POS_reduced_range	-0.029	0.808
LDA	-0.017	0.411
CTM_CUSTOM	0.011	0.678
BERTopic_POS	0.013	0.770
BERTopic_optimized_POS_full_range	0.014	0.798
BERTopic_POS_mpnet	0.019	0.727
NMF	0.082	0.399

Table 4.2: Average NPMI and diversity scores for the hyperparameter-optimized BERTopic model and the baseline models

Statistical significance

As mentioned earlier, we had 10 runs for each model and topic number combination. We will now assess whether there are statistically significant differences between the performance of the models. We check whether the assumptions for ANOVA are met, and if not, we use Welch's ANOVA or Kruskal-Wallis (non-parametric) tests.

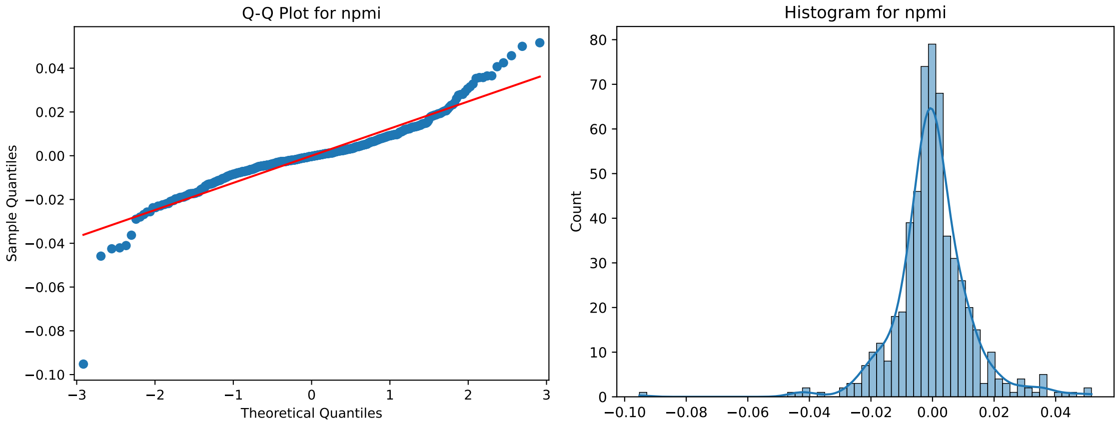
The first assumption for ANOVA is that the residuals are normally distributed. We apply the Shapiro-Wilk test to check whether the residuals for the NPMI values (the differences between the observed and predicted NPMI values) are normally distributed. The Shapiro-Wilk test returns a statistic of 0.905 and a p-value of 3.30e-18, indicating a significant deviation from normality. Although the statistic is relatively close to 1, suggesting that the residuals are not drastically non-normal, the extremely small p-value suggests that the deviation is statistically significant. As for the diversity values, the Shapiro-Wilk test returns a statistic of 0.968 and a p-value of 8.70e-10, also indicating a significant deviation from normality, but with a statistic even closer to 1.

Given this result, we further inspect the residuals using visual diagnostics with Q-Q plots and histograms to assess the nature and extent of the deviation from normality. If the deviation is minor, ANOVA may still be appropriate. Figure 4.9a and Figure 4.9b show the Q-Q plot and histogram of the residuals for the NPMI values, respectively. We observe that the residuals are approximately normally distributed, with some deviations at the tails. Figure 4.10a and Figure 4.10b show the Q-Q plot and histogram of the residuals for the diversity values, respectively. We observe a similar pattern for the diversity residuals, with some deviations at the tails. Given that ANOVA is relatively robust to deviations at the tails, since extreme values do not have a large impact on the F-statistic, we consider this assumption to be met.

The second assumption for ANOVA is that the residuals have equal variance. We apply the Levene test and the Bartlett test to check whether the residuals have equal variance, for each model and topic number combination. Table 4.3 shows the results of the Levene and Bartlett tests for the NPMI and diversity values. We observe that the p-values are mostly below 0.05, indicating that the residuals do not have equal variance. This violates the assumption of equal variance for ANOVA. Therefore, we will use Welch's ANOVA, since the first assumption is met, and the residuals are approximately normally distributed, but the assumption of equal variance is violated.

Applying Welch's ANOVA to the NPMI and diversity values, we find that there are statistically significant differences between the models for both metrics (Table 4.4). For NPMI, 90.75% of the variance is explained by the model, and for diversity, 81.50% of the variance is explained by the model. These are both large effect sizes, suggesting that the models have a substantial impact on both metrics.

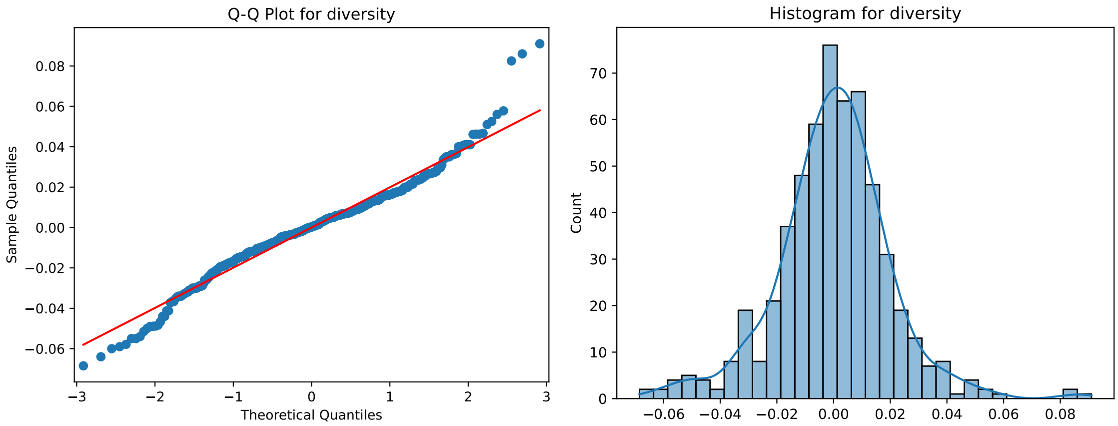
We then perform post-hoc tests using the Games-Howell method to determine which models are significantly different from one another, as this test is appropriate when the assumption of equal variance is violated. The results of the Games-Howell post-hoc test for the NPMI values are presented in Table C.1. We observe a notable homogeneity among the BERTopic models, with generally high p-values indicating a lack of statistically significant difference in their coherence



(a) Q-Q plot of the residuals for the NPMI values

(b) Histogram of the residuals for the NPMI values

Figure 4.9: Analysis of residuals for NPMI values



(a) Q-Q plot of the residuals for the diversity values

(b) Histogram of the residuals for the diversity values

Figure 4.10: Analysis of Residuals for Diversity Values

scores. The exception is `B_OPT_REDUCED`, which was configured to explore smaller topic sizes, showing statistically significant differences from other BERTopic models, suggesting that restricting the topic size in this way might negatively impact topic coherence. While NMF achieved the highest average NPMI score, the post-hoc test shows a statistically significant difference between NMF and all BERTopic models. The relatively high NPMI of `BERTopic_POS_mpnet` is worth noting, as this variant utilizes a smaller embedding model compared to others, suggesting that the choice of the embedding model might not be the primary determinant of topic coherence. The post-hoc analysis of diversity scores (Table C.2) further underscores the strength of the BERTopic approach, with all BERTopic models demonstrating significantly higher diversity compared to the baseline models. The `B_OPT_FULL` and `B_OPT_REDUCED` models exhibit the highest diversity scores. These results highlight a trade-off between coherence and diversity, suggesting that models excelling in one metric may not perform as well in the other.

Table 4.3: Levene's and Bartlett's tests for NPMI and diversity

	NPMI		Diversity	
nr_topics	**Statistic**	**p-value**	**Statistic**	**p-value**
Levene's Test				
10	1.4665	0.1930	1.6530	0.1346
20	4.0018	0.0009	1.2666	0.2791
30	2.9904	0.0082	2.0219	0.0638
40	2.6662	0.0164	1.1821	0.3239
50	2.9896	0.0082	2.5184	0.0225
100	1.0988	0.3733	1.4504	0.1990
200	2.4367	0.0268	2.6070	0.0186
Bartlett's Test				
10	19.3408	0.0072	17.4000	0.0150
20	39.6442	1.4724e-06	7.5180	0.3770
30	24.0258	0.0011	14.8213	0.0384
40	27.6841	0.0003	11.7576	0.1088
50	34.6455	1.3038e-05	19.0298	0.0081
100	9.8099	0.1996	15.5252	0.0298
200	20.3993	0.0048	31.0744	6.0240e-05

Table 4.4: Welch's ANOVA results for NPMI and diversity

Metric	Source	df1	df2	F-value	p-value	np2
NPMI	Model	7	231.601	2549.36	$3.165650 \times 10^{-215}$	0.90749
Diversity	Model	7	232.705	1091.24	$4.954462 \times 10^{-174}$	0.815016

4.3 Tag generation

In section 3.2.1 and Figure 3.2, we presented the tag generation pipeline on a high level. Figure 4.11, which is similar to Figure 3.2 shows the specifics of the tag generation pipeline. In particular, we show which specific submodels we used for the different steps in the pipeline:

1. **Original Descriptions**: Same as in the high-level pipeline, we start with the original OpenML dataset descriptions.

2. **Augmented Descriptions**: In section 4.1, we discussed how we augment the descriptions with additional information.

3. **Prompt Descriptions Human-Readable**: For this step, we used the `Llama-3-70b`, model, as it was a model offering a good balance between performance and computational resources at the time of the experiment. We engineered a prompt to extract keyword tags from each individual description. We do not include the prompt here for brevity, but it is available in the GitHub repository [90].

4. **Create Embeddings**: We use the `Salesforce/SFR-Embedding-2_R` model [3], which is one of the best performing models on the MTEB benchmark [4].

5. **Base BERTopic Model**: For dimensionality reduction, clustering, bag-of-words construction and c-TF-IDF calculation, we use the hyperparameter-optimized BERTopic model.

6. **Fine-tune to Extract Tags**: For the fine-tuning step, we use the `Llama-3-70b` model. We prompt the model to generate tags for each cluster. The prompt can again be found in the repository.

7. **Zeroshot Text Classifier**: We use the `MoritzLaurer/deberta-v3-large-zeroshot-v2.0` model [83], which at the time of the experiment was the best performing model for the zeroshot text classification task.

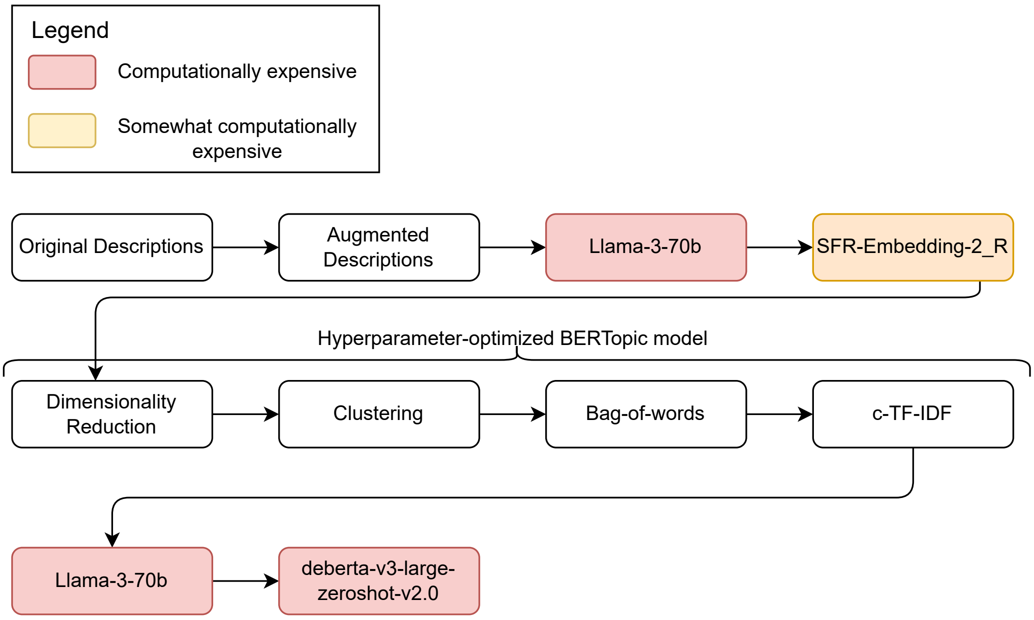


Figure 4.11: Tag generation pipeline specifics (*proposed model*)

4.3.1 Hardware

We run the tag generation pipeline, with the exception of the *Prompt Descriptions Human-Readable* step and the *Fine-tune to Extract Tags* step, on an Apple M2 Max MacBook Pro with 64 GB of internal memory. We run the *Prompt Descriptions Human-Readable* step and the *Fine-tune to Extract Tags* step on Together AI, a platform that offers a cloud-based environment for running large language models.

The LLM-based rewriting of all dataset descriptions into a more human-readable format took approximately 3 hours and cost around $8.00. Generating the embeddings using the *Salesforce/SFR-Embedding-2_R* [3] model took around 4 hours on the Apple M2 Max MacBook Pro. Training the base BERTopic model required only about 5 minutes. The LLM fine-tuning step using the Llama-3-70b model took roughly 2 hours and cost approximately $4.80. These reduced costs in the fine-tuning step compared to the rewriting step are likely due to the smaller size of the rewritten descriptions and the implementation of prompt truncation for excessively large inputs. The Llama 3 70B model provided by TogetherAI, `Meta Llama 3 70B Instruct`, was priced at $0.88 per 1M tokens at the time of this writing.

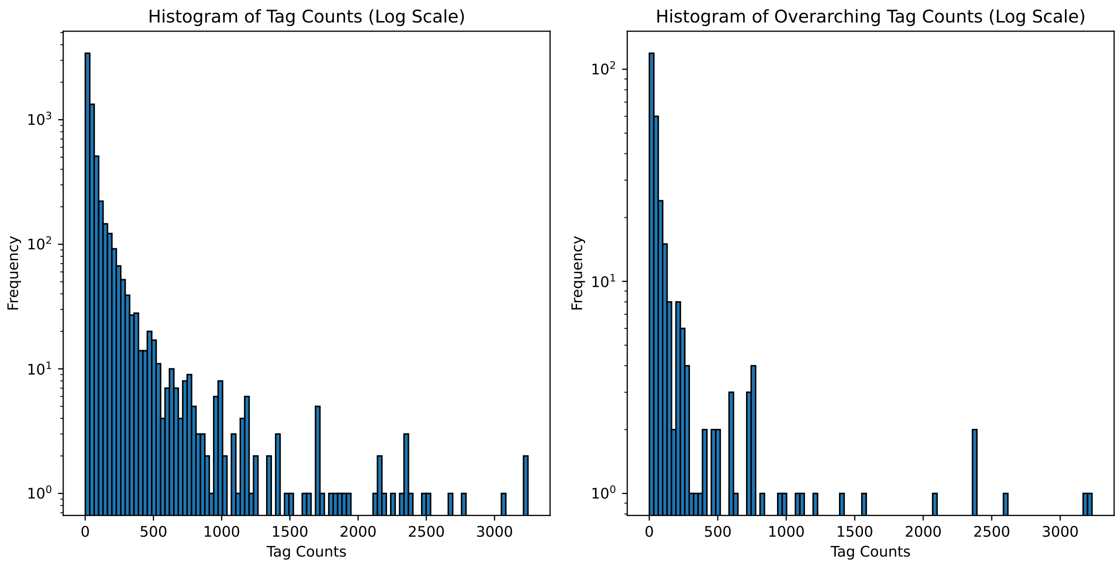
By far the most time-consuming step in the pipeline was the zeroshot classification. Despite using a relatively small model (MoritzLaurer/deberta-v3-large-zeroshot-v2.0 with a maximum context length of 8192 tokens [83]), this step took approximately 16 hours to complete. This is because each potential tag from the top n topic clusters for each dataset description is individually fed to the zeroshot classifier, generating a confidence score for each tag-description pair. This process involves a large number of individual classifications, leading to a substantial time requirement.

4.3.2 Results

We now show the results of the output of the tag generation pipeline. Approximately 6300 regular tags and 300 overarching tags were generated. It is expected that the number of regular tags is higher than the number of overarching tags, as the regular tags are more specific, while the overarching tags are more general and cover a wider range of topics.

