UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Master degree in Data Science

Data Science Lab course

A Text Analysis using Topic Modeling and Text Classification

Authors:
Andrea Cardinali, Adonis Kingsley Granita, Biagio Spiezia

2024/2025

Contents

1 Introduction **2**

2 Data Exploration and preliminary steps **2**
2.1 TF-IDF . . . 3

3 LSA **4**
3.1 LSA preprocessing . . . 4

4 pLSA **5**
4.1 Pre-processing . . . 6
4.2 Model and results . . . 6
4.3 Assigning titles to topics . . . 6

5 LDA **7**
5.1 Pre-processing . . . 8
5.2 Analysis of some results of topics . . . 8
5.3 Assigning Titles to Topics . . . 8
5.4 Clustering Topics into Groups . . . 10
5.4.1 Preliminary Choice of Number of Clusters with K Arbitrary . . . 10
5.4.2 Number of clusters selected with Silhouette Analysis . . . 11
5.4.3 Clustering Results and Interpretation . . . 11

6 BERT **12**
6.1 Pre-processing . . . 12
6.2 Model . . . 12

7 How is Topic Modeling evaluated? **12**

8 Comparing results **13**

9 Text Classification **14**
9.1 Models of text classification . . . 14
9.2 CountVectorizer and TF-IDF . . . 17

10 Modeling Topic Distributions on the Simplex **18**
10.1 Compositional Data and the Simplex . . . 18
10.2 Dirichlet Regression . . . 18
10.3 Neural Network-Based Modeling . . . 18
10.4 Evaluation Metrics for Distributional Outputs . . . 18
10.5 Model Configuration . . . 19
10.6 Performance Metrics . . . 20
10.7 Topic-wise Error Analysis . . . 20
10.8 Conclusions . . . 22

1 Introduction

We live in the information age, surrounded by an overwhelming amount of digital text, ranging from news articles and product reviews to social media content and institutional documents. The ability to extract relevant information and structured insights from such unstructured data is a fundamental challenge in data science.

To address this challenge, machine learning offers two complementary paradigms: unsupervised and supervised learning. Unsupervised techniques aim to uncover hidden patterns or structures in the data without relying on predefined labels. They are particularly useful for exploratory analysis, such as discovering thematic clusters or organizing content based on similarity. Supervised techniques, on the other hand, learn from labeled examples to predict known categories or outcomes, making them suitable for targeted classification or decision-making tasks.

In this project, we applied both approaches in distinct phases to analyze and classify a large collection of textual documents.

In the first part of the project, we employed topic modeling, an unsupervised technique that automatically identifies latent themes within a corpus. The core assumption is that each document consists of a mixture of topics, and each topic is characterized by a distribution over words. We compared several topic modeling methods including LSA, pLSA, LDA, and BERT based approaches to evaluate their ability to uncover meaningful thematic structures in the data.

To support this analysis, we implemented a comprehensive preprocessing pipeline to clean and normalize the textual data, ensuring consistent and informative representations across models.

In the second phase, we adopted supervised learning techniques to build models capable of classifying documents based on their content. For this task, we used the available topic labels in the dataset as ground truth, allowing the models to learn from examples and assign categories to new, unseen documents.

By applying both paradigms, the project provided a well-rounded view of how textual data can be explored, interpreted, and organized, combining the discovery power of unsupervised learning with the precision and applicability of supervised methods

2 Data Exploration and preliminary steps

The dataset consists of a parquet file containing approximately 205,000 observations. Each entry includes a document title, the full text, a main topic label (topic), and a soft assignment over 40 possible topics stored in the field topics-with-percentages. These topic annotations were not generated by us, but were pre-assigned as part of the dataset using a proprietary or pre-trained topic modeling algorithm. Each topics-with-percentages field is a JSON-formatted dictionary mapping each topic to a probability value between 0 and 1. These values sum to 1 and represent the degree to which the document is associated with each topic. The topic field contains a single categorical label corresponding to the topic with the highest probability. In cases where one topic showed a sufficiently dominant probability compared to the others, the corresponding label was assigned in the topic field. Conversely, when the topic distribution was more uniform — that is, when no single topic clearly stood out in terms of probability — the label "Mixed" was used to indicate ambiguity or overlapping thematic content. There are 1,700 such documents. To avoid disproportionate scaling in visualizations, this category was excluded from the initial topic distribution plot (Figure 1).

1.

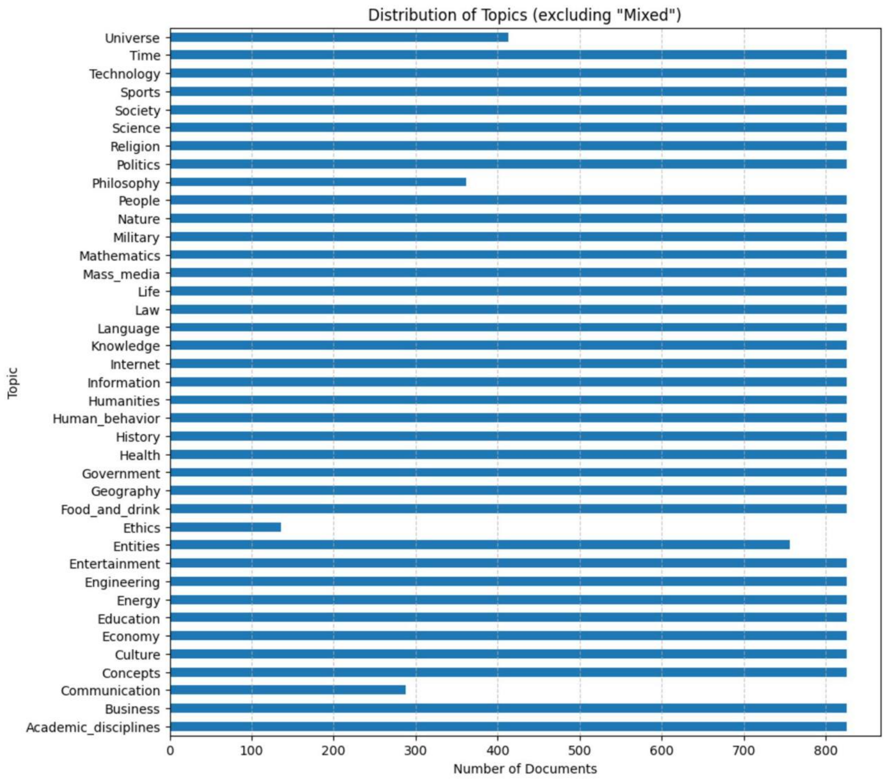

Figure 1: Topic distribution excluding "mixed"

To facilitate analysis and reduce computational load, we performed stratified sampling, selecting 30 % of the documents from each specific topic (excluding the "Mixed" category). This yielded a balanced subset of 60,019 documents, with preserved topic proportions and a fixed random seed to ensure reproducibility.

