Topic modeling methods comparison against LLM

Samuel Gonçalves
(supervisor: Fabrice Boissier, Marie Puren)

Technical Report n^o202501-techrep-goncalves, January 2025
revision bf5e93f

This report describes the development of a tool designed to compare different topic modeling methods. The tool has been applied to the method currently most widely used in this field, Latent Dirichlet Allocation (LDA), as well as to the experimental CREA method and to methods based on LLM vectorization via BERT. The main objectives of this project are: Firstly, to assess the relevance of different topic modeling methods based on the evaluation of the topics they generate. Secondly, to assess the impact of different database pre-processing on the results, and the match between this pre-processing and the topic modeling methods. Finally, to highlight and analyze the divergence between the different evaluations.

Ce rapport décrit le développement d'un outil conçu pour comparer différentes méthodes de modélisation de sujets. L'outil a été appliqué à la méthode actuellement la plus utilisée dans ce domaine, l'Allocation de Dirichlet Latente (LDA), ainsi qu'à la méthode expérimentale CREA et à des méthodes basées sur une vectorisation par LLM via BERT. Les objectifs principaux de ce projet sont : Premièrement, l'évaluation de la pertinence de différentes méthodes de modélisation de sujets à partir de l'évaluation des sujets qu'elles génèrent. Deuxièmement, l'évaluation de l'impact des différents pré-traitements de la base de donnée sur les résultats et de l'adéquation entre ces pré-traitements et les méthodes de modélisation de sujet. Enfin, la mise en lumière et l'analyse de la divergence entre les différentes évaluations.

Keywords
CREA method, analysis of language, topic modeling, tokenization, lemmatization, LLM, LDA, TreeTagger, RNNTagger, BERT

Laboratoire de Recherche de l'EPITA
14-16, rue Voltaire – FR-94276 Le Kremlin-Bicêtre CEDEX – France
Tél. +33 1 53 14 59 22 – Fax. +33 1 53 14 59 13
samuel.goncalves@epita.fr – http://www.lre.epita.fr/

Copying this document

Contents

Chapter 1

Introduction

Large Language Models (**LLMs**) have profoundly transformed our interaction with **knowledge** and the way we query **information**. Instead of relying on search engines like Google to display Wikipedia summaries for specific keywords, we now turn to conversational agents like **ChatGPT** for direct answers and insights. These advanced models are not only used for querying information but also for various other applications, such as **summarizing texts**, which presents numerous opportunities for enhancing **productivity** and understanding.

The primary objective of this research is to evaluate the current **state-of-the-art** capabilities of **LLMs**. To achieve this, an **experimental protocol** has been developed and implemented to compare the **performance** of LLMs with traditional **topic modeling** and **knowledge extraction** methods, such as the **CREA method** and **Latent Dirichlet Allocation** (**LDA**).

The current status of the protocol and the analysis of the associated results within this report compare **classic topic modeling methods** with methods using LLMs in a **vectorization** step before using conventional vector classification methods. The report includes ideas for the integration of methods **using LLMs directly** for topic modeling, but as these were unsuccessful, they are not included in the evaluation.

This study aims to provide a comprehensive and customizable **analysis** of the quality of results produced by LLMs in comparison to conventional techniques. By establishing this approach to evaluate these models, we seek to identify their **strengths** and **weaknesses** in summarizing various texts with various data **pre-processing** techniques, like **punctuation cleaning**, **lemmatization**, and **words to concepts transformation**.

This report will first provide a detailed account of the pre-processing methods employed to prepare the data, including the specific commands used within the developed protocol. Next, it will outline the implementation of these methods within the protocol. Lastly, the report will discuss the various **evaluation techniques** considered, explaining the rationale for selecting **coherence V** and its integration into the protocol.

A schematic representation of the protocol can be found in the appendix (Appendix .1).

Chapter 2

Context and State of the Art

To apply the protocol, we need a pre-established corpus of texts. Eight corpora made of French PHP courses will be used:

- The **test_scenario** corpus created to check the relevance of results before launching calculations on large corpora.
- The **php_slides** corpus made of texts taken from slides.
- The **php_texts** corpus made of texts taken from textual formats.
- The **php_java_texts** corpus made of texts taken from textual formats and java code.
- The **php_first_courses** corpus made of the nine first courses.
- The **php_java_first_courses** corpus made of the nine first courses and of the document with java code.
- The **php_courses** corpus made of all the courses except the document with java code.
- The **php_java_courses** corpus made of all the courses.

The used documents are the same that in F. Boissier's thesis Boissier (2022), included in the **scenarios** as follows:

Scenario/Corpus	Documents
test_scenario	C1-3
php_slides	C1-2,4,7-10,12-14,16,18
php_texts	C3,5-6,11,15,17,19
php_java_texts	C3,5-6,11,15,17,19 + CJA
php_first_courses	C1-9
php_java_first_courses	C1-9 + CJA
php_courses	C1-19
php_java_courses	C1-19 + CJA

The specification of what constitutes **topic modeling** needs to be taken into account: here, it is taken in the sense of **language analysis**. The words belonging to topic within our corpus will

therefore be defined by **clustering** Everitt et al. (2011) the texts.

The project therefore requires the application of pre-existing **clustering algorithms**, with a view to comparing the **accuracy** and **relevance** of their results (**??**). **Topic formation** using the **LDA method** Blei et al. (2003) and clustering using **conceptual similarity graphs** produced by the **CREA method** Boissier (2022) will be compared to methods based on **sentence vectorization** using BERT. The creation and analysis of the **evaluations** produced by the methods studied will serve as a result of the research question.

Application of these methods requires preparation of the **text corpus**. This can be divided into several stages.