Investigating the generated tag frequency in detail, we see that the distribution of the tags is skewed. Figure 4.12a shows the histogram of tag counts on a log scale. We see that the majority of tags have a low count, with a few tags having a very high count. This is expected, as natural language has been found to follow Zipf's law (Zipfian distribution), where a few terms are very common, while the majority of terms are rare [91, 92].

We observe a similar pattern when looking at histograms of the overarching tag counts, as seen in Figure 4.12b.

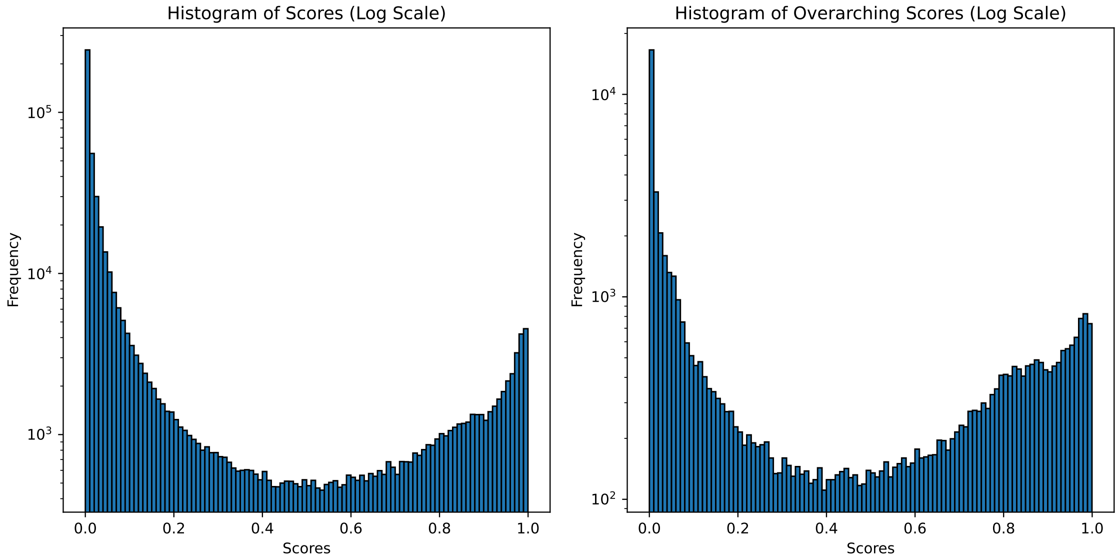


(a) Tag counts

(b) Overarching tag counts

Figure 4.12: Histograms of tag counts (log scale)

We now turn to investigating the tag scores, which are a measure from 0 to 1 that the zeroshot text classifier assigns to each tag based on each individual description. Figure 4.13a shows the histogram of tag scores, while Figure 4.13b shows the histogram of overarching tag scores. We see that the distribution of scores is skewed, with a few tags having a very low score (close to 0) and a few tags having a very high score (close to 1), with the remaining tags having scores in between. This means that the zeroshot text classifier is filtering out tags that are not relevant to the descriptions, while assigning high scores to tags that are relevant.



(a) Tag scores (b) Overarching tag scores

Figure 4.13: Histograms of tag scores

4.4 Human evaluation

In this section, we present the results of the human evaluation based on the definition of the study design in section 3.4.

4.4.1 Materials

The same dataset descriptions were used across all three surveys to maintain consistency. Based on a pilot study, we selected three texts for the *Individual Document Evaluation* stage and two for the *Document Pair Evaluation* stage. The selection criteria focused on appropriate length and complexity to ensure participant comprehension. This number of texts allowed participants to complete the survey within approximately 45 minutes, balancing the need for sufficient data collection while preventing participant fatigue.

For brevity, we include only a few examples of the text-tag pairs used in the surveys. All three surveys can be found in the GitHub repository [90] under the names of `proposed_model_survey.pdf`, `baseline_survey.pdf`, and `human_generated_survey.pdf`.

An example text from the first stage, *Individual Document Evaluation*:

> **FOREX USD/JPY Minute High**
>
> This dataset contains historical price data of the FOREX USD/JPY from Dukascopy. Each instance, or row, represents one candlestick of one minute. The dataset spans from January first to December thirteenth and does not include weekends, as the FOREX market is not traded on weekends. The timezone of the feature Timestamp is Europe/Amsterdam.

This text was contained in all three surveys, and for each survey, a different set of tags was provided. Table 4.5 shows a comparison of tags for different models.

For the first task, *Intruder Detection*, an intruder tag, which the participant had to identify, was added to the set of tags for each text. The intruder tag was selected at random from the generated tags of another OpenML dataset.

Proposed Model	Human-Generated	Baseline Model
Historical Price Data	Historical Price Data	Thyrotropin
Minute Interval	Forex	Minute
Historical Data	USD/JPY	USD
Forex	Currency Pairs	Releasing
Candlestick	Yearly Data	High
Minute	Finance	Bid
High	Minute High	Ask

Table 4.5: Comparison of tags for different models

In the second task, *Tag Quality Assessment*, for each tag, participants were asked to rate the relevance and generality. For each tag set, participants were asked to rate the coverage.

In the second stage, *Document Pair Evaluation*, participants were provided pairs of texts. For example, one pair consisted of the following two texts:

> **Movies on Netflix, Prime Video, Hulu, and Disney+**
>
> This dataset is an amalgamation of data that was scraped, comprising a comprehensive list of movies available on various streaming platforms, and the IMDb dataset, which provides inspiration for analysis.
>
> Which streaming platform or platforms can I find this movie on? This dataset allows us to explore the availability of movies across different streaming services. Additionally, we can examine the average IMDb rating of movies produced in a specific country, providing insights into the quality of films from different regions.
>
> **Popular Movies of IMDb**
>
> TMDB.org is a crowd-sourced movie information database widely used by various film-related consoles, sites, and apps, such as XBMC, MythTV, and Plex. Dozens of media managers, mobile apps, and social sites utilize its API. At the time of writing, TMDB lists a substantial number of films, which is considerably fewer than IMDb. While not as comprehensive as IMDb, it holds extensive information for most popular and Hollywood films.

In the first task, *Common Tags Identification*, participants were asked to identify tags that were common to both texts, i.e., the intersection of the two tag sets. For instance, for the two tag sets in Table 4.6, the common tags were *Film*, *Entertainment*, *Movies*, and *Media* (as predicted by the model).

Movies on Netflix, Prime Video, Hulu, and Disney+	Popular Movies of IMDb
Film	Entertainment
Media	Technology
Entertainment	Film
Movies	Media
Film Industry	Movies
Streaming Platforms	Popular Movies
	Film Information

Table 4.6: Comparison of tags for two datasets

In the second task, *Shared Coverage Assessment*, participants were asked to rate the shared coverage of the tags for both texts.

4.4.2 Participants

For the *proposed model* survey, we recruited 21 participants, all of whom completed the survey. For the *baseline model* survey, we recruited 19 participants, and for the *human-generated* survey, we recruited 18 participants. There was a large overlap between the participants in the three surveys, with 93.3% of participants completing all three surveys.

As shown in Table 4.7, the majority of participants across all three surveys hold a Bachelor's degree (57.1-63.2%), followed by those with a Master's degree (19.0-22.2%). This educational background aligns with our aim to recruit participants with backgrounds similar to OpenML users. The age distribution shows that most participants (81.0-84.2%) are between 25 and 34 years old. Regarding English proficiency, the majority of participants demonstrate advanced language skills, with 71.4-78.9% at C1 (Advanced) level and an additional 10.5-19.0% at C2 (Proficient) level.

Table 4.7: Demographic characteristics of participants across surveys

Category	Characteristic	Human-generated	Baseline model	Proposed model
Education	Bachelor's degree	63.2%	57.1%	61.1%
	Master's degree	21.1%	19.0%	22.2%
	Doctoral degree	5.3%	19.0%	5.6%
	Some college	5.3%	4.8%	5.6%
	High school or equivalent	5.3%	0.0%	5.6%
Age	25-34	84.2%	81.0%	83.3%
	18-24	10.5%	9.5%	11.1%
	35-44	5.3%	9.5%	5.6%
English	C1 (Advanced)	78.9%	71.4%	77.8%
	C2 (Proficient)	10.5%	19.0%	11.1%
	B1 (Intermediate)	5.3%	4.8%	5.6%
	Native speaker	5.3%	4.8%	5.6%

4.4.3 Results

Stage 1 — Task 1: Intruder Detection

Table 4.8 presents the intruder detection results for all three texts (*League of Legends*, *Forex* and *Lung Cancer*), across all three surveys. The proposed model demonstrated substantially better performance than the baseline model (95.2% vs 42.1% overall detection rate), though it did not quite match the perfect performance achieved by human-generated tags (100% detection rate). The baseline model showed particularly poor performance on the Forex text, with only 5.3% detection rate. These results suggest that human-generated tags exhibited the highest quality, as participants consistently identified the intruder tags, indicating stronger cohesion among the human-generated tag sets.

Stage 1 — Task 2: Tag Quality Assessment

Figures 4.14a to 4.14c show the tag quality assessment results for the baseline model, the proposed model, and the human-generated tags, respectively. The histograms display the distribution of scores for relevance, generality, and coverage across all surveys (Human-generated — HG, Proposed Model — PM, Baseline Model — BM), combining data from all three texts. For each measure, we include the mean scores, standard deviations, and 95% confidence intervals.

Analysis of the scoring metrics revealed that the proposed model demonstrated superior performance compared to the baseline model. For relevance scores (higher is better), the baseline model achieved a mean of 2.39, while the proposed model achieved 3.63, which is closer to the human-generated tags' score of 3.95. Regarding generality, where larger standard deviations indicate a better balance between general and specific tags, the proposed model (mean = 2.93,

Table 4.8: Intruder detection results across different text sources and survey types

Survey Type	Text Source	Detected	Not Detected
Baseline Model	League of Legends	63.2% (12)	36.8% (7)
	Forex	5.3% (1)	94.7% (18)
	Lung Cancer	57.9% (11)	42.1% (8)
	Total	42.1% (24)	57.9% (33)
Proposed Model	League of Legends	100.0% (21)	0.0% (0)
	Forex	90.5% (19)	9.5% (2)
	Lung Cancer	95.2% (20)	4.8% (1)
	Total	95.2% (60)	4.8% (3)
Human-generated	League of Legends	100.0% (18)	0.0% (0)
	Forex	100.0% (18)	0.0% (0)
	Lung Cancer	100.0% (18)	0.0% (0)
	Total	100.0% (54)	0.0% (0)

SD = 1.40) performed better than the baseline (mean = 3.13, SD = 1.33) and nearly matched human-generated tags (mean = 3.00, SD = 1.41). In terms of coverage, the proposed model (mean = 4.46) outperformed the baseline model (mean = 2.65) and was close to the human-generated tags (mean = 4.54). However, it is notable that the baseline model had a significantly higher number of tags marked as "Not Applicable" (97 tags), compared to the proposed model (11) and human-generated tags (3). This suggests that while the baseline model's valid tags showed similar generality levels, it produced many irrelevant tags that evaluators couldn't meaningfully assess for generality.

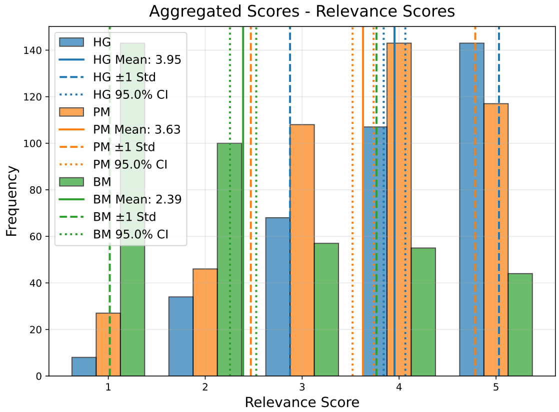
The human-generated tags achieved the best scores in relevance, generality and coverage, indicating that the human-generated tags were of higher quality. The proposed model outperformed the baseline model by a wide margin, and only somewhat lagged behind the human-generated tags. This suggests that the proposed model was able to generate better tags than the baseline model, but still not quite as high-quality as human-generated tags.

Another notable observation is the relationship between relevance and generality scores across the three tag sources. The proposed model exhibited the strongest negative correlation (-0.59), indicating that as tags became more specific, they were generally rated as more relevant. The baseline model showed a similar but moderately weaker pattern (-0.42). In contrast, human-generated tags demonstrated a much weaker negative correlation (-0.19), suggesting that human-generated tags were more relevant across both specific and general levels. For all three, we observe p-values below 0.001, indicating that the correlations are statistically significant. This balance between specificity and generality in human-generated tags indicates a more nuanced approach to tag creation, where both specific details and broader contextual tags can maintain high relevance.

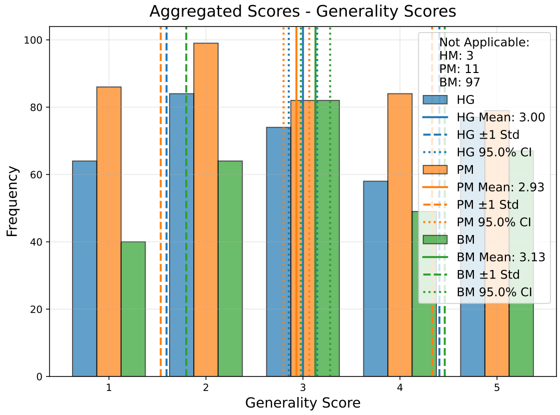
Additionally, we inspect the aggregated relevance and generality scores by tag type (regular vs. overarching tags, as explained in section 3.2.1) in Figure 4.16 for human-generated tags and in Figure 4.15 for the proposed model. We see that the relevance scores mean for the proposed model is slightly higher for regular tags (3.77) than for overarching tags (3.48), while the generality scores are higher for overarching tags (3.52) than for regular tags (2.38). This second observation indicates that the pipeline we designed was successful in generating more specific tags for regular tags and more general tags for overarching tags. For human-generated tags, we see a similar pattern, with higher relevance scores for regular tags (4.03) than for overarching tags (3.81), and higher generality scores for overarching tags (3.74) than for regular tags (2.61).

The histograms provided here only show the aggregated results across all three texts, across all tags. For readers interested in the details for each individual text or tag, additional information can be found in the *human_ evaluation_ analysis.ipynb* notebook in the GitHub repository [90].

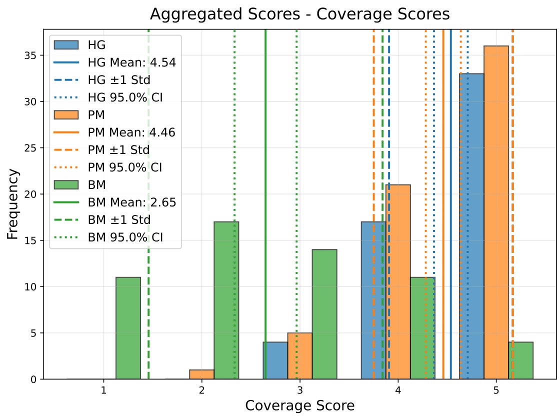
Inter-rater Reliability We calculated the inter-rater reliability for the relevance, generality, and coverage scores using multiple metrics, including Fleiss' Kappa, Interclass Correlation Coefficient



(a) Relevance scores



(b) Generality scores



(c) Coverage scores

Figure 4.14: Aggregated scores comparison by model

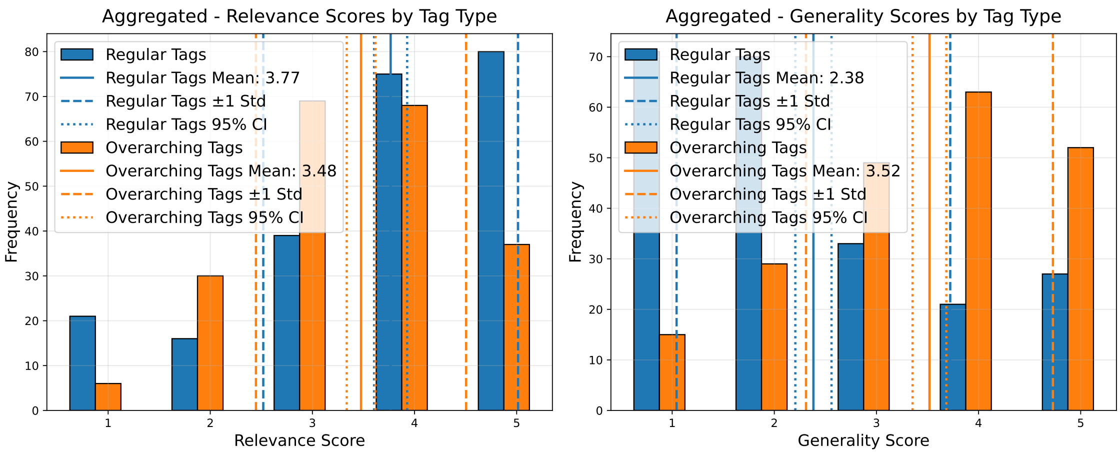
(ICC) and Krippendorff's Alpha. We do not use Cohen's Kappa, and use Fleiss' Kappa instead, as it is more suitable for multiple raters and multiple categories.