2.1 TF-IDF

To represent documents numerically for topic modeling, the Term Frequency-Inverse Document Frequency (TF-IDF) weighting scheme was applied before training the Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (pLSA). This transformation assigns importance to terms based on their occurrence in individual documents relative to their distribution across the corpus.

The TF-IDF score for a term t in a document d is computed as:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

where:

- $\text{TF}(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$ represents the term frequency, defined as the count of term t in document d divided by the total number of terms in d, ensuring normalization across documents of different lengths.

- $\text{IDF}(t) = \log\left(\frac{N}{n_t}\right)$ represents the inverse document frequency, where N is the total number of documents in the corpus and n_t is the number of documents containing the term t. This factor reduces the weight of frequent terms appearing across many documents, emphasizing discriminative words.

By applying TF-IDF before the models, the impact of uninformative terms is minimized, improving topic coherence and interpretability.

3 LSA

Latent Semantic Analysis (LSA) is a dimensionality reduction technique applied to topic modeling that utilizes Singular Value Decomposition (SVD) to decompose a document-term matrix into three lower-dimensional matrices. The primary objective of LSA is to capture latent semantic structures in a corpus by reducing noise and highlighting meaningful topic representations.

The process begins with the construction of the document-term matrix A, where each row corresponds to a document, each column to a term, and the values represent term frequency-inverse document frequency (TF-IDF) weights. This matrix is decomposed as follows:

$$A = USV^T$$

where:

- U is the document-topic matrix, where each row represents a document in the reduced topic space,
- S is a diagonal matrix containing singular values that represent the importance of each topic,
- V^T is the topic-term matrix, indicating the relationship between topics and terms, describing the contribution of terms to topics.

For this study, we retained the top 40 singular values, yielding an approximation $A_k = U_k S_k V_k^T$, where $k = 40$ represents the number of latent topics extracted. The transformation applied in our analysis computed both $U_k S_k$ and V_k^T.The highest values in the product $U_k S_k$ correspond to the most representative topics for each document. These values indicate the strongest associations between documents and the extracted latent concepts, as they are weighted by the singular values in S_k. As a result, documents with high values in specific dimensions of $U_k S_k$ are strongly related to those topics, making this representation effective for identifying dominant themes within the corpus.

LSA effectively captures hidden semantic relationships between words and documents. However, its reliance on linear algebra rather than probabilistic modeling means it does not explicitly capture word co-occurrence probabilities.

3.1 LSA preprocessing

The preprocessing steps performed for the LSA model were:

- HTML decoding: HTML entities such as ‘&quot;‘ were decoded, and any remaining HTML tags were removed using a parser to retain only the raw text content.

- Removal of links and unnecessary content: External links (e.g., ‘http://example.com‘), file references (e.g., ‘File:...‘), and placeholder patterns were removed to eliminate non-informative elements.
- Punctuation and special character handling: Punctuation marks, including dashes and vertical bars (‘|‘), were removed to ensure consistency in tokenization and feature extraction.
- Case normalization: All text was converted to lowercase to prevent duplicate representations of the same word due to capitalization differences.
- Tokenization: The text was split into individual words using a word punctuator tokenizer.
- Stopword removal: Common English stopwords, such as "the" and "and," were removed to retain only content-bearing words.
- Lemmatization: Each token was reduced to its base form using a lexical database (e.g., "running" $\rightarrow$ "run").
- Removal of non-alphabetic characters: Numbers and non-alphabetic characters were eliminated to prevent numerical noise from affecting topic extraction.

The final output of this process consisted of a cleaned text representation and a list of processed tokens, both of which were subsequently used to construct the document-term matrix for LSA.

4 pLSA

pLSA, or probabilistic latent semantic analysis, is a technique used in topic modeling based on a mixture decomposition of a latent model, which relies on statistics instead of linear algebra like LSA. The idea is then to find a probabilistic model with latent topics that is able to generate the observed documents (or texts). This means finding a model $P(D, W)$, where D is the collection of documents and W are the words, which correspond to that for each d and w in D and W respectively in the term-document matrix.

Three variables can be identified in the pLSA technique:

- Documents $d \in D$ are observed variables, defined by the size of the corpus.
- Words $w \in W$ are also observed variables and are the distinct words in the corpus.
- Topics $z \in Z$ are latent variables and their number must be specified a priori.

The model can be defined then as: $P(D, W) = \prod_{(d,w)} P(d, w)$ According to the conditional independence and Bayesian rule from the probability theory, the model can be firstly written as

$$P(w, d) = \sum_{z \in Z} P(z)P(d|z)P(w|z)$$

then as

$$P(w, d) = P(d) \sum_{z \in Z} P(z|d)P(w|z)$$

where $P(z|d)$ is the probability of topic z occurring in document d and $P(w|z)$ is the probability of observing the word w in topic z. To learn all three probabilities in a way that they're as close as possible to the real data it can be used the EM algorithm, an iterative procedure in two steps of Expectation and Maximization. In this case however, it was chosen to use an equivalent method that uses the Non-Negative matrix factorization (NMF) with the Kullback-Liebler divergence as the objective function.

4.1 Pre-processing

The actual analysis must be performed on processed data and not on raw texts. The basic steps taken in order are deleting website links, tokenization, punctuation and stop-words removal and lemmatization. This avoids results of topics related to non-meaningful and very frequent words such as "and", "or", "the", and so on. For each document in the collection, the words are vectorized using the tf-idf vectorizer, which takes into account the importance of each word in all documents of the collection. A limitation on the size of the vocabulary of 1000 (the most frequent words) was used to save memory and try to improve performance. The tokenized text is then converted back to the sentence by joining words with a space as separator, and then the model is applied.

4.2 Model and results

The NMF model is then applied to the text with the aim of finding exactly forty topics, just like in the original dataset. The components of the model are the term-topic matrices, in which it's possible to extract the ten most important words and their weight in each specific topic.

The results show that some topics have a more defined structured than others. For example, the first topic found has the word "quote" as its most representative, which is not really meaningful per se, or even another topic with the word "son". Other topics, like n.4, n.7 and n.13 which can be identified as "movies", "railways", and "education", respectively, have more informative words and can be easily associated with a meaningful name. There are also topics in which the top words are not related to each other, possibly causing problems when trying to find a name for the topic.