- **Document tokenization** - In the context of **language analysis** and the **Natural Language Toolkit** Bird (2006), a **token** is a string of characters between two spaces, whether one word or several, linked by a character such as a hyphen or apostrophe. **Babelfy's** Moro et al. (2014) application to the **Latent Dirichlet Allocation** Ekinci and İlhan Omurca (2020) and CREA handles tokenization using the classic decomposition of **identifiers** in the **BabelNet semantic network**, with tokens representing **concepts** or **named entities**.
- **Token lemmatization** - **Lemmatization** is a lexical treatment of tokens that returns them to a **canonical neutral form**. **TreeTagger** Schmid (1994, 1995) seeks to return to the masculine singular and the indicative for verbs. **RNNTagger** Schmid (2019) uses the deep learning library **PyTorch** Imambi et al. (2021) to get higher tagging accuracy than TreeTagger. Babelfy's application of **Latent Dirichlet Allocation** does not require lemmatization in that **identifiers** in the **BabelNet network** are equivalent to lemmas: two words of common meaning will be represented by the same identifier.
- **FCA** - The **analysis of the formal concept** Wille (1982) of the documents determined after a calculation of the number of **occurrences**, a **normalization**, and the application of the **high or direct strategy** Jaffal (2019) in the creation of the formal concept Belohlavek (2008).
- **CREA** - Use of the **Galois lattice** Wille (1982) to calculate **CREA metrics** Boissier (2022), **mutual impact** and **conceptual similarity**.
- **Representation of documents as embeddings** - To enable classic vector classification methods McInnes et al. (2017); McInnes et al. (2018); McInnes et al. (2018) to be used on documents, the sentences making up these documents must be vectorized Hayat et al. (2024); Reimers and Gurevych (2019).

To evaluate the topics resulting from the methods studied, the use of **coherences** is necessary. **Coherences** are metrics for evaluating the **interpretability** of topics. The python topic modeling library **Gensim** Řehůřek and Sojka (2010) will be used to compute these coherences. Building on pre-existing **coherence comparison** work Röder et al. (2015), the choice was made to use the **V coherence** developed by **Michael Röder**, **Andreas Both** and **Alexander Hinneburg** Röder et al. (2015) and considered **state-of-the-art** in the field as the reference metric for this study. The choice of metric, based on the **Gensim implementation**, is however adaptable to other coherences, as will be done with the **UMASS coherence** Mimno et al. (2011).

Chapter 3

Topic modeling methods comparison protocol

3.1 Usage and tools

3.1.1 Organisation

The tool developed takes the form of a repository Github which will be made public at the end of the study. It is made up of several folders that embody the different parts of the protocol. While the *sources/* folder contains the **code developed** during the course of this research, the *input/* folder contains the **texts** whose **subject modeling** will be evaluated, at the various **pre-processing stages**. The *input/data/Raw/* folder is the **entry point** to the protocol on the **data side**, and by modifying the texts in this folder, the entire protocol can be adapted to the **evaluation of other documents**. Finally, the *output/* folder groups together the protocol's **outputs**, displaying their **subject models** in the *output/<scenario name>/data* folder or their **evaluation metrics** enabling **comparison of different methods and pre-treatments** in *output/<scenario name>/evaluations/*.

3.2 Pre-Processing

The **pre-processing phase** groups together all the **combinations of input text processing**, enabling a more detailed **comparison of the methods used**, and preparing the **transformation of texts into concepts** accompanied by the **metrics required for the CREA method**.

3.2.1 Tokenization

The **tokenization step** consists of **removing punctuation elements**, followed by an **aggregation of characters forming words**. In addition, **stopwords** are removed at this stage for certain text transformations. **Stopwords** are **common language words** that don't convey any meaning with regard to **subject modeling**, such as "*une*" or "*donc*" in French.

3.2.2 TreeTagger

Lemmatization via TreeTagger consists of returning **masculine singular**, or **indicative verbs** from the **text corpus**, after setting the language to **French**. In addition, **stopclasses** are removed at this stage for certain **text transformations**. **Stopclasses** are **classes of words** returned to accompany **lemmas** generated by **TreeTagger**, which a priori don't convey any **meaning** with regard to **subject modeling**, such as **articles** and **determiners**.

3.2.3 RNNTagger

RNNTagger uses **deep learning** with Pytorch to achieve **lemmatization**. For each **term**, the **original term**, the **word class** (also used for **error detection**) and the **lemmatized term** are preserved. Depending on the **presence of errors** or **special characters**, the recovered **lemma** changes.

3.2.4 Babelfy

Babelization is the name given to the application of Babelfy's API to the **lexical disambiguation** of a text's **semantic units**. Unlike the **semantic units** of **tokenization** and **lemmatization** in the **classical method**, where the **tokens** were words, via Babelfy these **semantic units** are **concepts** and **named entities**. This means that they can be **words**, but also **groups of words**, if these groups are seen as more **decisive in their meaning** than the words that make them up separately.

Figure 3.1: Illustration of Babelfy's application on a french sentence.

The **Babelfy API** can be accessed via various **Python libraries**, notably pybabelfy and BabelPy. As both are equivalent, pybabelfy is **arbitrarily chosen** in this project.

Requests to the **Babelfy API** are made in the form of a **URI**, limiting the number of **characters** that can be given within a request. The work therefore involved setting up an **algorithm** for calculating **text decomposition** into a list of **index pairs** (start and end) for each **block**, based on an **average text block size** and a **maximum deviation** from this value in the form of a **percentage**.

The **babel-token** is the name given to the equivalent of the **token** in the analogy between the **CREA method** and the **LDA**. It is the **basic brick** containing all the information for applying the **CREA method**. It breaks down into 7 parts.