Table 4.9 shows the Fleiss' Kappa values for the different evaluation metrics for the baseline model, the proposed model, and the human-generated tags. We see that all values are small, indicating slight agreement at best. However, it is important to note that our data are Likert scale data, which are ordinal, and are not handled well by Fleiss' Kappa, as it treats categories as nominal.

Evaluation metric	Baseline model	Proposed model	Human-generated
Relevance	0.15	0.12	0.04
Generality	-0.02	0.18	0.10
Coverage	0.03	0.01	-0.02
Shared Coverage	0.03	0.06	-0.01

Table 4.9: Fleiss' Kappa values comparison for different evaluation metrics across models

We also used ICC to assess the consistency of ratings across our raters (Figures C.1 to C.3). Among the main forms of ICC, ICC(1) assumes random raters and absolute agreement, ICC(2) assumes random raters with consistency agreement, while ICC(3) is designed for fixed raters with consistency agreement. Furthermore, ICC(3,k) specifically measures the reliability of averaged



(a) Aggregated relevance scores (b) Aggregated generality scores

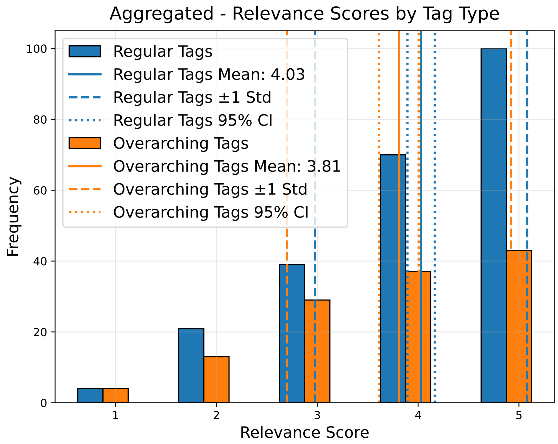
Figure 4.15: Aggregated relevance and generality scores by tag type (proposed model)

ratings rather than individual ratings. ICC(3,k) is most appropriate for our study design because we have a fixed set of participants rating the same texts across different conditions (baseline, proposed model, and human-generated tags), and we are interested in the reliability of averaged ratings rather than individual ratings. ICC(3,k) accounts for systematic differences in how individual participants use the rating scale while providing reliability estimates for the averaged scores, making it the most suitable choice for comparing the reliability of ratings across our three tag generation methods.

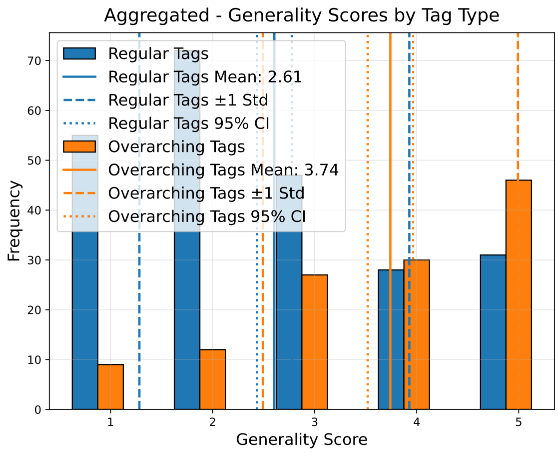
Looking at the ICC(3,k) values for relevance, where the F-statistic indicates the ratio of between-group to within-group variance (higher values showing stronger rater agreement), we see that the proposed model achieved an ICC of 0.95 ($F = 19.27$, $p = 7.72 \times 10^{-47}$, $\mathrm{CI}_{95\%}[0.91{,}0.98]$), the baseline model an ICC of 0.96 ($F = 25.92$, $p = 1.18 \times 10^{-57}$, $\mathrm{CI}_{95\%}[0.93{,}0.98]$), and the human-generated tags an ICC of 0.89 ($F = 8.70$, $p = 7.21 \times 10^{-20}$, $\mathrm{CI}_{95\%}[0.80{,}0.95]$). For generality, the proposed model performed best with an ICC of 0.97 ($F = 31.71$, $p = 4.86 \times 10^{-53}$, $\mathrm{CI}_{95\%}[0.94{,}0.99]$), followed by human-generated tags at 0.94 ($F = 16.76$, $p = 5.01 \times 10^{-32}$, $\mathrm{CI}_{95\%}[0.89{,}0.97]$), while the baseline model showed poor reliability with -0.01 ($F = 0.99$, $p = 0.416$, $\mathrm{CI}_{95\%}[-1.98{,}0.88]$) and a non-significant p-value indicating unreliable results. For coverage, the baseline model achieved the highest reliability at 0.95 ($F = 21.51$, $p = 7.14 \times 10^{-7}$, $\mathrm{CI}_{95\%}[0.81{,}1.00]$), followed by the proposed model at 0.82 ($F = 5.57$, $p = 0.007$, $\mathrm{CI}_{95\%}[0.27{,}1.00]$), while human-generated tags performed poorly at 0.08 ($F = 1.09$, $p = 0.348$, $\mathrm{CI}_{95\%}[-2.78{,}0.98]$) with non-significant results. For shared coverage, the proposed model showed the strongest reliability at 0.90 ($F = 10.25$, $p = 0.004$, $\mathrm{CI}_{95\%}[0.43{,}1.00]$), followed by human-generated tags at 0.76 ($F = 4.25$, $p = 0.055$, $\mathrm{CI}_{95\%}[-0.42{,}1.00]$) and the baseline model at 0.73 ($F = 3.65$, $p = 0.072$, $\mathrm{CI}_{95\%}[-0.64{,}1.00]$), though the latter two had borderline significant p-values suggesting less reliable results.

Similar to Fleiss' Kappa, ICC may not be ideal for ordinal data, as it assumes continuous data. We also provide results for Krippendorff's Alpha, which can handle both ordinal and nominal data. Table 4.10 shows the Krippendorff's Alpha values for the different evaluation metrics for the baseline model, the proposed model, and the human-generated tags. We see that the baseline model performs best in shared coverage ($\alpha = 0.4818$), while showing moderate reliability for generality ($\alpha = 0.2863$) and coverage ($\alpha = 0.2600$), but lower reliability for relevance ($\alpha = 0.1687$). The proposed model shows strongest reliability in coverage ($\alpha = 0.5647$), moderate reliability in shared coverage ($\alpha = 0.2497$) and relevance ($\alpha = 0.2263$), but lower reliability in generality ($\alpha = 0.1671$). The human-generated tags show the highest reliability for relevance ($\alpha = 0.4692$) and coverage ($\alpha = 0.4503$), but lower reliability for generality ($\alpha = 0.2044$) and shared coverage ($\alpha = 0.1898$).

This analysis reveals a complex and sometimes contradictory pattern of interrater reliability.



(a) Aggregated relevance scores



(b) Aggregated generality scores

Figure 4.16: Aggregated relevance and generality scores by tag type (human-generated)

The metrics often disagree substantially in their assessment of reliability. For instance, while the proposed model shows remarkably high ICC values for generality (0.97) and shared coverage (0.90), the corresponding Krippendorff's Alpha values are notably low (0.1671 and 0.2497 respectively). Similarly, the baseline model's high ICC scores for relevance (0.96) and coverage (0.95) contrast sharply with its low Krippendorff's Alpha (0.1687). Human-generated tags show their own contradictions - strong ICC for relevance (0.89) paired with moderate Krippendorff's Alpha (0.4692), and very low ICC for coverage (0.08) yet moderate Krippendorff's Alpha (0.4503). These discrepancies between metrics make it particularly challenging to draw definitive conclusions about relative reliability across the different approaches.

Metric	**Baseline model**	**Proposed model**	**Human-generated**
Relevance	0.1687	0.2263	0.4692
Generality	0.2863	0.1671	0.2044
Coverage	0.2600	0.5647	0.4503
Shared Coverage	0.4818	0.2497	0.1898

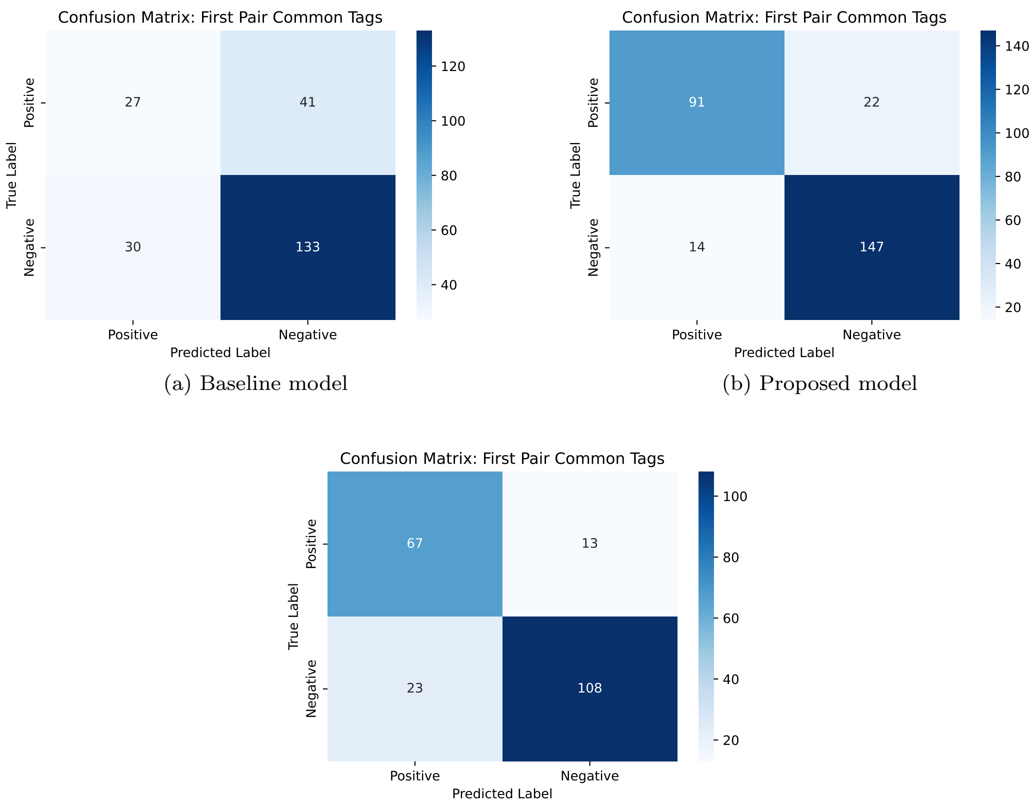
Table 4.10: Krippendorff's Alpha values for different rating types across models

Stage 2 — Task 1: Common Tags Identification

The confusion matrices in Figure 4.17 show the common tags confusion matrix comparison for the first pair of texts. The confusion matrices in Figure 4.18 show the common tags confusion matrix comparison for the second pair of texts.

The top-left quadrant of the confusion matrix represents the true positives, i.e., the number of tags that the model predicted as common tags that human evaluators also identified as common tags. The bottom-right quadrant represents the true negatives, i.e., the number of tags that the model predicted as not common tags that human evaluators also identified as not common tags. The top-right quadrant represents the false negatives, i.e., the number of tags that the model predicted as common tags that human evaluators identified as not common tags. The bottom-left quadrant represents the false positives, i.e., the number of tags that the model predicted as not common tags that human evaluators identified as common tags.

To interpret the confusion matrices, in Table 4.11, we compare accuracy, precision, recall, specificity, and F1-Score between the first and second pairs for the baseline model, the proposed



(a) Baseline model (b) Proposed model

(c) Human-generated

Figure 4.17: Common tags confusion matrix comparison for the first pair

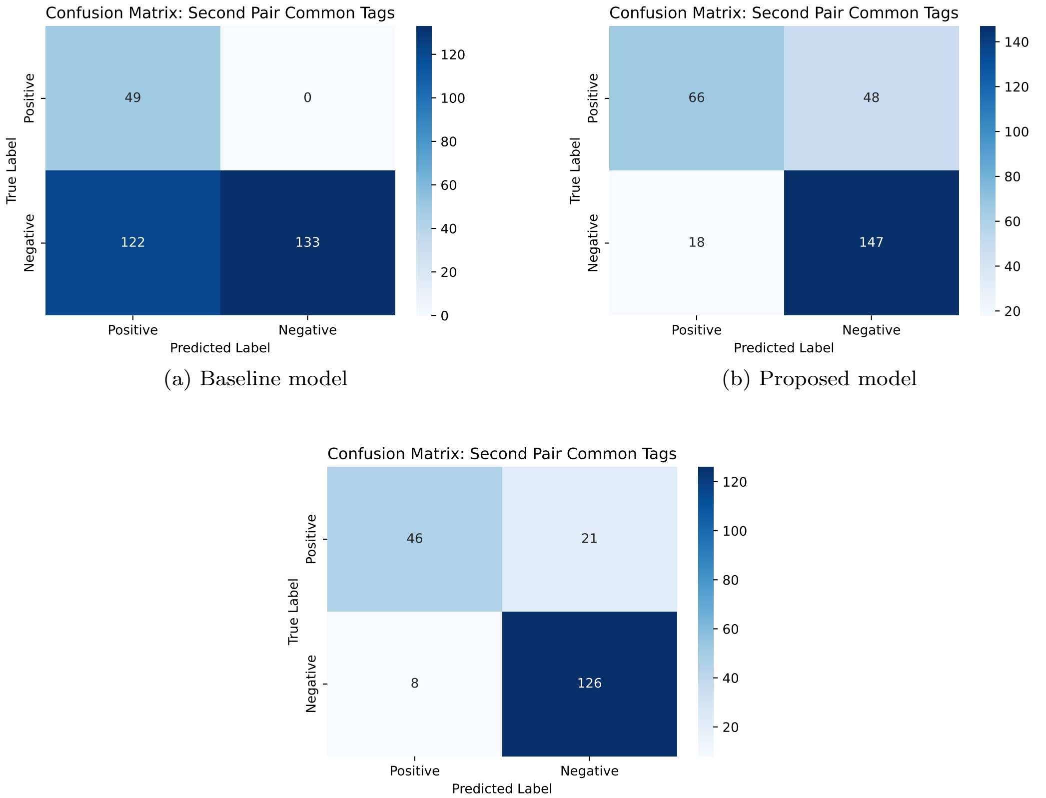
model, and human-generated tags, respectively. The results show varying performance patterns across different metrics. For the first pair, the proposed model achieved the highest accuracy (0.87) and F1-Score (0.83), outperforming both the baseline model (0.69 accuracy, 0.43 F1-Score) and slightly exceeding human-generated tags (0.83 accuracy, 0.79 F1-Score). However, for the second pair, human-generated tags demonstrated superior performance with 0.86 accuracy compared to the proposed model's 0.76 and the baseline's 0.60.

The baseline model showed a peculiar pattern in the second pair, with perfect precision (1.00) due to having no false positives, but poor recall (0.29) due to a high number of false negatives. This is because the two surveys were in the same topic cluster, and since the zeroshot classifier was not applied, all the tags from the cluster were provided. These results showcase how not using the LLM fine-tuning step and the zeroshot classifier can lead to poor performance.

The proposed model maintained more balanced metrics across both pairs, though with some decline in performance for the second pair, particularly in precision (dropping from 0.81 to 0.58). Human-generated tags showed the most consistent performance across both pairs, with all metrics maintaining relatively stable values between 0.69 and 0.89.

These results suggest that while the proposed model shows promising performance, particularly for the first pair, its effectiveness varies more between different text pairs compared to human evaluators. The baseline model's extreme variations in metrics indicate less reliable performance overall.