Overall, the model seems able to find meaningful topics in the collection.

4.3 Assigning titles to topics

By using a pre-trained word embedding model like Word2Vec, it is possible to name each topic found from the top ten most common and semantically similar words in each one of them. The function creates a vector of length ten that is the result of averaging the vectors of words from the embedding model, which is considered to represent the central theme of the topic.

The function then tries to generate a title for a topic by searching for words in the embedding model that are most similar to the topic vector, then filtering out by removing english stop words, single character words and numbers. Each topic with its title is reported in the following table:

The result shows that some topics appears to be labeled quite correctly, for example topic n. 4 is named "film", n. 13 is "school", n. 17 is "church" and n. 19 is "industry". However, there are also some topics that don't actually have a very meaningful name like topic n. 1 is called "way", n. 2 is "kissane" and n. 15 is just ";" which should have been removed from the words list during the pre-processing steps.

Topic	Title
1	Way
2	Kissane
3	League
4	Film
5	East
6	White
7	Station
8	Album
9	Elected
10	Tournament
11	Building
12	Web
13	School
14	October
15	;
16	Father
17	Church
18	Series
19	Industry
20	William
21	Army
22	First
23	Institute
24	Century
25	Particular
26	Available
27	North
28	New
29	Art
30	Form
31	Book
32	European
33	Vehicle
34	Official
35	Ireland
36	Olympics
37	South
38	Law
39	Chinese
40	California

Table 1: Titles for each topic

5 LDA

Latent Dirichlet Allocation (LDA) is a probabilistic generative model used for topic modeling, where each document is represented as a mixture of latent topics, and each topic is characterized by a distribution over words. Unlike Latent Semantic Analysis (LSA), which applies Singular Value Decomposition (SVD) to reduce dimensionality, LDA assumes a Bag of Words (BoW) representation, ignoring word order and capturing only word frequency. Mathematically, given a corpus of M documents and k topics, LDA models each document d as follows:

- Choose a topic distribution $\theta_d \sim Dir(\alpha)$.
- For each word $w_{d,n}$ in document d:
- Sample a topic $z_{d,n} \sim Mult(\theta_d)$.
- Sample a word $w_{d,n} \sim Mult(\phi_{z_{d,n}})$.

Here, θ_d represents the distribution over topics for document d, drawn from a Dirichlet distribution $Dir(\alpha)$, which acts as a prior enforcing that documents contain only a few dominant topics. Similarly, each topic k has a word distribution $\phi_k \sim Dir(\beta)$, ensuring that topics focus on a subset of words. The probability of a document is:

$$P(w_d) = \prod_{n=1}^{N_d} \sum_{z=1}^{k} P(w_{d,n}|z, \phi) P(z|\theta_d).$$

LDA differs from Probabilistic Latent Semantic Analysis (pLSA) by introducing Dirichlet priors, which regularize topic distributions and prevent overfitting, making LDA generalize better to unseen documents. Compared to LSA, LDA provides a probabilistic framework that enhances topic coherence and word disambiguation. However, LDA is computationally expensive, as inference requires methods like Gibbs Sampling or Variational Inference, whereas LSA, being purely algebraic, is faster but lacks interpretability and does not capture document-level topic distributions probabilistically.

5.1 Pre-processing

In the preprocessing steps for Latent Dirichlet Allocation (LDA), several transformations were applied to clean and prepare the text data. First, punctuation marks such as commas, periods, exclamation points, and question marks were removed to ensure a more uniform text format. Then, all text was converted to lowercase to avoid duplication caused by capitalization differences. After this normalization, sentences were tokenized into words using Gensim's simple-preprocess function, which also removed additional punctuation. Next, stop words in English were filtered out, eliminating common words that do not contribute significantly to the meaning of the text. Following this cleaning process, a dictionary (id2word) was created, mapping each unique word to a numerical identifier. Finally, a corpus was generated, where each document is represented as a list of (word, frequency) pairs using the doc2bow method. This processed corpus serves as the foundation for training the LDA model, allowing the extraction of latent topics from the analyzed text data.

5.2 Analysis of some results of topics

The results of the LDA model indicate a diverse range of topics extracted from the text data. Several topics appear to be well-defined, such as Topic 2, which is related to Saudi Arabia and the Gulf region, and Topic 5, which strongly focuses on mathematical concepts like topology and vectors. Similarly, Topic 13 is centered around politics and elections, while Topic 31 clearly relates to football and sports clubs. However, some topics, such as Topic 8 and Topic 32, seem to be dominated by HTML or formatting-related symbols like "lt" and "gt," which could indicate noise in the dataset. The frequent presence of the token "quot" in multiple topics suggests that quotation marks or encoded characters may not have been fully preprocessed, potentially affecting topic coherence. Additionally, some topics, such as Topic 37, which includes both film-related and general words like "also" and "one," could benefit from further refinement to improve clarity.

5.3 Assigning Titles to Topics

To enhance the interpretability of the topics generated by the LDA model, a procedure was implemented to assign meaningful title to each topic. The process begins by extracting the most relevant keywords along with their respective weights from each topic. These keywords are then filtered to remove stop words, symbols, and other non-informative terms using a customized stopword list tailored to the dataset and domain. After this filtering step, the top three keywords with the highest weights are selected. To ensure semantic coherence and select the most representative term, a pre-trained word embeddings model (Word2Vec) is used to compute the semantic centroid (average vector) of these keywords. The keyword closest to this centroid is then chosen as the main title for the topic. If the embeddings are not available for some keywords or the semantic similarity cannot be reliably computed, the title is formed by concatenating the top keywords with slashes ("/"). This combined approach leverages both the statistical importance of keywords derived from the LDA model and their semantic relationships captured by word

embeddings. It guarantees that each topic is assigned a clear, concise, and semantically coherent title, improving the overall clarity and usability of the topic modeling results. **For example, consider Topic 0:**
The extracted keywords and their weights were:

- "art" (0.15),
- "book" (0.12) ,
- "philosophy" (0.10)
- "century" (0.08)
- "published" (0.05)

After filtering, the top three keywords "art," "book," and "philosophy" were selected. Using Word2Vec embeddings, the semantic centroid of these keywords was computed, and "art" was identified as the closest word to this centroid. Therefore, the title assigned to Topic 0 is **"Art"**, which effectively summarizes the cultural and philosophical theme of the topic. In contrast, some topics may contain more generic or less coherent keywords. For instance, Topic 1 included terms such as "especially," "important," and "however," which are common and do not define a clear theme. In such cases, the title reflects the nature of the topic but indicates a less specific or more general theme. This methodology ensures that every topic is assigned a meaningful title that balances keyword relevance and semantic coherence, facilitating a better understanding of the main themes emerging from the data.