- The **index of the first character** of the content within the text.
- The **index of the last character** of the content within the text.
- The **text content** between the two indexes.
- An **identifier** in the **Babelnet semantic network**, representing a **concept** or **named entity** corresponding to the content.
- The **"score"**, without qualifier.
- The **"global score"**.
- The **"consistency score"**.

These different **scores** provide additional information on the **relevance** of finding this **concept** or **named entity** within this text (Babelfy's method being **contextual**), the **relevance** of finding this **concept** or **named entity** written in this form (errors due to **OCR noise**, for example), and the **relevance** of finding this **concept** or **named entity** written in this location (**agreement or conjugation errors**).

Babelization can be seen as **tokenization** + **lemmatization** if only the **identifiers** are kept. Indeed, by definition of the **semantic network**, two similar words will have the same **identifier**, so they will be represented by the same **token**. Thus, via this representation (called "**equivalent text**"), **babelization** is acceptable as an **input block** in the **processing chain** leading to **LDA**. However, there is still a **problem** in **measuring the results** obtained. **Babelfy identifiers** do not provide a satisfactory way of knowing, as a human, whether a specific **term** should belong to a specific **cluster**. There are two ways of solving this **problem**.

- Each time a new **id** is added to the list of known **identifiers**, by keeping the **equivalent string** in memory so as to have a **reference word/example** enabling a human eye to understand the results.
- By referring to the Babelnet **semantic network** for indications concerning the **identifier**.

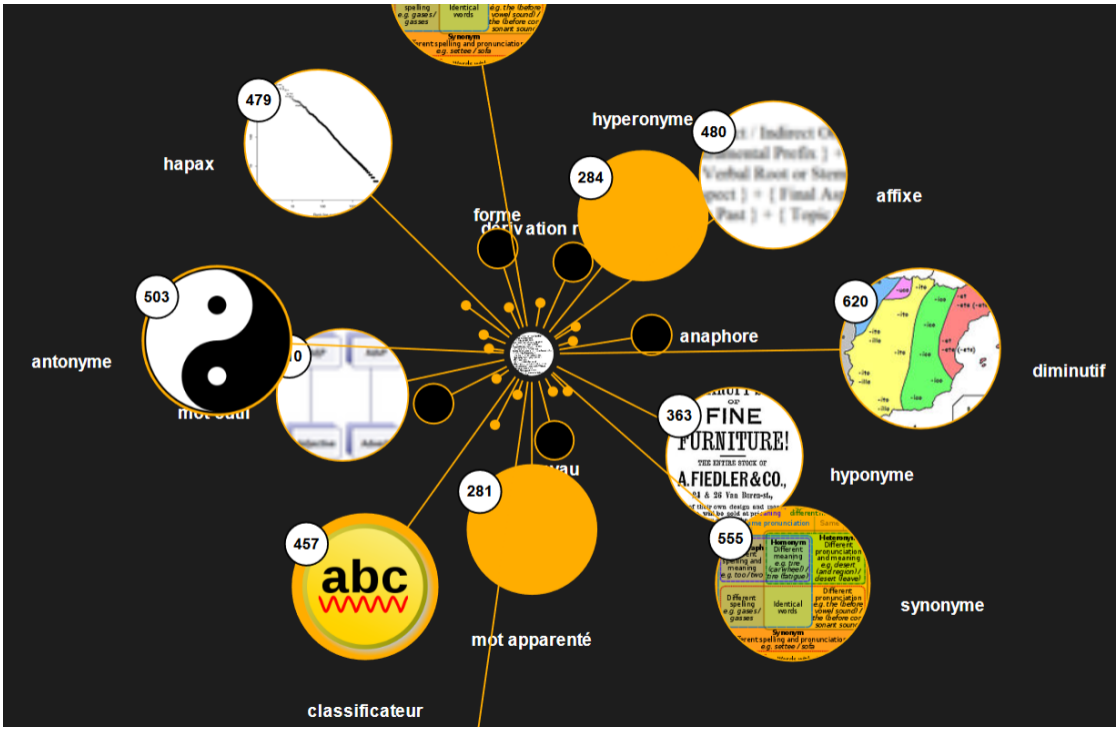

Figure 3.2: Illustration of the BabelNet **semantic network** of the french word "*mot*".

Keeping a list of **word-examples associated with ids** is the chosen solution because of its **simplicity**.

The call to **disambiguation** is long, and the texts requiring **multiple queries** add to the **time constraints** of the project.

3.3 Processing

The processing is used to apply existing data analysis methods (CREA Boissier (2022) using, in particular, Formal Concept Analysis Belohlavek (2008); Wille (1982) and Clustering Everitt et al. (2011); LDA Blei et al. (2003); BERT-based methods Hayat et al. (2024); Reimers and Gurevych (2019)).

3.3.1 CREA

Input		Output
babel-token	→	occurrences matrix
	→	frequency matrix
	→	formal context
	→	Galois lattice and CREA measures
conceptual similarity	→	topic models

Filtering and calculation of the number of occurrences

Input		Output
babel-token	→	occurrences matrix

Once pre-processing is complete, two operations are carried out in parallel: filtering and calculating the number of occurrences of each **concept** or **named entity**.

With regard to **filtering**, this involves using the **consistency score** calculated via babelization during pre-processing to retain only those **concepts** and **named entities** with a certain relevance in a text. This approach makes it possible to get rid of the most complex errors that could have appeared in the OCRization phase and changed the meaning of a **semantic unit**. These errors are mainly of two types.

- Errors on a short **semantic entity** (the shorter it is, the smaller the number of errors required to lose the meaning of the **concept** or **named entity**).
- Errors on a **semantic entity** that is difficult to access, such as a speech bubble or image (the less accessible, the more complex the OCRization and the greater the number of possible errors).

Empirically, the value of the **filtering threshold** is a **consistency score** of 0.05 Boissier (2022).