Statistical significance We additionally perform Kruskal-Wallis tests to determine if the differences in the evaluation metrics between the baseline model, the proposed model, and human-



(a) Baseline model

(b) Proposed model

(c) Human-generated

Figure 4.18: Common tags confusion matrix comparison

generated tags are statistically significant. The Kruskal-Wallis test is a non-parametric test that compares the medians of two or more groups. We use this test because our data are ordinal and do not meet the assumptions of parametric tests, which require a normal distribution and homogeneity of variances.

Table 4.12 shows the results of the Kruskal-Wallis H test for the different evaluation metrics. We see that the differences in relevance, coverage, and shared coverage are statistically significant, while the differences in generality are not statistically significant. The H-statistic indicates the magnitude of difference between the groups, with higher values suggesting greater differences. The highest H-statistic is observed for relevance (H = 239.29), indicating substantial differences in how participants rated tag relevance across the three methods. Coverage (H = 65.54) and shared coverage (H = 47.82) also show considerable differences, while generality shows minimal differences (H = 4.56).

The effect sizes, measured by η^2, provide additional insight into the practical significance of these differences. According to common interpretation guidelines, η^2 values of 0.01, 0.06, and 0.14 represent small, medium, and large effects, respectively. The results show large effect sizes for coverage ($\eta^2 = 0.562$), shared coverage ($\eta^2 = 0.405$), and relevance ($\eta^2 = 0.300$). In contrast, generality shows a medium effect size ($\eta^2 = 0.005$). The p-values (< 0.001) for relevance, coverage, and shared coverage indicate that these differences are highly unlikely to have occurred by chance, while the non-significant p-value for generality ($p = 0.102$) suggests that any observed differences in generality ratings could be due to random variation.

After finding these statistically significant differences, we perform post-hoc pairwise comparisons using Dunn's test with Bonferroni correction to identify which groups differ from each other.

Metric	Baseline model		Proposed model		Human-generated	
	First	Second	First	Second	First	Second
Accuracy	0.69	0.60	0.87	0.76	0.83	0.86
Precision	0.40	1.00	0.81	0.58	0.84	0.69
Recall	0.47	0.29	0.87	0.79	0.74	0.85
Specificity	0.76	1.00	0.87	0.75	0.89	0.86
F1-Score	0.43	0.45	0.83	0.67	0.79	0.76

Table 4.11: Comparison of evaluation metrics between first and second pairs across all models

Metric	H-statistic	p-value	η^2
Relevance	239.29	< 0.001	0.300
Generality	4.56	0.102	0.005
Coverage	65.54	< 0.001	0.562
Shared Coverage	47.82	< 0.001	0.405

Table 4.12: Kruskal-Wallis H test results for different evaluation metrics

Figure 4.19 shows the p-values for the post-hoc tests for relevance, generality, coverage, and shared coverage. The results indicate that the differences between the baseline model and the proposed model, as well as between the baseline model and human-generated tags, are statistically significant for all metrics, besides generality. The differences between the proposed model and human-generated tags are only statistically significant for relevance ($p = 0.011$). For generality, coverage, and shared coverage, there were no statistically significant differences between our proposed model and human-generated tags ($p = 0.158$, $p = 1.000$, and $p = 0.511$ respectively). This suggests that while our model still lags behind human performance in terms of tag relevance, it achieves comparable performance to humans in generating tags with appropriate generality levels and comprehensive coverage of the text content. Given our sample size of 21 participants, the lack of statistically significant differences between the proposed model and human-generated tags for generality, coverage, and shared coverage could be due to limited statistical power. A larger number of participants would increase our ability to detect smaller differences between these approaches, if they exist. Future work with more raters could provide stronger statistical evidence to either confirm the apparent equivalence between our model and human performance or reveal differences that our current sample size was unable to detect.

	Baseline	Human	Model
Baseline	1.000	< 0.001	< 0.001
Human	< 0.001	1.000	0.011
Model	< 0.001	0.011	1.000

(a) Relevance

	Baseline	Human	Model
Baseline	1.000	1.000	0.417
Human	1.000	1.000	0.158
Model	0.417	0.158	1.000

(b) Generality

	Baseline	Human	Model
Baseline	1.000	< 0.001	< 0.001
Human	< 0.001	1.000	1.000
Model	< 0.001	1.000	1.000

(c) Coverage

	Baseline	Human	Model
Baseline	1.000	< 0.001	< 0.001
Human	< 0.001	1.000	0.511
Model	< 0.001	0.511	1.000

(d) Shared Coverage

Figure 4.19: Dunn's post-hoc test p-values for different metrics

We include Cliff's Delta effect size to provide additional context on the magnitude of differences between groups. Cliff's Delta is a non-parametric effect size measure that quantifies the difference between two groups by calculating the probability that a randomly selected observation from one group will be greater than a randomly selected observation from the other group. The effect sizes are interpreted as small (0.147), medium (0.33), and large (0.474) based on common guidelines. Table 4.13 shows the effect sizes for the different metrics and comparisons. We see that baseline scores differ substantially from both human and model scores for most metrics. For relevance ratings, there are large negative effects when comparing baseline to both human ($\delta = -0.703$) and model ($\delta = -0.599$), indicating that baseline tags were rated consistently lower in relevance. The small positive effect between human and model ratings ($\delta = 0.163$) suggests that model-generated tags achieve relevance levels comparable to human-generated ones.

Generality shows a different pattern, with negligible effects across all comparisons (δ ranging from 0.014 to 0.112), indicating that all three methods produce tags with similar levels of generality. However, these results should be interpreted with caution, as the Not Applicable (NA) tags were excluded from the generality analysis.

Coverage metrics reveal the most pronounced differences. Both overall coverage and shared coverage show large negative effects when comparing baseline to human ($\delta = -0.885$ and -0.789 respectively) and model ($\delta = -0.888$ and -0.681 respectively). This strongly suggests that both human and model-generated tag sets provide substantially better coverage than baseline tags. The negligible to small effects between human and model coverage scores ($\delta = -0.053$ for coverage, $\delta = 0.200$ for shared coverage) indicate that the model achieves coverage levels similar to human performance.

Metric	**Comparison**	**Delta**	**Effect**
Relevance	Baseline model vs Human-generated	-0.703	Large
	Baseline model vs Proposed model	-0.599	Large
	Human-generated vs Proposed model	0.163	Small
Generality	Baseline model vs Human-generated	0.014	Negligible
	Baseline model vs Proposed model	0.112	Negligible
	Human-generated vs Proposed model	0.101	Negligible
Coverage	Baseline model vs Human-generated	-0.885	Large
	Baseline model vs Proposed model	-0.888	Large
	Human-generated vs Proposed model	-0.053	Negligible
Shared Coverage	Baseline model vs Human-generated	-0.789	Large
	Baseline model vs Proposed model	-0.681	Large
	Human-generated vs Proposed model	0.200	Small

Table 4.13: Cliff's Delta effect sizes for different metrics and comparisons

To provide a comprehensive analysis of effect sizes, we calculated Cohen's d alongside Cliff's Delta, while acknowledging its limitations for our data characteristics. Given our ordinal Likert scale data, non-normal distribution, and small sample size (n=21), Cohen's d results should be interpreted with caution. Nevertheless, as shown in Table C.3, the calculations reveal strong effects that align with our Cliff's Delta findings. For relevance, large negative effects were found comparing baseline to both human ($d = -1.589$) and model ($d = -1.254$), with a small to medium effect between human and model ($d = 0.293$). Generality showed negligible to small effects across comparisons (d ranging from 0.027 to 0.202). Coverage metrics demonstrated the largest effects, with very large negative effects when comparing baseline to both human ($d = -2.454$) and model ($d = -2.539$) for overall coverage, and similarly large effects for shared coverage (baseline vs human: $d = -1.882$; baseline vs model: $d = -1.485$). The human vs model comparisons showed negligible effects for coverage ($d = -0.078$) and small to medium effects for shared coverage ($d = 0.391$). While these results support our main findings, we primarily rely on Cliff's Delta as our effect size measure due to its better alignment with our data characteristics and non-parametric analysis approach.

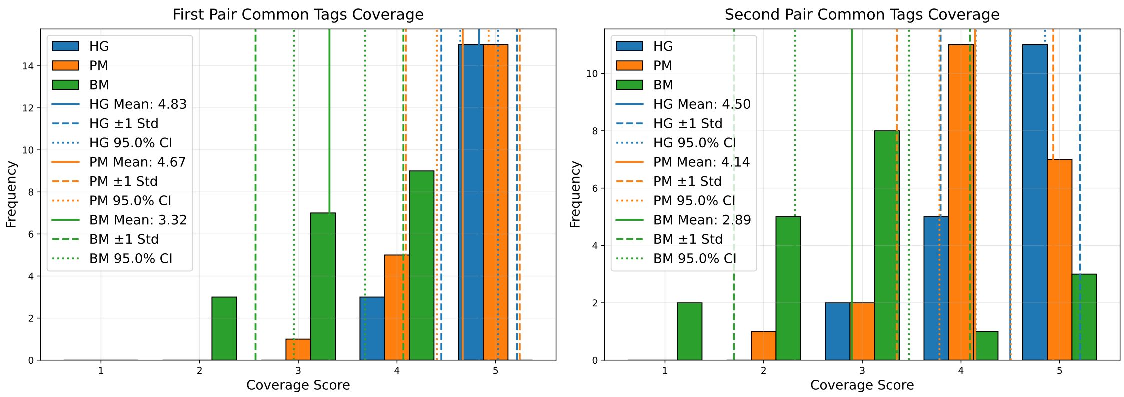
Stage 2 — Task 2: Common Tags Quality Assessment

Figures 4.20a and 4.20b show the common tags coverage comparison for the first and second pairs, respectively.

Similar to the tag quality assessment, we see that the proposed model outperformed the baseline model, but still lagged behind human-generated tags, for both pairs of texts.

For the first pair, the proposed model achieved a mean coverage of 4.67, while the baseline model achieved a mean coverage of 3.32. The human-generated tags achieved a mean coverage of 4.83. For the second pair, the proposed model achieved a mean coverage of 4.14, while the baseline model achieved a mean coverage of 2.89. The human-generated tags achieved a mean coverage of 4.50.

This indicates that the proposed model was able to identify common tags more accurately than the baseline model, but still not quite as accurately as the human-generated tags.



(a) Common tags coverage by model for the first pair (b) Common tags coverage by model for the second pair

Figure 4.20: Common tags coverage comparison by model for both pairs

4.4.4 Large-scale automated evaluation

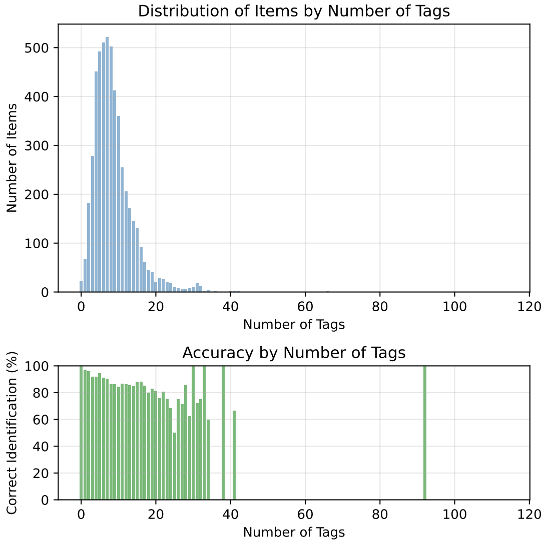
We follow the methodology outlined in section 3.4.2 to evaluate the proposed model, running the evaluation on all ∼5200 datasets. As an evaluation model, we pick *GPT-4o-mini*, as it is a smaller and relatively less expensive LLM compared to larger models, yet offers good performance compared to models of similar size [11].

Automated Intruder Detection

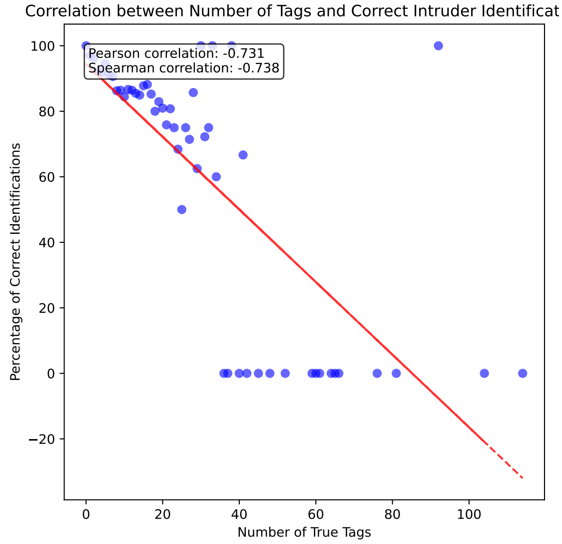
The automated intruder detection experiment revealed that the LLM successfully identified 88.3% of intruders, with only 11.7% of cases resulting in incorrect detection. This high accuracy in identifying intruder tags among true tags suggests that the tags generated by the proposed model form cohesive, semantically related groups. The ability to distinguish contextually unrelated tags from related ones provides evidence for the semantic consistency of the model's tag generation process. In contrast, for the baseline model, only 47.5% of intruders were correctly identified, indicating a much lower level of semantic coherence in the tag sets generated by the baseline model.

We observe that the intruder detection accuracy decreases as the number of tags increases, as shown in Figure 4.21a. This is expected, as the evaluation model has to consider more tags, making it more challenging to detect intruders. Choosing a larger model may help improve intruder detection accuracy for a larger number of tags.

This finding is corroborated by the correlation analysis in Figure 4.21b, which shows a negative correlation between the number of tags and intruder detection accuracy.



(a) Intruder detection accuracy by number of tags



(b) Correlation between number of tags and intruder detection accuracy

Figure 4.21: Analysis of Intruder Detection Accuracy and Number of Tags

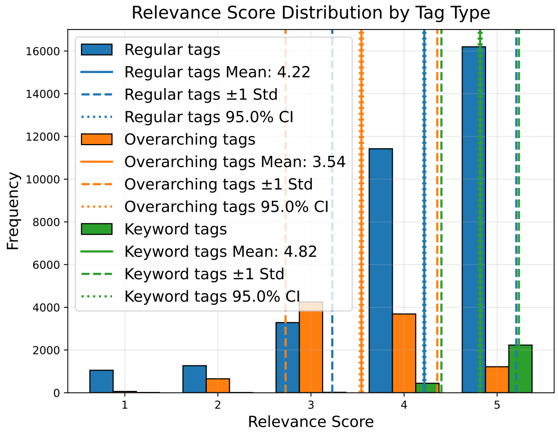
Automated Tag Quality Assessment

We investigate the correlation between relevance and generality and see that there is a positive correlation between them, with a Pearson correlation coefficient of 0.108 and a Spearman correlation coefficient of 0.012, both of which have a p-value of less than 0.01. Given the Likert scale nature of the data, we consider the Spearman correlation coefficient to be more reliable, indicating that there is practically no correlation between relevance and generality according to the LLM.

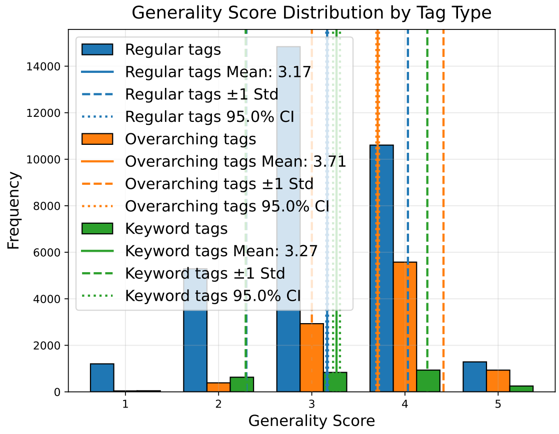
This is a surprising finding, as the human evaluation showed a negative correlation between relevance and generality, namely, -0.59. This discrepancy may be attributed to the differences in how humans and LLMs perceive relevance and generality, or to the limitations of the LLM in capturing the nuances of human judgment. However, upon closer inspection of the data, we find that for the same texts used in the human study, the LLM also shows a negative correlation between relevance and generality, with a Spearman correlation coefficient of -0.407 and a p-value of 0.075. This suggests that the discrepancy in the correlation between relevance and generality is likely due to the differences in the three datasets used for the human evaluation rather than a fundamental difference in how humans and LLMs perceive relevance and generality. For the tags generated by our proposed model, this finding may indicate that there is a lower or nonexistent correlation between relevance and generality, meaning that both regular and overarching tags are found as relevant. This means that we should not lean on choosing one over the other, but rather consider including both types of tags in the final tag set.