For this procedure the results are contained in the following table:

Topic	**Title**
0	Art
1	People
2	Saudi
3	Ecosystem
4	Data
5	Math
6	Line
7	Tree
8	Small
9	Style
10	Davenport
11	World
12	Used
13	Party
14	France
15	War
16	Species
17	Language
18	Disease
19	Song

Topic	**Title**
20	China
21	Royal
22	Swedish
23	Mass
24	Radio
25	Aires
26	University
27	County
28	Born
29	Ireland
30	Road
31	Football
32	Span
33	Roman
34	Court
35	School
36	Zealand
37	Film
38	Tropical
39	Season

Table 2: Titles for each topic

5.4 Clustering Topics into Groups

After assigning titles to each topic, clustering was performed to identify groups of similar topics based on their keyword distributions. This step aims to highlight macro-themes and relationships among the topics extracted by the LDA model. For each topic, a numerical vector was extracted, where each component represents the weight of a specific keyword in that topic. The resulting matrix has topics as rows and unique keywords as columns, with the corresponding weights as entries. The matrix was then standardized to ensure comparability of weights and to prevent keywords with larger values from dominating the clustering.

5.4.1 Preliminary Choice of Number of Clusters with K Arbitrary

In this part, the number of clusters k was initially set arbitrarily to 7 in order to explore the grouping of topics. The results obtained with this fixed $k = 7$ show an uneven distribution of topics among clusters: some clusters contain only a few topics, while one cluster (Cluster 4) contains the majority of topics.

- **Cluster 4's large size** suggests that many topics share overlapping or broadly related keywords, making them semantically similar enough to be grouped together. This cluster likely represents a general or diverse thematic area that covers multiple subtopics with common vocabulary. For example, topics related to culture, society, media, and international affairs often share keywords such as "people," "world," "language," and "university," which causes them to cluster together.
- **Smaller clusters** (Clusters 0, 1, 2, 3, 5, and 6) represent more specialized or distinct themes with unique keyword distributions. These topics have more focused vocabularies that differ significantly from those in other clusters, causing them to form separate groups. For instance, Cluster 0 containing only Topic 14 ("France") could reflect a very specific geographical or historical theme distinct from the broader topics in Cluster 4.
- **Cluster 4** includes topics such as:
  - Topic 0: "Art"
  - Topic 1: "People"
  - Topic 2: "Saudi"
  - Topic 11: "World"
  - Topic 24: "Radio"

  These topics share common cultural and societal vocabulary, which explains their grouping in the same cluster despite covering different specific subjects.
- **Cluster 1**, with only Topic 29 ("Ireland"), may represent a narrowly defined geographical or historical topic with unique keywords not shared by other topics.
- **Cluster 5**, containing only Topic 9 ("Style"), likely focuses on a very specific theme related to fashion or design, distinct from broader clusters.

5.4.2 Number of clusters selected with Silhouette Analysis

Instead in this second part, the optimal number of clusters (k) was not chosen arbitrarily, but estimated using **silhouette analysis**, a technique that evaluates the quality of separation between clusters for different values of k. Silhouette scores were computed for k ranging from 2 to 10, with the following results:

- $k = 2$: 0.065
- $k = 3$: 0.043
- $k = 4$: 0.045
- $k = 5$: 0.023
- $k = 6$: 0.045
- $k = 7$: 0.008
- $k = 8$: -0.022
- $k = 9$: 0.038
- $k = 10$: 0.010

The highest silhouette score is obtained for $k = 2$ (0.065), although the absolute value is quite low. The silhouette coefficient ranges from -1 to $+1$, with values close to $+1$ indicating well-separated clusters, values near zero indicating overlapping clusters, and negative values suggesting possible misclassification. In this case, all scores are low, indicating that the clusters are not well separated and that topics tend to overlap or are quite distinct from each other. This can suggests that the original topics extracted by LDA do not share strong commonalities in their keyword distributions, and the dataset as a whole may be highly heterogeneous or composed of many topics,

5.4.3 Clustering Results and Interpretation

Applying clustering with $k = 2$, the topics were divided into two main groups:

- **Cluster 0**: Contains only Topic 20, characterized by keywords such as “china”, “chinese”, “hong”, “kong”, “japanese”, “japan”, “shanghai”, “korea”, and similar. This cluster represents a specific geographic and cultural theme related to East Asia.
- **Cluster 1**: Includes all other topics, covering a wide range of subjects: art, people, ecosystems, data, mathematics, transportation, science, sports, politics, music, media, school, natural events, and more.

Despite the limitations in this case, clustering in general allows us to:

- Identify macro-themes or outlier topics (such as Topic 20).
- Organize topics in a more structured way, facilitating analysis and presentation of results.
- Highlight the presence of transversal or strongly related topics.

6 BERT

A BERT, stands for Bidirectional Encoder Representations from Transformers, model is a language model based on the transformer architecture and is able to understand the context of words in sentences more deeply than traditional models. As previous models treated each words independently, BERT can capture the meaning of a word based on its context. Unlike traditional models that read text sequentially (left-to-right or right-to-left), BERT reads entire sequences of words at once, allowing it to capture context from both directions. It is pre-trained on a large corpus of text using two main tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). This pre-training helps BERT understand the nuances of language, such as semantics and syntax. BERT can enhance topic modeling by leveraging its deep understanding of language context. There are some advantages like:

- Contextual Understanding: BERT's ability to understand context makes it better at capturing nuanced topics compared to traditional methods.
- Versatility: BERT can be fine-tuned for specific domains or languages.
- Improved Accuracy: By leveraging deep learning, BERT can achieve higher accuracy in identifying coherent and meaningful topics.

On the other hand, BERT requires significant computational resources for training and inference, which can be a limitation for some applications, and sometimes interpreting what these topics mean can still be challenging and may require additional analysis.

6.1 Pre-processing

The pre-processing steps required by the BERT model are less sophisticated than previous models. First of all, BERT models require texts to be in a sentence form and not just single words, so it first checks if the actual text in the data is a list and joins the words if it's not. Following that, all the text is lowered, then if a website link is found it's removed to avoid problems in the next steps.

6.2 Model

The BERT model used in this case was based on an embedding model called "paraphrase-MiniLM-L6-v2" which specifies the pre-trained language model for generating the embeddings. This specific model is generally used for its good performances in semantic similarity tasks. Another variable set is the number of topics to find in the data, in this case it was set 40, just like the number of topics in the original dataset.