In parallel, each **concept** or **named entity** passing the filter increments a slot in the **occurrence matrix**, which counts the number of each **concept** or **named entity** for each document. The creation of this matrix is in fact an information selection phase, in which the position of the **concept** or **named entity** within the text is no longer preserved, unlike the equivalent text for the CREA method.

Normalization

Input		Output
occurrences matrix	→	frequency matrix

Once the **occurrence matrix** has been retrieved, its values and amplitude depend on the size and number of documents in the corpus studied, as well as the number of **concepts** and **named entities** after the filtering step.

To correct the situation and **normalize** the **occurrence matrices**, their row values are divided by the total row sum. Column normalizations are performed using the same procedure, preceded and followed by matrix transposition.

At the end of **normalization**, the matrix contains **frequencies** Jaffal (2019) between 0 and 1.

Formal context creation

Input		Output
frequency matrix	→	formal context

The aim of this step is to transform the **frequency matrix** into a **Boolean matrix**, named "formal context". This matrix indicates whether there is a **significant link** between each concept, each named entity, and each document in relation to the others.

The notion of "significant link" is intrinsic to the strategy chosen to create the **formal context**.

- The **direct strategy**
  - will act as an entry block in the chain leading to the creation of a **mutual impact graph**.
  - is not parameterized.
  - separates frequencies into two groups: null frequencies (set to false) and non-null frequencies (set to true).
- The **high strategy**
  - will serve as an input block in the chain leading to the creation of a **conceptual similarity graph**.
  - is parameterized by β, half the amplitude of the medium frequencies.
  - separates frequencies into three groups: low frequencies ($< 0.5 - \beta$) (set to false), medium frequencies ($\geq 0.5 - \beta$ and $\leq 0.5 + \beta$) (set to false), high frequencies ($> 0.5 + \beta$) (set to true).

Analysis of the formal context

The aim of this step is to create the **Galois lattice** from the **formal context** and calculate the CREA method's metrics of **mutual impact** and **conceptual similarity**.

Input		Output
formal context	→	Galois lattice and CREA measures

The vast majority of Python libraries dedicated to **Formal Concept Analysis** allow the creation of the Galois lattice. For this project, the concepts library and its **lattice method** were chosen.

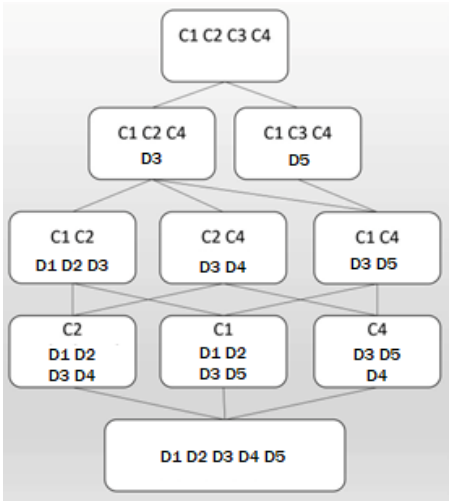

Figure 3.3: Illustration of a Galois lattice.

The **Galois lattice** is a graph that takes two types of information as input and whose nodes, called "formal concepts," contain a combination of these two types. In the chain leading to CREA topic modelisation, the input types are documents and concepts or named entities. Each node provides information on the content of the formal context. For example, the presence of a node containing document 1, document 2, concept 1, and concept 2 indicates that these two documents have a significant link with these two concepts. In this way, the strength of the link between one document and another can be determined by the number of significant concepts shared by these two documents. It's this kind of measurement that leads to the results of the CREA method.

- **Mutual impact**
  - is a matrix of documents and concepts or named entities, corresponding to the ratio of the number of nodes containing both the document and the concept or named entity to the number of nodes containing one, the other, or both.
  - is based on the Galois lattice calculated from the direct strategy.
- **Conceptual similarity**
  - is a square matrix of concepts or named entities, corresponding to the ratio of the number of nodes containing both the first and second concept or named entity to the number of nodes containing one, the other, or both. By definition, all values on the diagonal of the matrix are 1, because "A and A" = "A or A".
  - is based on the Galois lattice calculated from the high strategy.

Clusterization

Input		Output
conceptual similarity	→	topic models

In this last step, the **concept similarity** is used to form a **hierarchical clustering**, cut to the desired height to obtain a fixed number of non-overlapping clusters. Then, the **Galois lattice** is reused to compute the impact of each **topic** on each **document**.

3.3.2 LDA

Input		Output
token	→	random topic models
	→	topic models

Initialisation

Input		Output
token	→	random topic models

In this first step, every **word** within each **document** is allocated to a **topic** based on a **Dirichlet distribution** across a predefined set of topics.

This initial process constructs a foundational "topic model" revealing the **themes** found across documents and the **words** that characterize them. However, this model is speculative since it originates from **random assignment**.

Learning

Input		Output
random topic models	→	topic models

In this second step, the randomly initialized **topic model** is refined by iteratively updating the assignment of **topics** to **words** within each **document**. The topic that has the highest probability of generating a word in the document is assigned at this word. This approach assumes all other topics are correct except for the word currently being considered.

The operation is repeted until that the assignments stabilize. The distribution of **topics** within each **document** is determined by counting the occurrences of each topic assigned to the words in the document. Similarly, the words associated with each **topic** are identified by counting their occurrences across the entire **corpus**.

3.3.3 BERT-based original method

Input		Output
text	→	sentences
	→	embeddings
	→	reduced embeddings
	→	labelised reduced embeddings
	→	topic models

Corpus interpretation as sentences

Input		Output
text	→	sentences

The main feature of this method Hayat et al. (2024) is that the corpus is broken down into **sentences** rather than documents. In theory, this means **concatenating** the documents in the scenario under study and then separating the sentences that make up the resulting text. In practice, managing each text individually initially avoids repeating calculations between different scenarios.