When we investigate the scores for the generated tags by the proposed model, we find a mean relevance score of 4.11 and a mean generality score of 3.29 with a standard deviation of 0.87. The mean coverage score is 3.72 (Figure 4.22).

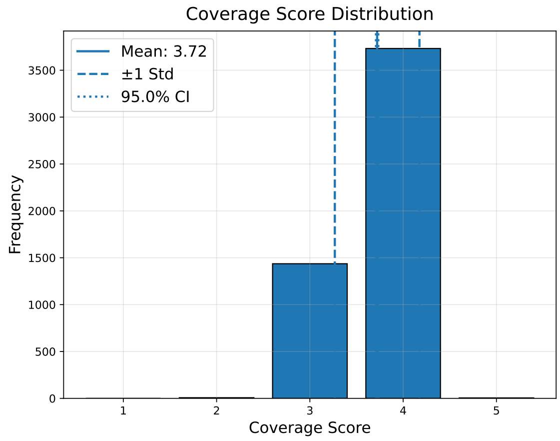
When comparing with the tags generated by the baseline model (Figure 4.23), for which we have a mean of 3.26 for relevance, 1.13 SD for generality and 3.11 mean for coverage, we see that the proposed model outperforms the baseline model in relevance and coverage, according to the LLM. Surprisingly, the generality score is lower for the proposed model compared to the baseline model, which seems contrary to the human evaluation results. However, in the evaluation of the human study results, we performed a statistical analysis to determine the significance of the differences in generality between the baseline model and the proposed model, and found no significant difference.



(a) Distribution of relevance scores by tag type



(b) Distribution of generality scores by tag type



(c) Distribution of coverage scores

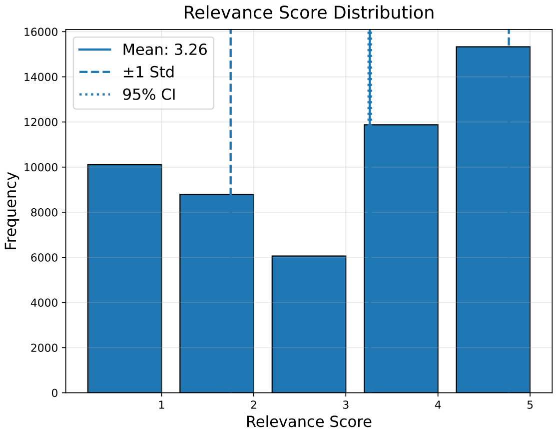
Figure 4.22: Distribution of relevance, generality, and coverage scores

Therefore, this findidng is not necessarily contradictory to the human evaluation results.

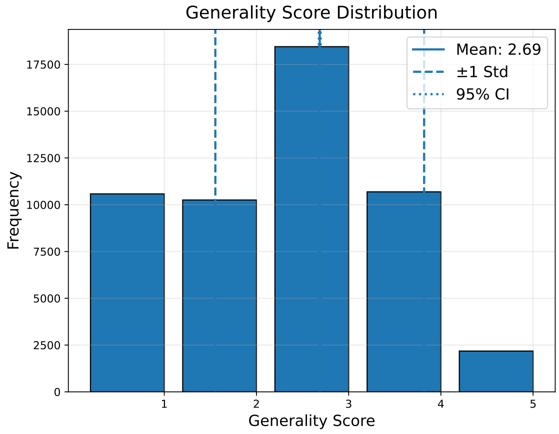
A comparison between the LLM's evaluation of the proposed model and the human evaluation of the proposed model reveals some interesting discrepancies. While the LLM rated the model's relevance at 4.11, humans rated it slightly lower at 3.63. For generality, the difference is more pronounced: the LLM assessed a mean score of 3.29 with a standard deviation of 0.87, whereas human ratings reflected in a mean of 2.93 and a standard deviation of 1.40. In terms of coverage the LLM and humans rated the model with scores of 3.72 and 4.46, respectively. These findings suggest that the LLM may be more lenient in its evaluation of generality compared to humans, while showing similar performance in relevance and coverage assessment. It could also be the case that the datasets chosen for the human evaluation were different from the datasets used for the LLM evaluation, leading to discrepancies in the results.

Normality testing The Q-Q plots in Figure B.6 show the distribution of relevance, generality, and coverage scores, respectively. We see that the scores are not normally distributed, which is expected given the ordinal nature of the Likert scale data, and is similar to the human evaluation results.

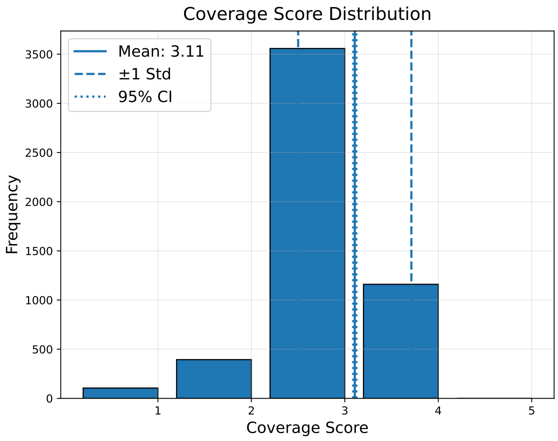
Table 4.14 shows the results of the normality tests for the relevance, generality, and coverage scores. We see that the Shapiro-Wilk test and Anderson-Darling test both indicate that the scores



(a) Distribution of relevance scores (baseline)



(b) Distribution of generality scores (baseline)



(c) Distribution of coverage scores (baseline)

Figure 4.23: Distribution of relevance, generality, and coverage scores (baseline)

are not normally distributed.

Table 4.14: Normality tests across metrics

Test	**Relevance**	**Generality**	**Coverage**
Shapiro-Wilk Statistic	0.8799	0.8799	0.5734
Shapiro-Wilk p-value	$<$0.0001	$<$0.0001	$<$0.0001
Anderson-Darling Statistic	1905.4133	2641.2348	1154.0435

Table 4.15 shows the distribution metrics for the relevance, generality, and coverage scores. Looking at the mean, median, mode, standard deviation, skewness, skewness z-score, and kurtosis, we see that the scores are generally well-distributed, with a slight to moderate negative skewness and kurtosis. The negative skewness shows that the distribution is left-skewed, and the kurtosis values indicate that the distribution is platykurtic (less peaked than a normal distribution). These findings corroborate the Q-Q plots and normality test results.

Table 4.15: Distribution metrics comparison

Metric	Relevance	Generality	Coverage
Mean	3.1651	3.2869	3.7210
Median	3.0	3.0	4.0
Mode	3	3	4
Standard Deviation	0.8667	0.8705	0.4536
Skewness	-0.3014	-0.3544	-0.9979
Skewness z-score	-0.33	-10.41	-1.08
Kurtosis	-0.0148	-0.0169	-0.7863

4.5 Evaluation of alternative configurations

In addition to the primary pipeline evaluation, we explored a more cost-effective configuration designed to reduce computational expenses, particularly regarding LLM usage, following the alternative configuration outlined in section 3.2.2. This alternative pipeline deviates from the original approach in three aspects:

1. **Omission of Description Rewriting**: We eliminated the step of rewriting augmented descriptions into a more human-readable format (Step 3 in the original pipeline).

2. **Simplified Tag Generation and Classification**: We combined the tag generation and classification steps into a single step (step 9), utilizing GPT-4o-mini with a maximum output of 256 tokens per description, and removing the zeroshot classifier entirely (step 10).

3. **Choice of LLM**: We opted for GPT-4o-mini for the combined tag generation and classification step, as it offers a good balance between performance and cost compared to larger models [11]. GPT-4o-mini was priced at $0.15 per 1M tokens of input and $0.60 per 1M tokens of output.

This configuration significantly reduced the overall processing time. Embedding generation (step 4) took approximately 4 hours using the *Salesforce/SFR-Embedding-2_R* model [3]. Omission of description rewriting reduced processing time by 3 hours and expenses by approximately $8.00 compared to the proposed pipeline. Training the base BERTopic model (steps 5-8) required about 5 minutes. Substantial resource saving was also achieved in the fine-tuning step (step 9), which took approximately 2 hours and cost only $0.80 using GPT-4o-mini, compared to using the Llama-3-70b model in the original pipeline, which took similar time but cost approximately $4.80.

While this configuration offers significant cost and time advantages, our initial observations suggest a potential trade-off in tag quality compared to the original pipeline. The absence of description rewriting led to a degradation in the quality of the c-TF-IDF representations. Specifically, the c-TF-IDF output for this configuration contained a higher proportion of nonsensical or irrelevant terms (e.g., *q59fi3*, *timmenziesus*, *145160*) compared to the original pipeline, which benefited from cleaner, human-readable descriptions. This difference in c-TF-IDF quality likely impacted the subsequent tag generation, resulting in some less relevant or nonsensical tags (e.g., *Rows*, *Columns*, *Seed*, *Intra-Class*).

The impact on clustering is more difficult to quantify objectively. However, subjective inspection revealed instances where unrelated descriptions were grouped into the same cluster. Additionally, we observed a tendency towards fragmentation, with similar descriptions being split into multiple smaller clusters instead of being grouped into a single, larger cluster. For example, instead of one large cluster for all QSAR-related datasets, there were many smaller clusters containing different subsets of QSAR datasets.

Using a cheaper LLM (GPT-4o-mini) for the combined tag generation and classification step introduced further challenges. The model tended to extract more specific tags (e.g., *Cropped Images* instead of *Images*) at the expense of more general, common tags. While the overarching tags remained largely unaffected, the model occasionally produced incorrectly formatted JSON outputs, requiring manual correction in roughly 10 out of 900 cases. Furthermore, the model sometimes failed to assign generated tags during the classification phase or introduced new tags not present in the initial generation step. The confidence scores produced by the LLM were also less reliable than those from the dedicated zeroshot classifier in the original pipeline, as the LLM is not specifically trained for this task. Finally, this configuration does not provide tags from the top n clusters for each description, potentially missing out on relevant tags.

Despite these limitations, the tags produced by this cost-effective configuration are still generally of good quality. This configuration offers a viable alternative when computational resources or budget constraints are a primary concern. Further evaluation using a large-scale automated approach, similar to the one performed for the original pipeline, would provide a more comprehensive assessment of the trade-offs between cost and quality.

4.6 Chapter conclusion

In this chapter, we explained how we followed the methodology outlined in chapter 3 to evaluate the proposed model. We addressed each research question from **RQ1** through **RQ4**.

We answered **RQ1** by performing an exploratory data analysis and data augmentation. For the exploratory data analysis, we analyzed the 5200 dataset descriptions, investigating traits such as length, readability, duplicate content, their tags, features, and versions. Our findings revealed that descriptions are generally short, with a median length comparable to two tweets, and exhibit a wide range of readability scores, ranging from 0 to 100, with many falling in the 0-60 range on the Flesch Reading Ease scale, indicating varying complexity. We observed that a significant portion of datasets already contain associated tags, with most datasets having between 0 and 5 tags, and many descriptions include URLs to external resources. We augmented the descriptions with metadata (dataset name, existing tags, column names) and text scraped from external URLs. We also removed 3̃00 duplicate datasets with high cosine similarity. Additionally, we applied Named Entity Recognition (NER) and Part-of-Speech (POS) tagging, which revealed challenges posed by domain-specific terms and author names for generating relevant tags.

RQ2 was answered by following the tag generation methodology in section 3.2.1. We detailed the implementation of our tag generation pipeline, extending the Base BERTopic model with an LLM (Llama-3-70b) for prompt-based tag generation and a zeroshot classifier (MoritzLaurer/deberta-v3-large-zeroshot-v2.0) for tag filtering. We provided a step-by-step description of the pipeline, including the rationale for selecting specific submodels. We highlighted the advantages of this approach over naive LLM-based methods, particularly in terms of contextual consistency, multi-level tag generation, and computational efficiency. The main limitation of the proposed tag generation pipeline is the computational cost of the Llama-3-70b model. We also explored a more cost-effective pipeline configuration, omitting the description rewriting step, using a cheaper LLM (GPT-4o-mini), and combining tag generation and classification into a single step. While this configuration reduced computational costs and time, it led to a decrease in tag quality, as observed through the degradation of c-TF-IDF representations and some clustering issues.

For **RQ3**, we evaluated the Base BERTopic model using the automated evaluation metrics we discussed in section 3.3. We compared the hyperparameter-optimized BERTopic model against several baselines, including LDA, NMF, CTM, Top2Vec, and different BERTopic configurations. Our results showed that the optimized BERTopic model achieved the highest combined NPMI and diversity score of 0.779, outperforming all baseline models on this combined metric. NMF achieved the highest NPMI, but had very low diversity. The optimized BERTopic model had the highest diversity (0.808). We performed hyperparameter tuning using Bayesian optimization with a custom weighted metric combining NPMI and diversity. The optimized hyperparameters included a *min_topic_size* of 2, *umap_n_neighbors* of 3, *umap_n_components* of 5, *hdbscan_min_cluster_size* of 3, along with specific settings for other parameters detailed in Table 4.1. Statistical analysis using ANOVA, Kruskal-Wallis, and post-hoc tests confirmed significant differences between models for NPMI, diversity, with large effect sizes, highlighting the strong performance of our optimized BERTopic model and the BERTopic family as a whole.

Finally, we answered **RQ4** by evaluating the proposed model using human evaluation and large-scale automated evaluation. For the human evaluation, we picked a suitable baseline model (Base BERTopic), and created a human-generated set of tags. We conducted a two-stage evaluation, where we first asked participants to perform intruder detection, and evaluated the relevance, generality, coverage, and shared coverage of the tags generated by the proposed model, the baseline model and the human-generated tags. Our model achieved a mean relevance score of 3.63, outperforming the baseline (2.39) and approaching the performance of human-generated tags (3.95). For generality, our model showed a better balance between specific and general tags (SD = 1.40) compared to the baseline (SD = 1.33) and was close to human-generated tags (SD = 1.41). In terms of coverage, our model (4.46) outperformed the baseline (2.65) and was close to human-generated tags (4.54). The intruder detection task demonstrated our model's ability to generate cohesive tag sets, with a 95.2% detection rate, compared to 42.1% for the baseline and 100% for human-generated tags. For inter-rater agreement, we calculated Fleiss' Kappa, Interclass Correlation Coefficient (ICC), and Krippendorff's Alpha. While Fleiss' Kappa indicated only slight agreement at best (all values below 0.18), ICC and Krippendorff's Alpha showed more nuanced results, with some metrics indicating moderate to strong agreement (e.g., ICC(3,k) values of 0.95 for relevance for the proposed model, and 0.97 for generality, and Krippendorff's Alpha of 0.5647 for coverage). However, there was also disagreement between the metrics, suggesting complexity in assessing inter-rater reliability for this task.

The common tag identification task revealed that our model achieved balanced precision and recall across two text pairs, with an accuracy of 0.87 and 0.76 respectively, which was higher than the baseline (0.69 and 0.60), slightly higher than the human-generated tags for the first pair (0.83), but slightly lower than human-generated tags for the second pair (0.86). Statistical tests, including Kruskal-Wallis and Dunn's post-hoc test, showed statistically significant differences between our model and the baseline for all metrics. While our model's tags were rated significantly lower in relevance compared to human-generated tags, there were no significant differences in generality, coverage, and shared coverage. Effect size analysis using Cliff's Delta confirmed large differences between the baseline and both our model and human-generated tags for relevance, coverage, and shared coverage. However, the effect sizes between our model and human-generated tags were small to negligible, indicating that our model's performance was comparable to human-generated tags.

For the large-scale automated evaluation, we evaluated the proposed model using an LLM, *GPT-4-mini*. We found that the LLM managed to detect the majority of intruders in the dataset (88.3%), indicating that the tags generated by the proposed model are cohesive. Tag quality assessment by the LLM yielded a mean relevance score of 4.11, a mean generality score of 3.29 (SD = 0.87), and a mean coverage score of 3.72. The LLM found that keyword tags were the most relevant, and it found them to be slightly more general than regular tags. The automated evaluation also found a weakly positive (near-neutral) correlation between relevance and generality, which differed from the negative correlation found in the human evaluation. The discrepancy in correlation may be attributed to differences in how humans and LLMs perceive these qualities, or to the different dataset samples used in the two evaluations.