7 How is Topic Modeling evaluated?

Evaluating a topic modeling task involves both qualitative and quantitative approaches to assess the quality of the topics. Qualitative techniques include “eye-balling” models by examining top-n words and their distribution across topics and documents, while human judgment is often employed to assess topic interpretability and internal coherence. Quantitative methods can be grouped into intrinsic and extrinsic evaluation metrics. Intrinsic metrics focus on the model's semantic quality, primarily using perplexity, coherence, and diversity. Perplexity measures how well the model predicts unseen data: lower perplexity indicates a better predictive model. Coherence, on the other hand, evaluates the semantic similarity among top-ranked words within a topic; higher coherence suggests more interpretable and meaningful topics. Diversity measures the uniqueness of topics and aims to reduce redundancy metrics such as cosine similarity between topic vectors or assessing topic distribution across documents help quantify this. Extrinsic evaluation focuses on task-based performance, examining how well the topic model supports downstream tasks like classification or clustering. These three metrics together ensure that a topic model is not only statistically reliable but also semantically meaningful.

8 Comparing results

	LSA	pLSA	LDA	BERT
Perplexity	X	4.00	-12.50	3.68
Coherence	0.45	0.24	0.52	0.37
Diversity	0.35	0.98	0.79	0.71

Table 3: Results of each metrics for LSA, pLSA and LDA

In conclusion, the evaluation of LSA, pLSA, and LDA is based on three intrinsic measures: perplexity, coherence, and diversity. These metrics provide an internal assessment of topic model quality, focusing on predictive performance, semantic consistency, and topic distinctiveness.

Among probabilistic models, LDA achieves the lowest perplexity (-12.50), suggesting better generalization compared to pLSA (4.00). This advantage comes from LDA's Bayesian framework, which helps avoid overfitting, whereas pLSA lacks a mechanism to handle new documents effectively.

In terms of coherence,LDA again performs the best (0.52), indicating that its topics are more interpretable compared to pLSA (0.24) and LSA (0.45). pLSA's lower coherence may stem from its tendency to overfit, leading to less well-structured topics.

Regarding diversity, pLSA achieves the highest score (0.98), followed by LDA (0.79) and LSA (0.35). While greater diversity suggests more distinct topics, excessive diversity can reduce coherence. LDA strikes a good balance, generating topics that are both distinct and meaningful.

Overall, LDA emerges as the best-performing model, offering a strong trade-off between topic coherence and diversity while maintaining the lowest perplexity among probabilistic models. pLSA, despite generating diverse topics, struggles with coherence and overfitting, whereas LSA, as a non-probabilistic method, lacks the ability to model uncertainty and assess perplexity.

BERTopic was also evaluated using coherence and diversity. As a transformer-based model that does not rely on probabilistic word distributions,perplexity cannot be computed for BERTopic. However, a pseudo perplexity score was estimated by calculating the inverse of the average maximum topic probability assigned to each document. This approximation captures the model's confidence in topic assignment: higher maximum probabilities imply lower uncertainty and thus lower pseudo perplexity. Although it does not directly correspond to traditional perplexity, this metric serves as a qualitative indicator of how sharply the model differentiates topics across documents.

In terms of semantic coherence, BERTopic achieved a score of 0.37, computed via the average cosine similarity between the embeddings of the 10-top topic words. Although lower than LDA, this score reflects a reasonable degree of internal topic consistency. BERTopic also achived a good level of topic diversity (0.71), suggesting it is capable of generating and find distinct topics with moderate semantic coherence.

9 Text Classification

Following the topic modeling, a task of text classification is performed on the documents using the ground truth already available on the topic for each text. The objective of text classification is to predict to which a predefined set of categories a data item belongs to, without knowing it with absolute certainty. There are different types of classification, however in this case only the single-label multi-class classification is useful for this purpose.

SLMC hard classification is defined like:

$$h : D \to C$$

$$C = \{c_1, c_2, ..., c_n\},\ with\ n > 2$$

so each element of D belongs to exactly one class.

The general reason for applying text classification are to help organize the knowledge making it more effective in sciences and humanities, to enhance data retrieval and analysis and improve decision making processes. However, it can also be helpful in filtering out non relevant information like spam filters and unsuitable content detection systems do.

A text classification task can be performed either with an unsupervised or a supervised approach. In this case it was chosen to use a SLMC supervised approach to take advantage of information already available about the topic assigned to the texts.

The classifier doesn't work with actual words, so training documents are converted into vectors in a common vector space making it readable for the classifier. The pre-processing steps involve converting each word to lowercase, tokenize and removing common stop words, while a Bag of words (or CountVectorizer) is used first and then a Tf-idf later to tokenize the text to check eventually different results. The different models applied for the classification are very computationally expensive, so it was necessary to take an even smaller subset of 3.5% from the dataset, which is about 7000 documents, to get results in a reasonable time. The set of documents is split into 70% train and 30% test sets considering both each document text and the associated topic.

9.1 Models of text classification

The first models are applied on text vectorized using a bag of word model on both the train and test sets. The SVM, or Support Vector Machine, tries so find the separating surface that maximizes the margin from the training examples. Normally classification problems like this one are not linearly separable, however they can be mapped to a higher dimensional space to become linearly separable by using different kernels. A kernel is a similarity function defined as:

$$K(\vec{x_i}, \vec{x_j}) = \rho(x_i)\rho(x_j) \tag{1}$$

where $\rho(\cdot)$ is a mapping into a higher dimensional space. The SVM applied to the texts use four different kernels with a cross validation of 2 to check if the different functions can yield different results. After performing the classification, it can extract the kernel with the best score achieved. The different kernels used are:

$$linear : K(\vec{x_i}, \vec{x_j}) = x_i * x_j \tag{2}$$

$$RBF : K(\vec{x_i}, \vec{x_j}) = \exp\left(-\gamma\|\vec{x_i} - \vec{x_j}\|^2\right) \quad \gamma > 0 \tag{3}$$

$$Polynomial : K(\vec{x_i}, \vec{x_j}) = (\gamma\vec{x_i} \cdot \vec{x_j} + r)^d \quad \gamma > 0 \tag{4}$$

$$Sigmoid : K(\vec{x_i}, \vec{x_j}) = \tanh(\gamma\vec{x_i} \cdot \vec{x_j} + r) \tag{5}$$

At the end of the second cross validation iteration, the results show that the SVM with a linear kernel has the lowest accuracy of 0.8080 while the highest of 0.8497 was reached by choosing the RBF kernel.