Creation of sentence embeddings

Input		Output
sentences	→	embeddings

Once the corpus has been summarized into a **list of sentences**, a **LLM** can be applied to generate the **embedding** representing each sentence. This is achieved by using the **BERT** model ***paraphrase-MiniLM-L6-v6*** Reimers and Gurevych (2019).

Dimensionality reduction

Input		Output
embeddings	→	reduced embeddings

With the text transformed into a list of embeddings, it is possible to apply classic **vector classification** methods, once **dimensionality** has been **reduced**. For this purpose, **UMAP** McInnes et al. (2018); McInnes et al. (2018) is used in this method.

Clusterization

Input		Output
reduced embeddings	→	labelised reduced embeddings

Once the sentences have been **vectorized** and **reduced**, they can be classified using a **clustering** tool. In conjunction with UMAP, **HDBSCAN** McInnes et al. (2017) is used with this method.

Topic extraction

Input		Output
labelised reduced embeddings	→	topic models

Each **sentence cluster** must now be interpreted as a **topic**: to do this, we first retrieve all the words making up the sentences in the cluster, sorted by **occurrences**, before retaining only the k most frequent words.

3.3.4 BERT-based modified method

First version

Input		Output
text	→	**line**
	→	embeddings
	→	reduced embeddings
	→	**filtered** labelised reduced embeddings
	→	topic models

In practice, separating text into sentences can be difficult, especially for OCRized texts. To simplify this step, the method has been adapted to separate text into lines rather than sentences. In the case of HDBSCAN output, certain lines can no longer be assigned to a cluster, so these lines must be eliminated from the cluster creation process and the method adapted accordingly.

Second version

Input		Output
text	→	**babelfy notions**
	→	embeddings
	→	reduced embeddings
	→	filtered labelised reduced embeddings
	→	topic models

One attempt to improve the results was to apply pre-processing to the input data. Babelfy was used to reduce the text to a set of notions. As some notions can be on several lines, the separation of the text into lines loses its meaning and is replaced by a separation of the text into notions.

3.3.5 ChatGPT / Llama2

Input		Output
text	→	prompt
	→	topic models ?

Prompt creation

Input		Output
text	→	prompt

In this step, a **prompt** is generated from the original text, which is a string. The prompt should include several components:

- The **desired action** to be performed on the text (topic modeling).
- The **input text itself**.
- Optionally, the **preferred output format**.
- Optionally, a **description of any preprocessing** applied to the input text.
- Additional relevant information.

Generation

Input		Output
prompt	→	topic models ?

The response is generated by providing the prompt to a **large language model** (**LLM**). However, this generation step is currently blocked due to the **limitation on the prompt size**, exceeded by the input text. For **GPT**, the limitation was reached by the funds allocated to the project (the use of the **API** requires payment).

To evaluate the topics resulting from the methods studied, the use of **coherences** is necessary. **Coherences** are metrics for evaluating the **interpretability** of topics. The python topic modeling library **Gensim** Řehůřek and Sojka (2010) will be used to compute these coherences. Building on pre-existing **coherence comparison** work Röder et al. (2015), the choice was made to use the **V coherence** developed by **Michael Röder**, **Andreas Both** and **Alexander Hinneburg** Röder et al. (2015) and considered **state-of-the-art** in the field as the reference metric for this study. The choice of metric, based on the **Gensim implementation**, is however adaptable to other coherences, as will be done with the **UMASS coherence** Mimno et al. (2011).

3.4 Evaluation

Input		Output
topic models	→	score

To evaluate the topics resulting from the methods studied, the use of **coherences** is necessary. **Consistencies** are metrics for evaluating the **interpretability** of topics. For all scenarios, constitued of a pre-defined subset of texts from the input, for all options of the protocol, the resulting modelisation of topics is evaluated by all the selected coherences, which are in the actual protocol **V coherence** Röder et al. (2015) and **UMASS coherence** Mimno et al. (2011).

Using the **framework developed** in Röder et al. (2015), all **standard coherence** measures can be described using four parameters.

- A **probability estimation** (**P**) - Assesses the **probability** of **word co-occurrence**. This can be based on the **entire text** (**Boolean document**) or a specified **window of words** around the **target word** (**Boolean sliding window**).

- A **segmentation / subset of words** (**S**) - Defines the **words** being **compared** and their **reference words**. For instance, **UCI coherence** involves **comparisons between pairs of words** (**S = 1-1**), whereas **UMass coherence** compares a word with all **preceding words** (**S = 1-preceding**). The first part of a pair is the subset for which the support by the second part of the pair is determined.
- A **confirmation measure** (**M**) - Measures how the **word** being **compared** relates to the **reference word**. In the case of **NPMI coherence**, the confirmation measure is the Normalized **Pointwise Mutual Information** (**PMI**). It's the **core** of the metrics.
- An **aggregation method** ($\sum$) - **Combines** the **results** of each **word comparison**. This can be done using various methods such as the **mean**, **median**, or **minimum**.

3.4.1 V coherence

V coherence will be the main coherence used in this work. In particular, it will be used for the **final evaluation** of results and the production of **metrics**.