Our results demonstrate that the proposed model generates cohesive, relevant and diverse tags for OpenML dataset descriptions with a good coverage, outperforming the baseline model and approaching human-level performance in several aspects.

Chapter 5

Recommendations

In this chapter, we present several key recommendations for improving and extending the tag generation system, ranging from immediate practical enhancements to longer-term research directions.

5.1 Pipeline component updates

The modular nature of our pipeline presents ongoing opportunities for improvement through the integration of newer, better-performing submodels. The rapid advancement in language models and embedding techniques means that regularly evaluating and incorporating new submodels could yield improvements. Particularly promising areas include embedding models achieving high scores on the MTEB benchmark [4], newer LLMs for fine-tuning, and improved zeroshot text classifiers. Our automated evaluation pipeline and large-scale automated evaluation methodology allow for quick testing of these components, making continuous improvement practical and efficient.

5.2 Improved zeroshot classification

While the current zeroshot text classifier performs adequately, there is potential for improvement in this area. Training a custom zeroshot model with a better base architecture could be an entire Master's project in itself, given the complexity and resource requirements involved. The zeroshot text classifier in this thesis is based on DeBERTa [93], a relatively small and old model.

A more practical short-term solution would be to use a pre-trained LLM for zeroshot text classification. Dedicated zeroshot models are usually smaller and trained less frequently because they are highly specialized for one task. As a result, there is less demand for them compared to general-purpose LLMs, which tend to perform better across a wider range of tasks, including text classification (for our experiments demonstrating this, refer to the *preliminary_model.ipynb* notebook in the GitHub repository [90]). Even though the LLM would not provide confidence scores, it would still likely outperform the current zeroshot model. For long-term development, following Laurer et al. [61]'s work on training zeroshot text classifiers could provide a valuable starting point for training a custom model.

5.3 External dataset validation

To validate the broader applicability of our approach, we recommend conducting word intruder tests on popular datasets beyond OpenML. This evaluation would help assess the model's generalizability and identify potential areas for improvement when handling different types of data. Furthermore, comparing results against current state-of-the-art topic models would provide valuable insights into the strengths and limitations of our approach. To start this process, we recommend finding the relevant literature in which word intruder tests are used to evaluate topic models and adapting these methods to our context [62, 65, 67, 69, 70, 84, 85].

5.4 OpenML platform integration

For the OpenML platform, we recommend keeping track of which tags are generated by the model and which are human-generated. This would allow for easy updates to the tags in the future when the entire model is rerun, without affecting the human-generated tags.

5.5 Online topic modeling

We recommend exploring iterative (online) dimensionality reduction and, especially, clustering algorithms to address the limitations of the current static approaches. To take the example of clustering, hierarchical clustering algorithms such as HDBSCAN have been shown to be effective in many applications [6, 7, 8], but they are static. Static clustering has limitations in that they are run once for a fixed set of data and do not adapt to size changes in the dataset.

Recalculating the clustering from scratch for each new dataset is not computationally expensive for our dataset of approximately 5000 descriptions, but could become an issue as the dataset grows. For instance, HDBSCAN's time complexity is $O(n^2)$ for computing the distance matrix, which could become a bottleneck for larger datasets. Iterative clustering algorithms, on the other hand, can adapt to changes in the dataset without having to recalculate the clusters from scratch.

Researching and implementing iterative dimensionality reduction algorithms such as Incremental PCA [94, 95, 96], and iterative clustering algorithms [97] such as Mini Batch K-means [98, 99] or BIRCH [100] could provide a more scalable and efficient solution for the long-term.

Chapter 6

Conclusions

In this master thesis, we developed a novel approach to automatically generate tags for OpenML dataset descriptions using topic modeling techniques. Our research addressed four main research questions, making several key contributions to both the OpenML platform and the field of topic modeling.

Addressing **RQ1** (What are the specifications of the OpenML dataset descriptions, and how do they impact model performance?), our exploratory data analysis of 5200 OpenML dataset descriptions revealed that they are generally short, with a median length comparable to two tweets, and exhibit a wide range of readability scores, ranging from 0 to 100 on the Flesch Reading Ease scale, with many falling in the 0-60 range, indicating relatively high and varying complexity. We found that many descriptions contain domain-specific technical terms, and that a significant portion of datasets already have associated tags, with most having between 0 and 5 tags. We augmented the descriptions with metadata (dataset name, existing tags, column names up to 50), and text scraped from external URLs. We also removed approximately 300 duplicate datasets identified through high cosine similarity. Named Entity Recognition (NER) and Part-of-Speech (POS) tagging revealed challenges in tag generation due to domain-specific terms and the prevalence of author names.

For **RQ2** (What are the different approaches to topic modeling that can be applied to the OpenML dataset descriptions, and what are the advantages and disadvantages involved in their use?), we developed a modular, ten-step pipeline (section 3.2.1) that extends the Base BERTopic model. We combined it with an LLM (Llama-3-70b) for prompt-based tag generation and a zeroshot classifier (MoritzLaurer/deberta-v3-large-zeroshot-v2.0) for tag filtering. This approach was chosen after evaluating various topic modeling techniques, including LDA, NMF, Top2Vec, and CTM. The rationale for selecting each submodel in the pipeline was provided in sections 3.2.2 and 4.3, highlighting the advantages over naive LLM-based methods in terms of contextual consistency, multi-level tag generation, and computational efficiency. We also explored a cost-effective configuration, omitting description rewriting, using a smaller LLM (GPT-4o-mini), and combining tag generation and classification into a single step. This alternative configuration reduced computational costs by decreasing total time from approximately 25 hours to approximately 6 hours and total cost from $12.80 to $0.80, but resulted in a decrease in tag quality, as reflected in the degraded c-TF-IDF representations and described clustering issues.

In response to **RQ3** (How to objectively optimize the quality of the topics and terms?), we implemented and analyzed multiple evaluation metrics, including NPMI, diversity, and silhouette score. We compared the hyperparameter-optimized Base BERTopic model against several baselines: LDA, NMF, CTM, Top2Vec, and different BERTopic configurations. Statistical analysis using Welch's ANOVA, Kruskal-Wallis, and post-hoc tests confirmed significant differences between models for NPMI and diversity, with large effect sizes. Our optimized BERTopic model achieved the highest combined NPMI and diversity score of 0.779, outperforming all baseline models on this combined metric. Notably, NMF achieved the highest NPMI score but had very low diversity (0.399), while our optimized model had the highest diversity score (0.808). We performed hyperparameter tuning using Bayesian optimization with a custom weighted metric combining NPMI and diversity, exploring a wide range of hyperparameter values as detailed in section 4.2.1. The optimized hyperparameters were all recorded in Table 4.1.

Finally, addressing **RQ4** (How to evaluate the tags from a human perspective?), we conducted a comprehensive human evaluation study with 21 participants, and a large-scale automated evaluation using GPT-4-mini. For the human evaluation, we used the Base BERTopic model as a baseline and created a human-generated set of tags for comparison. The evaluation involved two stages: individual document evaluation and document pair evaluation. In the individual document evaluation stage, participants performed intruder detection tasks and evaluated tag relevance, generality, and coverage. Our model achieved a mean relevance score of 3.63, outperforming the baseline (2.39) and approaching human-generated tags (3.95). For generality, our model showed a better balance between specific and general tags (SD = 1.40) compared to the baseline (SD = 1.33) and was close to human-generated tags (SD = 1.41). In terms of coverage, our model (mean = 4.46) outperformed the baseline (mean = 2.65) and was close to human-generated tags (mean = 4.54). The intruder detection task demonstrated our model's ability to generate cohesive tag sets, with a 95.2% detection rate, compared to 42.1% for the baseline and 100% for human-generated tags. In the document pair evaluation stage, participants identified common tags between document pairs and assessed their shared coverage. Our model achieved balanced precision and recall, with accuracies of 0.87 and 0.76 for the two text pairs, which was higher than the baseline (0.69 and 0.60) and comparable to human-generated tags (0.83 and 0.86). While our model's tags were rated (statistically) significantly lower in relevance compared to human-generated tags, there were no significant differences in generality, coverage, and shared coverage. Inter-rater agreement analysis using Fleiss' Kappa, ICC, and Krippendorff's Alpha showed varying levels of agreement, with some metrics indicating moderate to strong agreement, highlighting the complexity of assessing inter-rater reliability for this task.

The large-scale automated evaluation using GPT-4-mini across the entire OpenML dataset showed that the LLM detected 88.3% of intruders, indicating the cohesiveness of the generated tags. Tag quality assessment by the LLM yielded a mean relevance score of 4.11, a mean generality score of 3.29 (SD = 0.87), and a mean coverage score of 3.72. The LLM found that keyword tags were the most relevant, and slightly more general than regular tags. The evaluation also found a weakly positive (near-neutral) correlation between relevance and generality, differing from the negative correlation observed in the human evaluation.

Our results demonstrate that the proposed model generates high-quality, relevant, and diverse tags for OpenML dataset descriptions, outperforming the baseline model and approaching human-level performance in several aspects. The modular design of our pipeline, along with the exploration of a cost-effective configuration, showcases the flexibility and adaptability of our approach to different resource constraints.

From a practical perspective, our approach provides OpenML with an automated, scalable solution for dataset tagging that produces highly relevant, diverse tags with good coverage. The modular nature of our pipeline ensures that it can evolve alongside advances in language models and embedding techniques, making it a sustainable long-term solution for the platform. The techniques we developed could be applied to similar problems in other domains where automated categorization of content is needed. Our method of combining traditional topic modeling with modern language models and zeroshot text classification represents a novel contribution to the field of topic modeling itself.

Bibliography

[1] 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. ii, 1

[2] Maarten Grootendorst. BERTopic: Neural topic modeling with a class-based TF-IDF procedure, March 2022. ii, 10, 22, 23, 33, 93

[3] Salesforce/SFR-Embedding-2_R · Hugging Face. https://huggingface.co/Salesforce/SFR-Embedding-2_R, December 2024. ii, 23, 47, 49, 53, 54, 71

[4] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive Text Embedding Benchmark, March 2023. ii, 12, 23, 47, 49, 53, 74

[5] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, September 2020. ii, 9, 10, 12, 23

[6] 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. ii, 9, 14, 15, 23, 75

[7] 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. ii, 9, 14, 75

[8] 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. ii, 9, 14, 75

[9] Introducing Meta Llama 3: The most capable openly available LLM to date. https://ai.meta.com/blog/meta-llama-3/, . ii

[10] MilaNLProc/contextualized-topic-models. MilaNLP, October 2024. ii, 49

[11] GPT-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, . ii, 67, 71

[12] David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. ii, 1, 4, 5, 22, 33, 89

[13] 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. ii, 6, 22, 33, 90

[14] 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. ii, 6, 22, 33, 42, 90

[15] 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. ii, 6, 22, 33, 90

[16] Dimo Angelov. Top2Vec: Distributed representations of topics, August 2020. ii, 8, 22, 33, 92

[17] Federico Bianchi, Silvia Terragni, and Dirk Hovy. Pre-training is a Hot Topic: Contextualized Document Embeddings Improve Topic Coherence, June 2021. ii, 33

[18] Federico Bianchi, Silvia Terragni, Dirk Hovy, Debora Nozza, and Elisabetta Fersini. Cross-lingual Contextualized Topic Models with Zero-shot Learning. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1676–1683, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.143. ii, 33

[19] 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, 42

[20] 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, 42

[21] 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

[22] 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

[23] 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

[24] 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

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

[26] 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

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

[28] 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

[29] 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

[30] 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

[31] 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, 19

[32] 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, 21, 90

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

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

[35] Quoc V. Le and Tomas Mikolov. Distributed Representations of Sentences and Documents, May 2014. 8, 92

[36] Radim Řehůřek and Petr Sojka. *Software Framework for Topic Modelling with Large Corpora*. University of Malta, May 2010. ISBN 978-2-9517408-6-0. 8

[37] 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. 8

[38] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, August 2019. 8, 10, 11, 93

[39] Jey Han Lau and Timothy Baldwin. An Empirical Evaluation of doc2vec with Practical Insights into Document Embedding Generation, July 2016. 9

[40] Maarten P. Grootendorst. The Algorithm - BERTopic. https://maartengr.github.io/BERTopic/algorithm/algorithm.html. 10

[41] 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. 10, 12

[42] Maarten Grootendorst. MaartenGr/KeyBERT, February 2024. 10, 18

[43] Nils Reimers and Iryna Gurevych. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation, October 2020. 10

[44] 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. 10, 11, 93

[45] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. 11, 93

[46] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. . 11, 18, 93

[47] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. . 11, 18, 93

[48] 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. 11, 18, 28, 93

[49] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. ISSN 1533-7928. 11

[50] 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. 12, 93

[51] 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. 12

[52] 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. 12

[53] 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. 12

[54] 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. 12

[55] 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. 12

[56] 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. 14

[57] Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. 15

[58] 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. 15, 23

[59] 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. 17

[60] Explosion/spaCy: Industrial-strength Natural Language Processing (NLP) in Python. https://github.com/explosion/spaCy/tree/master, . 18

[61] Moritz Laurer, Wouter van Atteveldt, Andreu Casas, and Kasper Welbers. Building Efficient Universal Classifiers with Natural Language Inference, March 2024. 19, 74

[62] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan Boyd-graber, and David Blei. Reading Tea Leaves: How Humans Interpret Topic Models. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. 20, 37, 41, 75

[63] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic Evaluation of Topic Coherence. In Ron Kaplan, Jill Burstein, Mary Harper, and Gerald Penn, editors, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108, Los Angeles, California, June 2010. Association for Computational Linguistics. 20

[64] Gerlof Bouma. Normalized (Pointwise) Mutual Information in Collocation Extraction. 20

[65] Jey Han Lau, David Newman, and Timothy Baldwin. Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality. In Shuly Wintner, Sharon Goldwater, and Stefan Riezler, editors, *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. doi: 10.3115/v1/E14-1056. 20, 23, 37, 41, 75

[66] Caitlin Doogan and Wray Buntine. Topic Model or Topic Twaddle? Re-evaluating Semantic Interpretability Measures. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3824–3848, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.300. 20, 23, 32, 41

[67] Alexander Hoyle, Pranav Goel, Andrew Hian-Cheong, Denis Peskov, Jordan Boyd-Graber, and Philip Resnik. Is Automated Topic Model Evaluation Broken? The Incoherence of Coherence. In *Advances in Neural Information Processing Systems*, volume 34, pages 2018–2033. Curran Associates, Inc., 2021. 20, 23, 32, 37, 41, 75

[68] Tak Yeon Lee, Alison Smith, Kevin Seppi, Niklas Elmqvist, Jordan Boyd-Graber, and Leah Findlater. The human touch: How non-expert users perceive, interpret, and fix topic models. *International Journal of Human-Computer Studies*, 105:28–42, September 2017. ISSN 1071-5819. doi: 10.1016/j.ijhcs.2017.03.007. 20

[69] David Newman, Youn Noh, Edmund Talley, Sarvnaz Karimi, and Timothy Baldwin. Evaluating topic models for digital libraries. In *Proceedings of the 10th Annual Joint Conference on Digital Libraries*, pages 215–224, Gold Coast Queensland Australia, June 2010. ACM. ISBN 978-1-4503-0085-8. doi: 10.1145/1816123.1816156. 20, 37, 41, 75

[70] David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing Semantic Coherence in Topic Models. 20, 23, 41, 75

[71] 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. 20, 92

[72] Ketan Rajshekhar Shahapure and Charles Nicholas. Cluster Quality Analysis Using Silhouette Score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, October 2020. doi: 10.1109/DSAA49011.2020.00096. 21

[73] Silvia Terragni, Elisabetta Fersini, Bruno Giovanni Galuzzi, Pietro Tropeano, and Antonio Candelieri. OCTIS: Comparing and Optimizing Topic models is Simple! In Dimitra Gkatzia and Djamé Seddah, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 263–270, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-demos.31. 21, 23, 41

[74] Francesco Archetti and Antonio Candelieri. *Bayesian Optimization and Data Science*. SpringerBriefs in Optimization. Springer International Publishing, Cham, 2019. ISBN 978-3-030-24493-4 978-3-030-24494-1. doi: 10.1007/978-3-030-24494-1. 21, 47