The second model to be fitted to the data is a decision tree. The tree is built by splitting the source set, constituting the root node of the tree, into subsets which constitute the successor children. The splitting is based on a set of splitting rules based on classification features. The process is then repeated until a node has the same value of the target variable. The model is tested with different hyperparameters and with a cross validation of k = 3. For the parameters it was decided to control for the maximum depth of 2, 5, 10, 15 and 20, a minimum number of samples to be in the leaf nodes of 5, 10, 15 and 20 and either a gini or entropy criterion function to measure the quality of the splits. The result shows an accuracy of 0.8439 for a single iteration and 0.8494 for the cross validation accuracy by using the entropy as criterion, a maximum depth of 5 and 15 minimum samples per leaf. In both cases, the results are very similar to SVM, meaning that they can perform equally well.

The third model is a random forest classifier, which basically performs a classification task by creating a multitude of decision trees during the training phase, where the output is the class selected by most trees. The hyperparameters of the random forest are the same of the decision tree in the previous model, however there's an additional parameter to consider and it's the number of estimators of trees in the forest between 10, 20, 50, 100 and 150. The accuracy values for the single random forest and cross validated accuracy are 0.8491 and 0.8497 respectively. However, the best parametres identified are much different from the previous decision tree: the chosen criterion is gini, a max depth of 2, a minimum of 5 samples per leafs and with 10 trees in the forest.

Another model used for this classification task was the Naive Bayes classifier. The NB belongs to the family of probabilistic classifiers, which assumes that the information provided by each variable is unrelated from the information from the others. To predict the class y given the feature vector X is written as

$$P(x_1, x_2, ..., x_n|y) \tag{6}$$

however given the features independence it's possible to write

$$P(x_1|y) \cdot P(x_2|y) \cdot ... \cdot P(x_n|y) \tag{7}$$

Giving then the Bayes' theorem, it becomes

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y) \cdot \prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)...P(x_n)} \tag{8}$$

The denominator is constant for a given input, so the equation can be simplified to

$$P(y|x_1, x_2, ..., x_n) \propto P(y) \cdot \prod_{i=1}^{n} P(x_i|y) \tag{9}$$

To get the Naive Classifier each class y is computed and choose the class with the highest probability:

$$\hat{y} = \arg\max_{y} P(y) \cdot \prod_{i=1}^{n} P(x_i|y) \tag{10}$$

The model applied uses a Multinomial Naive Bayes, which is usually used for text classification tasks and features like count words. This model has an alpha parameter which is a smoothing parameter and that can to be tuned during the 3 fold cross validation between the values of 3,2 and 1.5. The function GridSearchCV tries every combination of hyperparameter defined earlier. The result shows that the best value of alpha is 1.5 while the accuracy and cross validation accuracy are 0.8497 and 0.8501 respectively.

The last model applied is the logistic regression classifier that tries to predict the probability that an input belongs to a specific class. The sigmoid function is used to convert the raw model output into a probability value between 0 and 1 forming an "S" shaped curve. A threshold value of 0.5 is commonly picked to choose the class label: if the output of the sigmoid function is equal or greater than 0.5 then the input is classified as Class 1, otherwise as Class 0. The model tries a combination of values for the penalty and solver (and the elasticnet penalty) hyperparameters. Once the best paramenters are found, the final model is trained on the entire training dataset and calculated the accuracy. There were some problems during the fitting phase, as for some parameters it failed to converge becuase it reached the limit on the number of iterations possibly by the fact that the resources are limited. However, the model finds that the best parameters are "elasticnet", "saga" and 0.5, where the first parameter is a combination of a Lasso (L1) and Ridge (L2) regularization techniques to reduce overfitting by penalizing large coefficients and the 0.5 represents the mix of L1 and L2 to use; "saga" is the algorithm to minimize the logistic regression loss function and it's best suited for problems with a large number of features. Also this model has a value of 0.8497 and 0.8492 for accuracy and cross validated accuracy respectively, like the previous models.

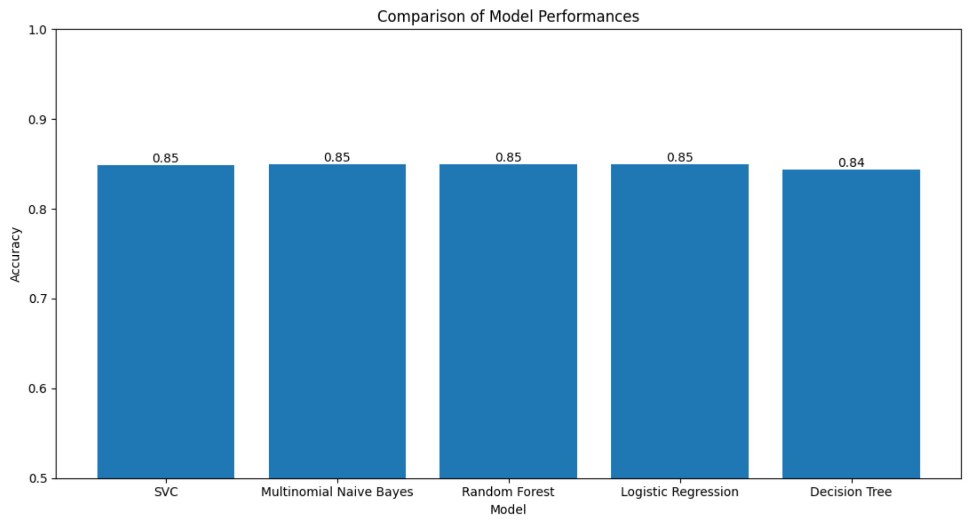

Figure 2: Accuracy values for classification models

Figure 2 shows the results of the classification models used to classify the texts which reach all the same accuracy value.

9.2 CountVectorizer and TF-IDF

The same models are performed again but this time the vectorization of the texts is done by using the TF-IDF instead of the CountVectorizer.

The SVM shows slightly different results from the SVM on countvectorizer: the best parameter for which the highest accuracy of 0.8503 is the linear kernel, followed by poly with an accuracy of 0.8501 and by RBF and sigmoid both at 0.8499.

The Decision tree shows almost the same result of 0.8487 and 0.8499 for the accuracy and cross validated accuracy respectively. This time the best parameter is the gini criterion, a maximum depth of 2 and 20 minimum number of samples per leaf. The random forest classifier shows the same results in accuracy and parameters as the decision tree, however the ideal number of trees in the forest remains 10.

The last two models also show quite similar results of the previous models, however it should be noted that for the alpha parameter in the multinomial naive bayes is 2 while the logistic regression set "none" and "newton-cg" as parameters: "none" means that no regularization is applied to data, while the "newton-cg" is a different algorithm used to minimize the logistic regression loss function.