According to the **reference framework**, **V coherence** can be described by the parameters

$(P_{sw}(110), S^{one}_{set}, m_{cos(nlr,1)}, \sigma_a)$:

- $P_{sw}(110)$ - The boolean estimations are a class of probability estimation which reflects the link between two words by a group they share, without impact for the occurrence number. Here the boolean sliding window is used, so the link between words are made checking their common belonging to a subpart of a document. This subpart is a moving-over-the-document window with a specified size, here it's 110, where each step defines a new virtual document by copying the window content, applying after that a virtual "boolean document" estimation. The boolean sliding window captures proximity between word tokens.
- S^{one}_{set} - This segmentation method belongs to the "one-multiple" class, which compare pairs of subsets of words.. The S^{one}_{set} segmentation compares every single word to every subset. It's a derivation of the $Sone_{any}$ segmentation, which only compares a word with every **disjoint** subset.
- $m_{cos(nlr,1)}$ - The cosinus similarity measure belongs to the "indirect confirmation measures" class. It can be formalized by representing the word sets as vectors (a small constant ϵ is added to prevent logarithm of zero) :

  $$\overrightarrow{v}_{nlr,1}(W') = \left\{ \sum\nolimits_{w_i \in W} \left(NPMI(w_i, w_j)\right)^1 \right\}_{j=1,\dots,|W|}$$

  with

  $$NPMI(w_i, w_j) = \frac{\log\left(\frac{P(w_i,w_j)+\epsilon}{P(w_i)*P(w_j)}\right)}{-\log\left(P(w_i,w_j)+\epsilon\right)}$$

  and by applying a similarity measure on them :

  $$m_{cos(nlr,1)}(W', W^*) = s_{cos}(\overrightarrow{v}_{nlr,1}(W'), \overrightarrow{v}_{nlr,1}(W'))$$

with

$$s_{cos}(\overrightarrow{u}, \overrightarrow{v}) = \frac{\sum_{i=1}^{|W|} u_i \cdot v_i}{||\overrightarrow{u}||_2 * ||\overrightarrow{v}||_2}$$

- σ_a - The standard aggregation method σ_a combines the results of each word comparison by their average.

3.4.2 UMASS coherence

UMASS coherence will be the secondary coherence of this work. In particular, it will be used to check that the results produced by **V coherence** follow the **same trend**, in order to certify the result in some way.
According to the **reference framework**, **UMASS coherence** can be described by the parameters $(P_{bd}, S^{one}_{pre}, m_{lc}, \sigma_a)$:

- P_{bd} - The boolean estimations are a class of probability estimation which reflects the link between two words by a group they share, without impact for the occurrence number. Here, the boolean document is used, so the link between words are made checking their common belonging to a document.
- S^{one}_{pre} - This segmentation method belongs to the "one-one" class, which compare pairs of single words: every single word is paired with every other single word. The S^{one}_{pre} segmentation compare a word only to the preceding words.
- m_{lc} - The log-conditional-probability measure belongs to the "direct confirmation measures" class. It used the following measure (a small constant ϵ is added to prevent logarithm of zero) :

$$m_{lc} = \log\left(\frac{P(W_1, W_2) + \epsilon}{P(W_2)}\right)$$

- σ_a - The standard aggregation method σ_a combines the results of each word comparison by their average.

3.5 Results

In theory, coherences can be interpreted as follows:

- **Coherence V** - Ranges from 0 to 1, with higher values indicating more coherent generated subjects.
- **Coherence Umass** - Ranges from $-\infty$ to 0, with values closer to 0 indicating more coherent generated subjects.

In practice, the correlation between these two metrics is almost consistently negative in the "CREA vs LDA" evaluation. A higher score according to V coherence usually corresponds to a lower score according to Umass coherence.

One possibility to explain this result could be the size of the subjects, on the one hand, and their repetitiveness, on the other. In the majority of scenarios, the word php appears in all the documents in the corpus. It is therefore present in all the topics generated by LDA, which leads

to lower scores via V consistency. On the other hand, the CREA method will classify each notion in the corpus in one of the topics, creating gigantic topics, with possibly low relevance of words two by two within these topics, leading to a very sharp drop in scores via UMASS consistency. At this stage, therefore, the results obtained do not allow us to draw any conclusions concerning the comparison between the different methods evaluated.

Despite this, the results of the evaluation can be found in the appendix (Appendix .2).

Chapter 4

Discussion and Conclusion

4.1 Discussion

4.1.1 Related Work

The work carried out within this project is mainly based on the *Technical Report no202406-techrep-goncalves, June 2024, revision bf5e93f*, my last year's work, itself based on Fabrice Boissier's thesis on the CREA method Boissier (2022). It is in fact an alteration of the method to adapt it to a generalized protocol producing scores from an input set of texts. It is also on this thesis that the corpora carried out is based.

4.1.2 Future Work

Future work on this project could involve overcoming the limitations encountered, and increasing the scope of the project by including new methods or new documents in the database.

With regard to the use of LLMs, the main objective is to solve the problem of the size limit of prompts. One way of doing this is to summarize texts beforehand by LLM in order to stay within the prompt size limit, "summarize" being used here in the sense of reducing the original text to a short one. Alternatively, we could work with two LLMs, the first of which would transform each sentence or paragraph into embeddings along the lines of the BERT-based method, and the second LLM would then interpret these vectors to obtain the subject model. A fine-tuning of the last layers of Llama2 should make it possible to achieve this result. With regard to GPT, an allocation of funds to the project would enable the tool to be used and included in the comparison.

With regard to improving the BERT-based method, the main objective would be to reduce noise by managing stopwords. One way of doing this would be to mark the stopwords detected in advance so as to be able to eliminate the clusters containing them, this method assuming that the stopwords would then be grouped together in these clusters. Another method would be to simply get rid of them in advance via pre-processing other than Babelfy. Finally, it should be possible to train a model so that vectorization results in a cluster grouping all the noise during classification.

Work could also be carried out on the text corpus, applying the protocol to other databases to study in greater detail the evolution of results in the presence of noise, or on a multi-thematic corpus.

The subjects generated at the output of the CREA method could also be adapted to be more faithful to the expectations of UMASS consistency. One way of doing this could be, as with the BERT-based method, to filter out from the subject words only the k words with the most occurrences in the corpus.