[75] B. G. Galuzzi, I. Giordani, A. Candelieri, R. Perego, and F. Archetti. Hyperparameter optimization for recommender systems through Bayesian optimization. *Computational Management Science*, 17(4):495–515, December 2020. ISSN 1619-6988. doi: 10.1007/s10287-020-00376-3. 21, 47

[76] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. 21, 47

[77] 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. 22, 93

[78] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013. 22, 91

[79] Laure Thompson and David Mimno. Topic Modeling with Contextualized Word Representation Clusters. https://arxiv.org/abs/2010.12626v1, October 2020. 22, 93

[80] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023. 28

[81] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?, October 2022. 28

[82] Sonish Sivarajkumar, Mark Kelley, Alyssa Samolyk-Mazzanti, Shyam Visweswaran, and Yanshan Wang. An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing: Algorithm Development and Validation Study. *JMIR Medical Informatics*, 12(1):e55318, April 2024. doi: 10.2196/55318. 28

[83] MoritzLaurer/deberta-v3-large-zeroshot-v2.0 · Hugging Face. https://huggingface.co/MoritzLaurer/deberta-v3-large-zeroshot-v2.0, April 2024. 31, 40, 54

[84] Tomáš Musil and David Mareček. Exploring Interpretability of Independent Components of Word Embeddings with Automated Word Intruder Test. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6922–6928, Torino, Italia, May 2024. ELRA and ICCL. 37, 39, 41, 75

[85] Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. An Automatic Approach for Document-level Topic Model Evaluation, June 2017. 37, 41, 75

[86] 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. 42, 91

[87] 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. 42, 91

[88] 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. 42, 90

[89] Rudolph Flesch. A new readability yardstick. *Journal of Applied Psychology*, 32(3):221–233, 1948. ISSN 1939-1854. doi: 10.1037/h0057532. 45

[90] Ivan Germanov. Topic modeling of OpenML dataset descriptions, October 2024. URL `https://github.com/ivangermanov/openml-tags`. 45, 53, 56, 59, 74

[91] G. K. Zipf. *The Psycho-Biology of Language*. The Psycho-Biology of Language. Houghton, Mifflin, Oxford, England, 1935. 55

[92] Steven T. Piantadosi. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic Bulletin & Review*, 21(5):1112–1130, October 2014. ISSN 1531-5320. doi: 10.3758/s13423-014-0585-6. 55

[93] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with Disentangled Attention, October 2021. 74

[94] Akshay Balsubramani, Sanjoy Dasgupta, and Yoav Freund. The Fast Convergence of Incremental PCA. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. 75

[95] M. Artac, M. Jogan, and A. Leonardis. Incremental PCA for on-line visual learning and recognition. In *2002 International Conference on Pattern Recognition*, volume 3, pages 781–784 vol.3, August 2002. doi: 10.1109/ICPR.2002.1048133. 75

[96] Issam Dagher. Incremental PCA-LDA algorithm. In *2010 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, pages 97–101, September 2010. doi: 10.1109/CIMSA.2010.5611752. 75

[97] Jacob Montiel, Hoang-Anh Ngo, Minh-Huong Le-Nguyen, and Albert Bifet. Online Clustering: Algorithms, Evaluation, Metrics, Applications and Benchmarking. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, pages 4808–4809, New York, NY, USA, August 2022. Association for Computing Machinery. ISBN 978-1-4503-9385-0. doi: 10.1145/3534678.3542600. 75

[98] Javier Béjar Alonso. K-means vs Mini Batch K-means: A comparison. May 2013. 75

[99] Stephanie C. Hicks, Ruoxi Liu, Yuwei Ni, Elizabeth Purdom, and Davide Risso. Mbkmeans: Fast clustering for single cell data using mini-batch k-means. *PLOS Computational Biology*, 17(1):e1008625, January 2021. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1008625. 75

[100] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114, June 1996. ISSN 0163-5808. doi: 10.1145/235968.233324. 75

[101] 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. 89

[102] 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. 89

[103] 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. 89

[104] Pierre Simon Laplace (marquís de). *Théorie analytique des probabilités.* Courcier, 1814. 89

[105] 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. 89

[106] 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. 89

[107] John Lafferty and David Blei. Correlated Topic Models. In *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2005. 89

[108] 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. 89

[109] 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. 89

[110] 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. 89

[111] Chong Wang, David Blei, and David Heckerman. Continuous Time Dynamic Topic Models. https://arxiv.org/abs/1206.3298v2, June 2012. 89

[112] Tomoharu Iwata, Shinji Watanabe, Takeshi Yamada, and Naonori Ueda. *Topic Tracking Model for Analyzing Consumer Purchase Behavior.* January 2009. 89

[113] 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. 89

[114] 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. 89

[115] 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. 89

[116] 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. 89

[117] Jon Mcauliffe and David Blei. Supervised Topic Models. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. 90

[118] 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. 90

[119] 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. 90

[120] 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. 90

[121] 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. 90, 91

[122] 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. 90

[123] 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. 91

[124] 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. 91

[125] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey, May 2023. 91

[126] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A Neural Probabilistic Language Model. In *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2000. 91

[127] 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. 91

[128] 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. 91

[129] 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. 92

[130] 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. 92

[131] Christopher E. Moody. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec, May 2016. 92

[132] 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. 92

[133] 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. 92

[134] 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. 92

[135] 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. 92

[136] Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. The Dynamic Embedded Topic Model, October 2019. 92

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. [101]. 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 [102] 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. [103] 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 [104]. 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. [12] 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 [105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116].

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 [117]. 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 [32]. 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 [32]. 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. [13]. They showcased the potential of NMF as a tool for extracting topics from a collection of documents. Following this, [14] expanded the application of NMF to temporal topic models. Yan et al. [15] 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. [118] 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. [119] 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 [88] 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. [120] 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. [121], 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 [88]'s GSDMM.

Building upon DMM, Li et al. [122] 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. [86] were among the first to implement a graph-based model using a directed graph to detect emerging topics. Subsequently, de Arruda et al. [123] 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. [124], applying graph-based methods to track the evolution of topics over time. Following this, Churchill and Singh [87] 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 [125], 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. [126], 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. [78]. 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. [121] 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. [127] 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 [128] 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. [129] 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. [130] 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 [131] 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 [35]. 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 [16] 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. [132] 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. [133] 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 [134] and CluHTM [135], 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) [71] and the Dynamic Embedded Topic Model (D-ETM) [136]. 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 [44], 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. [44], OpenAI introduced the GPT-1, GPT-2 and GPT-3 models [46, 47, 48]. 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. [45] 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. [44], 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 [50] and Sentence-BERT (SBERT) [38].

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 [38], 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 [79] later proposed using BERT [45] 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 [77] and Word2Vec which are context-free embedding spaces with a single embedding representation for each word.

Grootendorst [2] 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.

Appendix B

Figures

Figure B.1 shows the top 50 regular tags by frequency, while Figure B.2 shows the top 50 overarching tags by frequency. We see that the tags are quite diverse, covering a wide range of topics. The regular tags are more specific, while the overarching tags are more general.

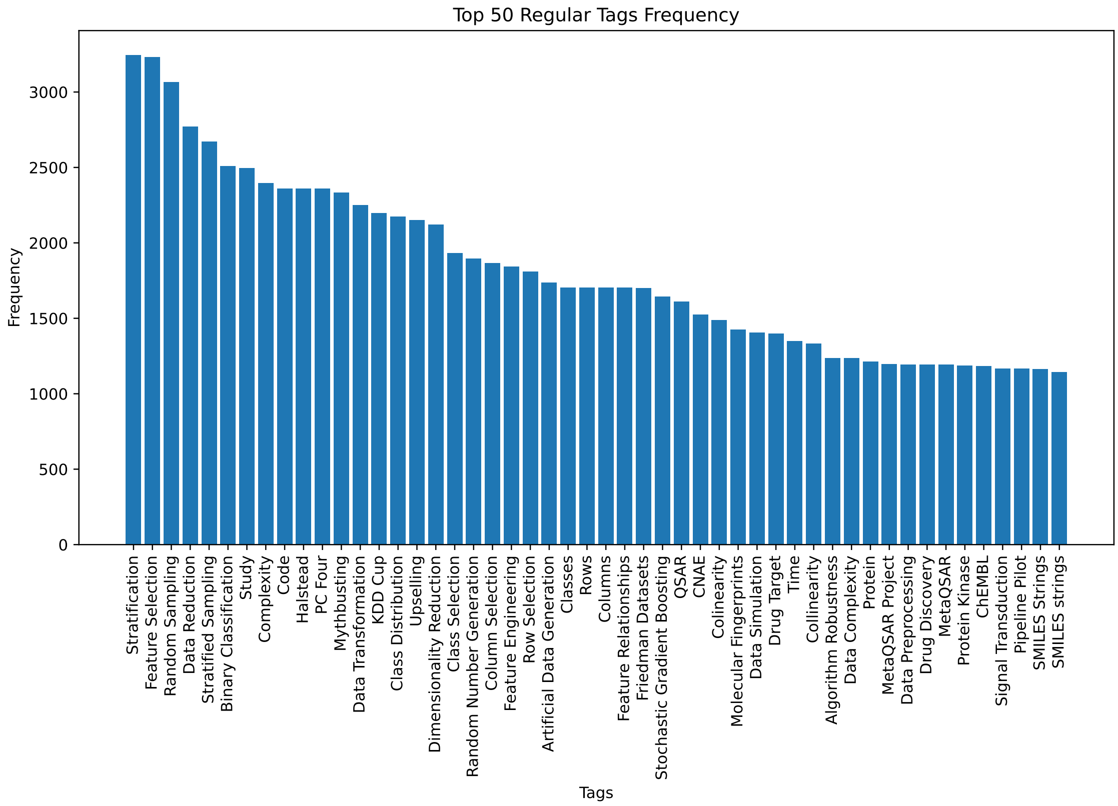


Figure B.1: Top 50 tags by frequency

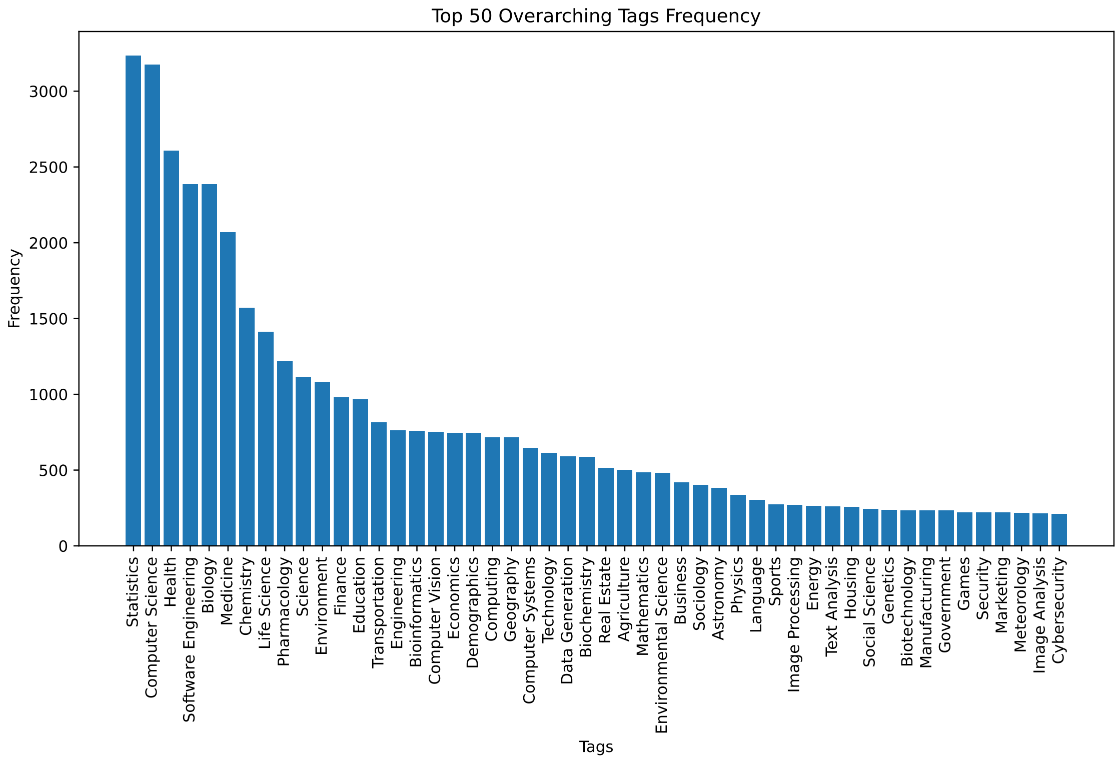


Figure B.2: Top 50 overarching tags by frequency

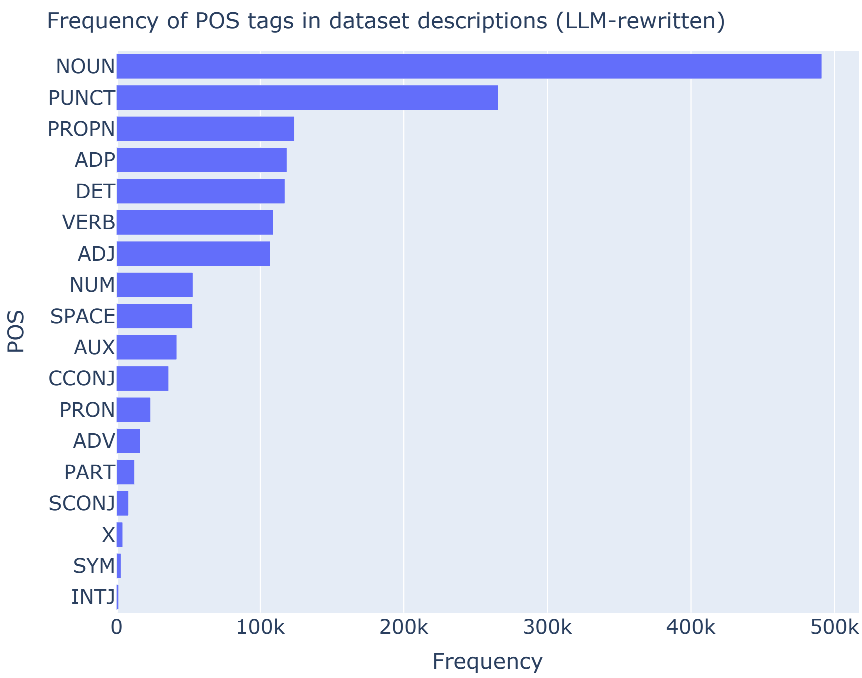


Figure B.3: Bar chart of the most common parts of speech in dataset descriptions (LLM-rewritten)