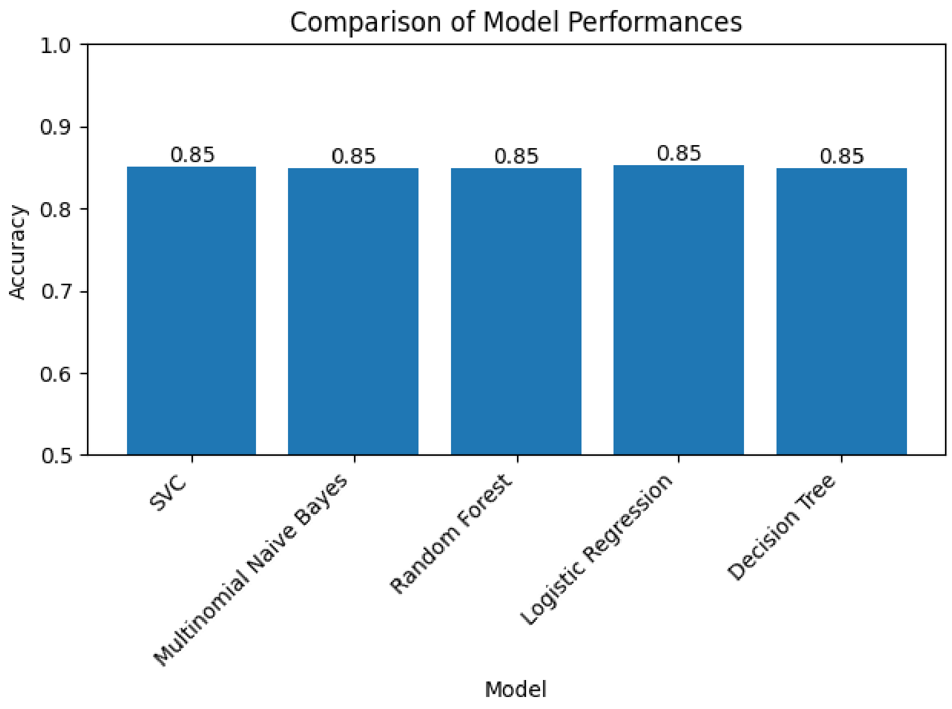

Figure 3: Accuracy values for classification models

Figure 3 shows that every classifier reach an accuracy of around 0.85, with a very subtle variation, perhaps the quantity of data fed as train examples is not sufficient to develop better classifiers.

10 Modeling Topic Distributions on the Simplex

10.1 Compositional Data and the Simplex

In many applications involving compositional data, the response variables are vectors of proportions that lie on the *simplex*:

$$\Delta^{K-1} = \left\{ \mathbf{p} = (p_1, \dots, p_K) \in \mathbb{R}^K \mid p_i > 0, \quad \sum_{i=1}^{K} p_i = 1 \right\}.$$

Such data arise naturally when modeling topic distributions in text documents, where each component represents the proportion of a given topic present in a document.

10.2 Dirichlet Regression

Dirichlet regression is a statistical modeling approach tailored to predict compositional vectors. It assumes that the response vector $\mathbf{Y}$ follows a Dirichlet distribution parameterized by a positive vector $\boldsymbol{\alpha}$:

$$\mathbf{Y} \sim \text{Dirichlet}(\boldsymbol{\alpha}), \quad \alpha_i = g_i(\mathbf{X}),$$

where $\mathbf{X}$ represents the predictor variables (e.g., text embeddings), and g_i are functions mapping the predictors to the positive domain. The Dirichlet distribution has the following density function:

$$f(\mathbf{y}; \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} y_i^{\alpha_i - 1}, \quad \mathbf{y} \in \Delta^{K-1},$$

where $B(\boldsymbol{\alpha})$ denotes the multivariate Beta function that ensures normalization.

10.3 Neural Network-Based Modeling

To model the dependence between the Dirichlet parameters and the input features, a neural network can be used to output a vector in the simplex. In our case, we used a feedforward network that ends with a `softmax` activation:

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{W}\mathbf{X} + \mathbf{b}),$$

which guarantees that the prediction $\hat{\mathbf{y}}$ lies within Δ^{K-1}.

10.4 Evaluation Metrics for Distributional Outputs

To evaluate the predicted distributions $\hat{\mathbf{y}}$, multiple complementary metrics were employed to capture different aspects of similarity and error between probability vectors:

- **Kullback-Leibler (KL) Divergence**, used as the training loss function, measures the information loss when $\hat{\mathbf{y}}$ is used to approximate the true distribution $\mathbf{y}$:

$$D_{\text{KL}}(\mathbf{y} \,||\, \hat{\mathbf{y}}) = \sum_i y_i \log \left(\frac{y_i}{\hat{y}_i} \right).$$

It is asymmetric and highly sensitive to errors in low-probability regions, which can dominate the divergence if $\hat{y}_i \to 0$ while $y_i > 0$. KL divergence is widely used in probabilistic learning due to its foundation in information theory.

- **Jensen-Shannon (JS) Divergence** is a smoothed and symmetric variant of KL divergence, defined as:

$$D_{\mathrm{JS}}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{2} D_{\mathrm{KL}}\left(\mathbf{y} \,\middle\|\, \frac{\mathbf{y}+\hat{\mathbf{y}}}{2}\right) + \frac{1}{2} D_{\mathrm{KL}}\left(\hat{\mathbf{y}} \,\middle\|\, \frac{\mathbf{y}+\hat{\mathbf{y}}}{2}\right).$$

Unlike KL, it is always finite and bounded between 0 and $\log 2$, making it more stable when comparing distributions with near-zero components.

- **Hilbert Projective Distance**:

$$d_H(\mathbf{y}, \hat{\mathbf{y}}) = \log\left(\max_i \frac{y_i}{\hat{y}_i}\right) - \log\left(\min_i \frac{y_i}{\hat{y}_i}\right),$$

is a scale-invariant metric often used in conic geometry and positive vector spaces. It captures proportional distortions across components, with particular sensitivity to the ratios between the largest and smallest discrepancies. This makes it effective in highlighting inconsistencies in the tails of the distribution, even when absolute errors are small.

- **Cosine Similarity**, defined as:

$$\cos(\mathbf{y}, \hat{\mathbf{y}}) = \frac{\mathbf{y} \cdot \hat{\mathbf{y}}}{\|\mathbf{y}\| \, \|\hat{\mathbf{y}}\|},$$

measures the angular similarity between vectors and is invariant to scale. When applied to probability vectors, it quantifies how well the predicted direction aligns with the true one in the simplex, focusing more on relative proportions than on magnitude differences.