4.2 Conclusion

This report introduced a new protocol that enables the application of automatic language processing techniques to a customizable set of texts.
The study led to the development of versatile language analysis tools designed to be applied to many research questions in Human, Social and Computer Sciences.

The introduction of a robust framework for language analysis makes it possible to conduct a comprehensive and customizable analysis of the quality of results produced by LLMs in comparison to conventional techniques. This framework facilitates a detailed comparison between the outcomes of traditional methods and those generated by large language models (LLM), enhancing the evaluation process and providing deeper insights into their respective strengths and weaknesses.

Chapter 5

Bibliography

Belohlavek, R. (2008). Introduction to formal concept analysis. *Palacky University, Department of Computer Science, Olomouc*, 47. (pages 6 and 10)

Bird, S. (2006). Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72. (page 6)

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022. (pages 6 and 10)

Boissier, F. (2022). *CREA: méthode d'analyse, d'adaptation et de réutilisation des processus à forte intensité de connaissance: cas d'utilisation dans l'enseignement supérieur en informatique*. PhD Thesis, Université Panthéon-Sorbonne-Paris I. (pages 5, 6, 10, 11, and 21)

Ekinci, E. and İlhan Omurca, S. (2020). Concept-lda: Incorporating babelfy into lda for aspect extraction. *Journal of Information Science*, 46(3):406–418. (page 6)

Everitt, B., Landau, S., Leese, M., and Stahl, D. (2011). *Cluster analysis*. Wiley, 5th edition. (pages 6 and 10)

Hayat, F., Shatnawi, S., and Haig, E. (2024). Students' experiences and challenges during the covid-19 pandemic: A multi-method exploration. In *European Conference on Technology Enhanced Learning*, pages 152–167. Springer. (pages 6, 10, and 15)

Imambi, S., Prakash, K. B., and Kanagachidambaresan, G. (2021). Pytorch. *Programming with TensorFlow: Solution for Edge Computing Applications*, pages 87–104. (page 6)

Jaffal, A. (2019). *Aide à l'utilisation et à l'exploitation de l'analyse de concepts formels pour des non-spécialistes de l'analyse des données*. PhD thesis, Université Panthéon-Sorbonne-Paris I. (pages 6 and 12)

McInnes, L., Healy, J., and Astels, S. (2017). hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205. (pages 6 and 15)

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*. (pages 6 and 15)

McInnes, L., Healy, J., Saul, N., and Grossberger, L. (2018). Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861. (pages 6 and 15)

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., and McCallum, A. (2011). Optimizing semantic coherence in topic models. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, pages 262–272. (pages 6 and 17)

Moro, A., Raganato, A., and Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244. (page 6)

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en. (pages 6 and 17)

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. (pages 6, 10, and 15)

Röder, M., Both, A., and Hinneburg, A. (2015). Exploring the Space of Topic Coherence Measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 399–408, Shanghai China. ACM. (pages 6 and 17)

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *New methods in language processing*, page 154. (page 6)

Schmid, H. (1995). Improvements in part-of-speech tagging with an application to german. In *In Proceedings of the ACL SIGDAT-Workshop*, pages 47–50. (page 6)

Schmid, H. (2019). Deep learning-based morphological taggers and lemmatizers for annotating historical texts. In *Proceedings of the 3rd international conference on digital access to textual cultural heritage*, pages 133–137. (page 6)

Wille, R. (1982). Restructuring lattice theory: An approach based on hierarchies of concepts. In Rival, I., editor, *Ordered Sets*, volume 83 of *NATO Advanced Study Institutes Series*, pages 445–470. Springer Netherlands. (pages 6 and 10)

Thank you to those who reviewed and corrected this report prior to submission. Their valuable contributions have improved the quality and clarity of this research.

.1 Protocol overview

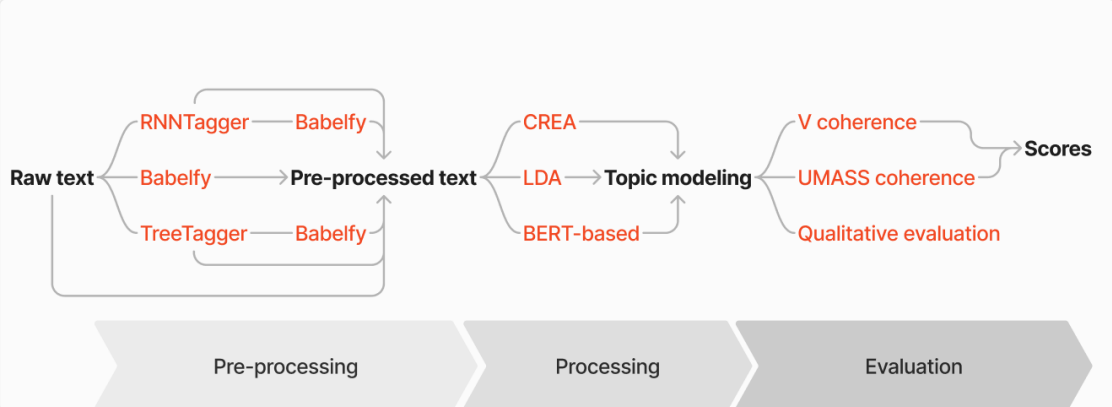

Figure 1: Overview of the implemented steps and options of the topic modeling protocol

.2 Evaluations for the test_scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.368	0.335	0.423	0.373	0.408	0.358
CREA		0.555			**0.674**	0.432
BERT_01p	0.486	0.54				
BERT_05p	0.439	0.507				
BERT_1p	0.497	0.45				
BERT_10	0.478	0.537				
BERT_20	0.456	0.529				