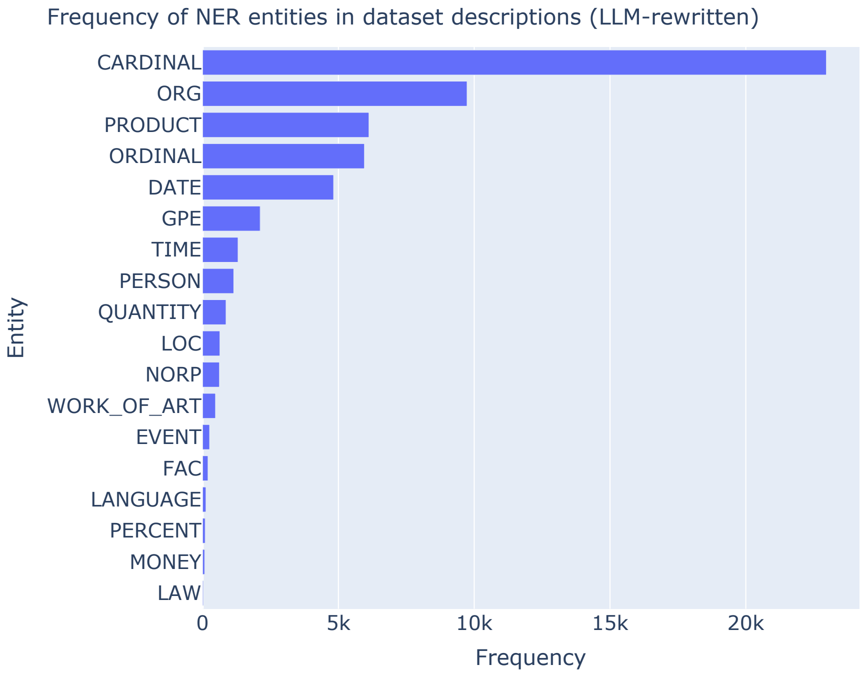


Figure B.4: Bar chart of the most common named entities in dataset descriptions

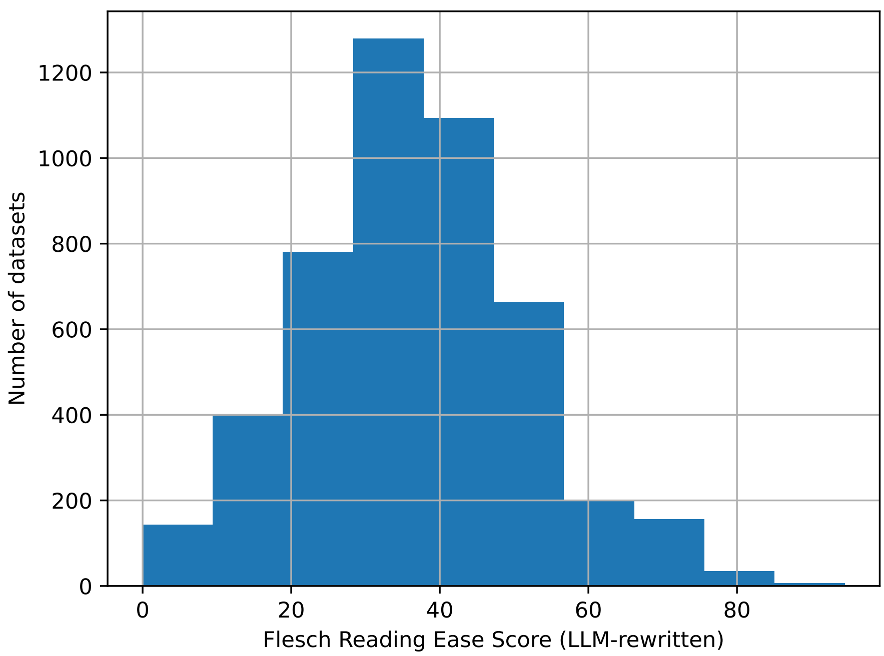
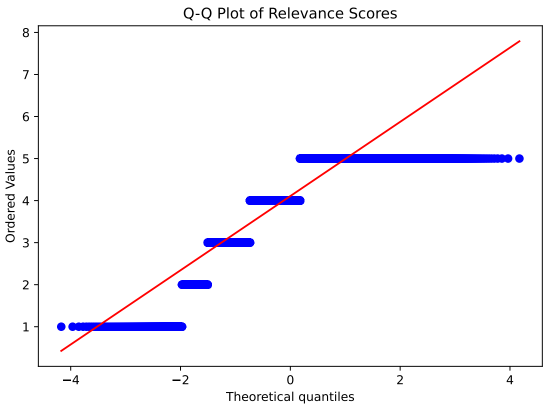
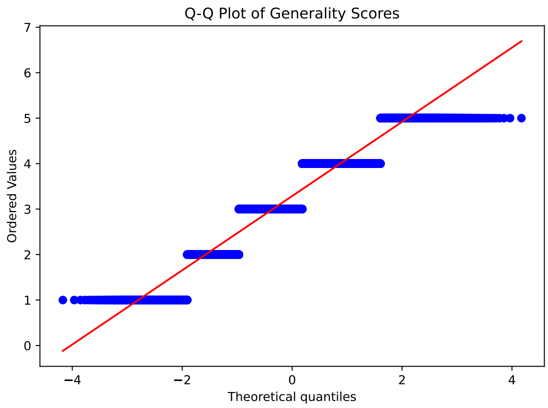


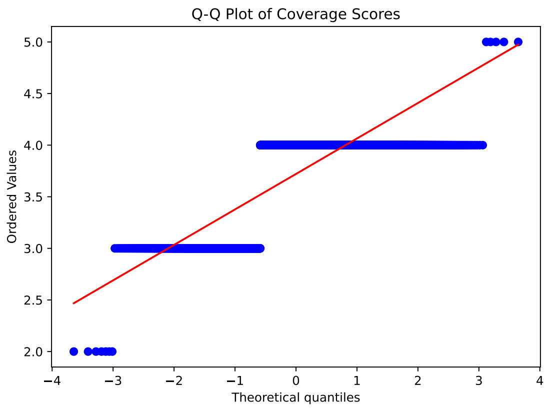
Figure B.5: Histogram of the Flesch Reading Ease scores for dataset descriptions (LLM-rewritten)



(a) Relevance scores



(b) Generality scores



(c) Coverage scores

Figure B.6: Q-Q plots for evaluation metrics

Appendix C

Tables

Table C.1: Games-Howell post-hoc test results for NPMI

Comparison (A vs. B)	Diff	p-value	Hedges' g
B_POS vs. B_POS_mpnet	-0.0057	0.9385	-0.197
B_POS vs. B_OPT_FULL	-0.0013	0.999988	-0.050
B_POS vs. B_OPT_REDUCED	0.0423	1.498801e-14	1.546
B_POS vs. CTM	0.0021	0.999622	0.085
B_POS vs. LDA	0.0300	7.661649e-13	1.439
B_POS vs. NMF	-0.0687	4.252154e-14	-2.993
B_POS vs. Top2Vec	0.2147	0.000000	12.024
B_POS_mpnet vs. B_OPT_FULL	0.0044	0.990508	0.141
B_POS_mpnet vs. B_OPT_REDUCED	0.0480	1.374456e-13	1.488
B_POS_mpnet vs. CTM	0.0077	0.779209	0.260
B_POS_mpnet vs. LDA	0.0356	8.421752e-11	1.324
B_POS_mpnet vs. NMF	-0.0630	3.841372e-14	-2.205
B_POS_mpnet vs. Top2Vec	0.2203	0.000000	8.929
B_OPT_FULL vs. B_OPT_REDUCED	0.0436	1.706413e-13	1.471
B_OPT_FULL vs. CTM	0.0034	0.995362	0.125
B_OPT_FULL vs. LDA	0.0313	6.274092e-11	1.317
B_OPT_FULL vs. NMF	-0.0674	0.000000	-2.631
B_OPT_FULL vs. Top2Vec	0.2160	1.643130e-14	10.202
B_OPT_REDUCED vs. CTM	-0.0403	1.185607e-12	-1.422
B_OPT_REDUCED vs. LDA	-0.0124	0.085287	-0.485
B_OPT_REDUCED vs. NMF	-0.1110	1.110223e-16	-4.074
B_OPT_REDUCED vs. Top2Vec	0.1724	6.661338e-16	7.455
CTM vs. LDA	0.0279	2.411494e-10	1.266
CTM vs. NMF	-0.0707	0.000000	-2.940
CTM vs. Top2Vec	0.2126	9.992007e-15	11.039
LDA vs. NMF	-0.0987	0.000000	-4.776
LDA vs. Top2Vec	0.1847	0.000000	12.492
NMF vs. Top2Vec	0.2834	1.543210e-14	16.054

Table C.2: Games-Howell post-hoc test tesults for diversity

Comparison (A vs. B)	**Diff**	**p-value**	**Hedges' g**
B_POS vs. B_POS_mpnet	0.0431	0.000041	0.853
B_POS vs. B_OPT_FULL	-0.0278	0.000044	-0.845
B_POS vs. B_OPT_REDUCED	-0.0378	0.000072	-0.827
B_POS vs. CTM	0.0921	0.000849	0.741
B_POS vs. LDA	0.3587	4.329870e-15	10.075
B_POS vs. NMF	0.3706	3.330669e-15	4.486
B_POS vs. Top2Vec	0.4058	4.107825e-15	5.537
B_POS_mpnet vs. B_OPT_FULL	-0.0709	1.008860e-12	-1.494
B_POS_mpnet vs. B_OPT_REDUCED	-0.0809	1.283196e-12	-1.417
B_POS_mpnet vs. CTM	0.0490	0.326768	0.380
B_POS_mpnet vs. LDA	0.3156	1.221245e-15	6.387
B_POS_mpnet vs. NMF	0.3275	2.609024e-14	3.662
B_POS_mpnet vs. Top2Vec	0.3627	1.887379e-15	4.483
B_OPT_FULL vs. B_OPT_REDUCED	-0.0100	0.852516	-0.236
B_OPT_FULL vs. CTM	0.1199	4.375404e-06	0.975
B_OPT_FULL vs. LDA	0.3865	0.000000	12.439
B_OPT_FULL vs. NMF	0.3984	0.000000	4.933
B_OPT_FULL vs. Top2Vec	0.4336	0.000000	6.090
B_OPT_REDUCED vs. CTM	0.1299	9.880746e-07	1.023
B_OPT_REDUCED vs. LDA	0.3965	0.000000	8.929
B_OPT_REDUCED vs. NMF	0.4084	0.000000	4.707
B_OPT_REDUCED vs. Top2Vec	0.4436	2.919887e-14	5.691
CTM vs. LDA	0.2666	0.000000	2.154
CTM vs. NMF	0.2785	5.573320e-14	1.928
CTM vs. Top2Vec	0.3137	0.000000	2.251
LDA vs. NMF	0.0120	9.880892e-01	0.146
LDA vs. Top2Vec	0.0471	5.115797e-03	0.649
NMF vs. Top2Vec	0.0351	4.794142e-01	0.338

Type	**ICC**	**F**	**p**	**CI95%**
ICC1	0.45	16.65	1.68e-40	[0.31, 0.64]
ICC2	0.46	25.92	1.18e-57	[0.31, 0.65]
ICC3	0.57	25.92	1.18e-57	[0.42, 0.74]
ICC1k	0.94	16.65	1.68e-40	[0.90, 0.97]
ICC2k	0.94	25.92	1.18e-57	[0.89, 0.97]
ICC3k	0.96	25.92	1.18e-57	[0.93, 0.98]

(a) Relevance

Type	**ICC**	**F**	**p**	**CI95%**
ICC1	-0.03	0.53	0.711	[-0.04, 0.15]
ICC2	0.00	0.99	0.416	[-0.02, 0.17]
ICC3	0.00	0.99	0.416	[-0.04, 0.28]
ICC1k	-0.87	0.53	0.711	[-4.49, 0.78]
ICC2k	0.00	0.99	0.416	[-0.56, 0.80]
ICC3k	-0.01	0.99	0.416	[-1.98, 0.88]

(b) Generality

Type	**ICC**	**F**	**p**	**CI95%**
ICC1	0.33	10.16	1.80e-04	[0.08, 0.95]
ICC2	0.34	21.51	7.14e-07	[0.10, 0.95]
ICC3	0.52	21.51	7.14e-07	[0.18, 0.98]
ICC1k	0.90	10.16	1.80e-04	[0.61, 1.00]
ICC2k	0.91	21.51	7.14e-07	[0.67, 1.00]
ICC3k	0.95	21.51	7.14e-07	[0.81, 1.00]

(c) Coverage

Type	**ICC**	**F**	**p**	**CI95%**
ICC1	0.04	1.69	0.202	[-0.04, 0.99]
ICC2	0.06	3.65	0.072	[-0.01, 0.99]
ICC3	0.12	3.65	0.072	[-0.02, 0.99]
ICC1k	0.41	1.69	0.202	[-2.24, 1.00]
ICC2k	0.55	3.65	0.072	[-0.18, 1.00]
ICC3k	0.73	3.65	0.072	[-0.64, 1.00]

(d) Shared Coverage

Figure C.1: ICC values for different rating types (baseline model)

Type	ICC	F	p	CI95%
ICC1	0.34	11.93	2.88e-30	[0.22, 0.53]
ICC2	0.35	19.27	7.72e-47	[0.22, 0.54]
ICC3	0.47	19.27	7.72e-47	[0.32, 0.65]
ICC1k	0.92	11.93	2.88e-30	[0.85, 0.96]
ICC2k	0.92	19.27	7.72e-47	[0.86, 0.96]
ICC3k	0.95	19.27	7.72e-47	[0.91, 0.98]

(a) Relevance

Type	ICC	F	p	CI95%
ICC1	0.46	18.68	1.94e-35	[0.30, 0.68]
ICC2	0.46	31.71	4.86e-53	[0.30, 0.69]
ICC3	0.59	31.71	4.86e-53	[0.43, 0.78]
ICC1k	0.95	18.68	1.94e-35	[0.90, 0.98]
ICC2k	0.95	31.71	4.86e-53	[0.90, 0.98]
ICC3k	0.97	31.71	4.86e-53	[0.94, 0.99]

(b) Generality

Type	ICC	F	p	CI95%
ICC1	0.05	2.06	0.137	[-0.02, 0.79]
ICC2	0.07	5.57	0.007	[0.01, 0.79]
ICC3	0.18	5.57	0.007	[0.02, 0.91]
ICC1k	0.51	2.06	0.137	[-0.91, 0.99]
ICC2k	0.63	5.57	0.007	[0.12, 0.99]
ICC3k	0.82	5.57	0.007	[0.27, 1.00]

(c) Coverage

Type	ICC	F	p	CI95%
ICC1	0.19	5.99	0.019	[0.00, 1.00]
ICC2	0.20	10.25	0.004	[0.02, 1.00]
ICC3	0.31	10.25	0.004	[0.03, 1.00]
ICC1k	0.83	5.99	0.019	[0.09, 1.00]
ICC2k	0.84	10.25	0.004	[0.34, 1.00]
ICC3k	0.90	10.25	0.004	[0.43, 1.00]

(d) Shared Coverage

Figure C.2: ICC values for different rating types (proposed model)

Type	ICC	F	p	CI95%
ICC1	0.13	3.70	5.26e-07	[0.06, 0.28]
ICC2	0.15	8.70	7.21e-20	[0.07, 0.31]
ICC3	0.30	8.70	7.21e-20	[0.18, 0.50]
ICC1k	0.73	3.70	5.26e-07	[0.52, 0.87]
ICC2k	0.77	8.70	7.21e-20	[0.59, 0.89]
ICC3k	0.89	8.70	7.21e-20	[0.80, 0.95]

(a) Relevance

Type	ICC	F	p	CI95%
ICC1	0.34	10.38	9.38e-21	[0.20, 0.56]
ICC2	0.35	16.76	5.01e-32	[0.21, 0.57]
ICC3	0.47	16.76	5.01e-32	[0.31, 0.68]
ICC1k	0.90	10.38	9.38e-21	[0.82, 0.96]
ICC2k	0.91	16.76	5.01e-32	[0.83, 0.96]
ICC3k	0.94	16.76	5.01e-32	[0.89, 0.97]

(b) Generality

Type	ICC	F	p	CI95%
ICC1	-0.02	0.59	0.560	[-0.05, 0.55]
ICC2	0.00	1.09	0.348	[-0.02, 0.56]
ICC3	0.00	1.09	0.348	[-0.04, 0.70]
ICC1k	-0.71	0.59	0.560	[-5.77, 0.96]
ICC2k	0.05	1.09	0.348	[-0.65, 0.96]
ICC3k	0.08	1.09	0.348	[-2.78, 0.98]

(c) Coverage

Type	ICC	F	p	CI95%
ICC1	0.10	3.09	0.088	[-0.02, 0.99]
ICC2	0.12	4.25	0.055	[-0.01, 0.99]
ICC3	0.15	4.25	0.055	[-0.02, 1.00]
ICC1k	0.68	3.09	0.088	[-0.78, 1.00]
ICC2k	0.70	4.25	0.055	[-0.20, 1.00]
ICC3k	0.76	4.25	0.055	[-0.42, 1.00]

(d) Shared Coverage

Figure C.3: ICC values for different rating types (human-generated)

Metric	Comparison	Cohen's d	Effect
Relevance	Baseline model vs Human-generated	-1.589	Very Large
	Baseline model vs Proposed model	-1.254	Very Large
	Human-generated vs Model	0.293	Small-Medium
Generality	Baseline model vs Human-generated	0.027	Negligible
	Baseline model vs Proposed model	0.202	Small-Medium
	Human-generated vs Proposed model	0.178	Small
Coverage	Baseline model vs Human-generated	-2.454	Very Large
	Baseline model vs Proposed model	-2.539	Very Large
	Human-generated vs Proposed model	-0.078	Negligible
Shared Coverage	Baseline model vs Human-generated	-1.882	Very Large
	Baseline model vs Proposed model	-1.485	Very Large
	Human-generated vs Proposed model	0.391	Small-Medium

Table C.3: Cohen's d effect sizes for different metrics and comparisons