- **Component-wise Mean Absolute Error (MAE)**:

$$\mathrm{MAE} = \frac{1}{n} \sum_i |y_i - \hat{y}_i|,$$

is a simple and interpretable measure of average deviation per component. It treats each topic equally and linearly penalizes absolute differences. However, it does not capture the geometry of probability distributions and may underrepresent small but structurally important errors.

10.5 Model Configuration

We embedded input texts using the `all-MiniLM-L6-v2` Sentence-BERT model. The final model was a neural network with two hidden layers and softmax output, trained with KL divergence. After tuning hyperparameters, the optimal configuration was:

- Hidden layers: [512, 256]
- Dropout: 0.3
- Learning rate: 0.0005
- Batch size: 32, Epochs: 20

10.6 Performance Metrics

Metric	Value
Validation KL Divergence	0.8399
Mean Hilbert Distance	22.012
Mean Jensen-Shannon Divergence	0.140
Mean Cosine Similarity	0.786

Table 4: Global Evaluation Metrics

Despite the moderate KL divergence and a high cosine similarity, the Hilbert distance appears unusually high. This metric is particularly sensitive to extreme ratios in probability values, and thus highlights the relative mismatches between predicted and true distributions, even when component-wise errors are small.

10.7 Topic-wise Error Analysis

Topic	MAE
People	0.069
Geography	0.046
History	0.046
Government	0.042
Culture	0.040
Society	0.040
Mass_media	0.038
Entertainment	0.037
Time	0.032
Humanities	0.027
Business	0.027
Technology	0.024
Sports	0.024
Politics	0.022
Academic_disciplines	0.021
Education	0.020
Universe	0.017
Economy	0.016
Nature	0.016
Science	0.016

Topic	MAE
Human_behavior	0.016
Engineering	0.016
Military	0.014
Entities	0.014
Health	0.013
Religion	0.013
Life	0.012
Language	0.011
Knowledge	0.011
Information	0.010
Law	0.009
Concepts	0.007
Energy	0.007
Food_and_drink	0.006
Internet	0.006
Mathematics	0.005
Philosophy	0.004
Communication	0.003
Ethics	0.001

Table 5: Mean Absolute Error for each topic

The topic-wise MAE results reveal significant variation across semantic domains, pointing to deeper patterns in the model's predictive behavior.

The model achieves good alignment in the overall shape of predicted distributions, as indicated by a mean cosine similarity of 0.786. Additionally, over 70% of the topics achieve a mean absolute error (MAE) below 0.02, demonstrating the model's ability to approximate dominant semantic content with reasonable accuracy.

Topics such as `People` (0.069), `Geography` (0.046), `History` (0.046), `Government` (0.042), and `Culture` (0.040) exhibit the highest MAE values. These categories are semantically broad and often co-occur across documents, which introduces ambiguity in the target distributions. For example, a historical article may touch on people, locations, political context, and cultural aspects, making the correct assignment of proportions more difficult. This leads to dispersion in the predicted distributions, especially across overlapping topics.

Low-error topics. Conversely, topics such as `Ethics` (0.001), `Communication` (0.003), `Philosophy` (0.004), and `Mathematics` (0.005) show significantly lower MAE. These topics tend to be linguistically self-contained and semantically distinct, leading to higher predictive accuracy. When a document is primarily about one of these topics, it rarely contains significant content from unrelated topics, making the model's task more deterministic. The embeddings (MiniLM) likely capture the distinctive vocabulary patterns of these topics more effectively.

Categories such as `Technology`, `Politics`, `Science`, and `Education` fall into a middle ground. Their vocabulary is partially domain-specific, which helps the model, but their frequent appearance across multiple domains introduces moderate prediction variance.

To better interpret these patterns, we organize the topics into three theoretical groups :

- High-Error Topics
$$MAE > 0.035$$
Broad, entangled categories with high co-occurrence rates. Their semantic ambiguity often leads to flattened predictions. Examples: `People`, `Culture`, `History`, `Society`.

- Moderate-Error Topics
$$0.015 < MAE < 0.035$$
Topics with moderate lexical clarity, sometimes domain-specific but often shared across document types. Examples: `Technology`, `Education`, `Politics`.

- Low-Error Topics
$$MAE < 0.015$$
Semantically isolated and linguistically distinct domains that exhibit clear separation in embedding space. Examples: `Mathematics`, `Philosophy`, `Ethics`, `Religion`.

Although many topics achieve low component-wise MAE, the Hilbert projective distance remains relatively high (22.012). This metric, unlike MAE or cosine similarity, is sensitive to relative proportions across all components. It disproportionately penalizes inconsistencies in low-probability components. In practice, this means:

- The model reliably captures dominant topics in each document (top-2 or top-3), as confirmed by cosine similarity;

- It often underestimates or misallocate the probability mass of minor topics, which are less influential in MAE but significantly affect divergence-based metrics;

- This leads to high KL and Hilbert values even when average prediction accuracy appears acceptable.

This discrepancy suggests that MAE is not sufficient to assess compositional fidelity in detail. Divergence metrics expose weaknesses in how the model handles the tail of the distribution—the set of low-probability but semantically relevant topics.

10.8 Conclusions

These findings confirm that neural models can effectively approximate topic mixtures on the simplex, especially when topics are distinct and semantically isolated. However, our current architectures tend to struggle with nuanced allocations among overlapping or minor topics. To improve performance in these areas, future work could explore:

- Topic-specific subnetworks or attention mechanisms to better disentangle overlapping semantic signals;
- Training regimes incorporating Dirichlet priors or uncertainty-aware calibration to improve predictions in low-mass regions;
- Regularization strategies that explicitly penalize overconfidence in ambiguous contexts.

Overall, the model captures the global structure of document-topic distributions with reasonable fidelity, but refinements are necessary to ensure robustness and interpretability in the long tail of semantic composition.

References

[1] Marco Viviani. *Slide - Topic Modeling*

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.*

[3] John Aitchison. *The Statistical Analysis of Compositional Data.* Journal of the Royal Statistical Society: Series B (Methodological), 44(2):139–177, 1982.

[4] Marco J. Maier. *DirichletReg: Dirichlet Regression for Compositional Data in R.* Research Report Series / Department of Statistics and Mathematics, WU Vienna University of Economics and Business, Report 125, 2014.

[5] Thomas M. Cover, Joy A. Thomas *Elements of Information Theory.* Wiley-Interscience, 2nd edition, 2006.

[6] Akash Srivastava, Charles Sutton. *Autoencoding Variational Inference For Topic Models.* International Conference on Learning Representations (ICLR), 2017.