UMASS coherence	Raw	Babel	Tree	RNN	Tree +Babel	**RNN +Babel**
LDA	0.906	0.906	0.88	0.909	0.873	**0.951**
CREA		0.441			nan	0.517
BERT_01p	nan	nan				
BERT_05p	nan	nan				
BERT_1p	nan	nan				
BERT_10	nan	nan				
BERT_20	nan	nan				

.3 Evaluations for the php_texts scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.459	0.347	0.422	0.458	0.429	0.302
CREA		0.643			**0.653**	0.484
BERT_01p	0.408	0.477				
BERT_05p	0.43	0.437				
BERT_1p	0.533	0.426				
BERT_10	0.435	0.524				
BERT_20	0.414	0.494				

UMASS coherence	**Raw**	Babel	**Tree**	RNN	Tree +Babel	RNN +Babel
LDA	**0.988**	0.986	**0.988**	0.982	0.994	0.847
CREA		0.029			0.033	0.169
BERT_01p	0.061	nan				
BERT_05p	0.078	nan				
BERT_1p	0.542	nan				
BERT_10	nan	nan				
BERT_20	0.063	nan				

.4 Evaluations for the php_java_texts scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.441	0.363	0.429	0.434	0.467	0.311
CREA		0.596			**0.61**	0.483
BERT_01p	0.413	0.477				
BERT_05p	0.34	0.462				
BERT_1p	0.342	0.465				
BERT_10	0.437	0.533				
BERT_20	0.416	0.525				

UMASS coherence	**Raw**	Babel	Tree	RNN	Tree +Babel	RNN +Babel
LDA	**0.988**	0.987	0.966	0.994	0.979	0.831
CREA		0.152			0.043	0.283
BERT_01p	0.073	nan				
BERT_05p	0.032	nan				
BERT_1p	0.072	nan				
BERT_10	nan	nan				
BERT_20	nan	nan				

.5 Evaluations for the php_first_courses scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.469	0.381	0.406	0.519	0.425	0.289
CREA		0.59			**0.591**	0.508
BERT_01p	0.452	0.493				
BERT_05p	0.47	0.454				
BERT_1p	0.453	0.39				
BERT_10	0.453	0.532				
BERT_20	0.452	0.525				

UMASS coherence	Raw	Babel	Tree	**RNN**	Tree +Babel	RNN +Babel
LDA	0.887	0.928	0.695	**0.942**	0.909	0.864
CREA		0.045			0.026	0.123
BERT_01p	nan	nan				
BERT_05p	0.012	nan				
BERT_1p	0.037	0.013				
BERT_10	nan	nan				
BERT_20	nan	nan				

.6 Evaluations for the php_java_first_courses scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.471	0.371	0.38	0.433	0.439	0.288
CREA		0.594			**0.629**	0.466
BERT_01p	0.455	0.487				
BERT_05p	0.475	0.477				
BERT_1p	0.447	0.373				
BERT_10	0.451	0.525				
BERT_20	0.463	0.521				

UMASS coherence	**Raw**	Babel	Tree	RNN	Tree +Babel	RNN +Babel
LDA	**0.932**	0.905	0.847	0.902	0.859	0.732
CREA		0.055			0.005	0.092
BERT_01p	nan	nan				
BERT_05p	0.039	0.017				
BERT_1p	0.047	0.025				
BERT_10	nan	nan				
BERT_20	nan	nan				

.7 Evaluations for the php_courses scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.465	0.368	0.436	0.445	0.464	0.345
CREA		0.629			**0.631**	0.525
BERT_01p	0.424	0.481				
BERT_05p	0.387	0.47				
BERT_1p	0.385	0.474				
BERT_10	0.437	0.546				
BERT_20	0.425	0.523				

UMASS coherence	Raw	**Babel**	Tree	RNN	Tree +Babel	RNN +Babel
LDA	0.947	**0.969**	0.895	0.959	0.943	0.863
CREA		0.011			0.003	0.039
BERT_01p	nan	nan				
BERT_05p	0.014	nan				
BERT_1p	0.004	nan				
BERT_10	nan	nan				
BERT_20	nan	nan				

.8 Evaluations for the php_java_courses scenario

V coherence	Raw	Babel	Tree	RNN	**Tree +Babel**	RNN +Babel
LDA	0.471	0.383	0.43	0.472	0.476	0.332
CREA		0.621			**0.644**	0.551
BERT_01p	0.424	0.447				
BERT_05p	0.429	0.447				
BERT_1p	0.37	0.51				
BERT_10	0.435	0.532				
BERT_20	0.425	0.513				

UMASS coherence	Raw	**Babel**	Tree	RNN	Tree +Babel	RNN +Babel
LDA	0.94	**0.958**	0.889	0.939	0.914	0.872
CREA		0.018			0.002	0.018
BERT_01p	nan	nan				
BERT_05p	0.007	nan				
BERT_1p	0.027	nan				
BERT_10	nan	nan				
BERT_20	nan	nan				

.9 Evaluations for the php_slides scenario

V coherence	Raw	**Babel**	Tree	RNN	Tree +Babel	RNN +Babel
LDA	0.403	0.392	0.408	0.439	0.485	0.341
CREA		**0.646**			0.627	0.598
BERT_01p	0.442	0.513				
BERT_05p	0.419	0.523				
BERT_1p	0.346	0.523				
BERT_10	0.455	0.536				
BERT_20	0.444	0.533				

UMASS coherence	Raw	Babel	Tree	**RNN**	Tree +Babel	RNN +Babel
LDA	0.89	0.914	0.791	**0.947**	0.912	0.785
CREA		0.017			0.003	0.06
BERT_01p	nan	nan				
BERT_05p	nan	nan				
BERT_1p	0.007	nan				
BERT_10	nan	nan				
BERT_20	nan	nan				