Topic modeling methods comparison against LLM

Samuel Gonçalves
(supervisor: Fabrice Boissier, Marie Puren)

Technical Report n^o202406-techrep-goncalves, June 2024
revision bf5e93f

This report describes the development of a tool designed for comparing different topic modeling methods. The tool has been applied to the current leading method in the field, Latent Dirichlet Allocation (LDA), as well as to the experimental CREA method and a large language model (LLM), Llama2. The primary objective of this project is to assess how various data pre-processing techniques impact the relevance and accuracy of the topic modeling results. Specifically, the study focuses on the effects of punctuation cleaning, lemmatization and words to concepts transformation. Additionally, the project aims to evaluate how different types of prompts influence the performance and output of large language models.

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 latent (LDA), ainsi qu'à la méthode expérimentale CREA et à un grand modèle de langage (LLM), Llama2. L'objectif principal de ce projet est d'évaluer l'impact des différentes techniques de prétraitement des données sur la pertinence et la précision des résultats de la modélisation thématique. Plus précisément, l'étude se concentre sur les effets du nettoyage de la ponctuation, de la lemmatisation et de la transformation des mots en concepts. En outre, le projet vise à évaluer comment différents types d'entrées utilisateur influencent les performances et les résultats des grands modèles de langage.

Keywords
CREA method, analysis of language, topic modeling, tokenization, lemmatization, LLM, LDA, TreeTagger, RNNTagger, Llama2

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**).

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.

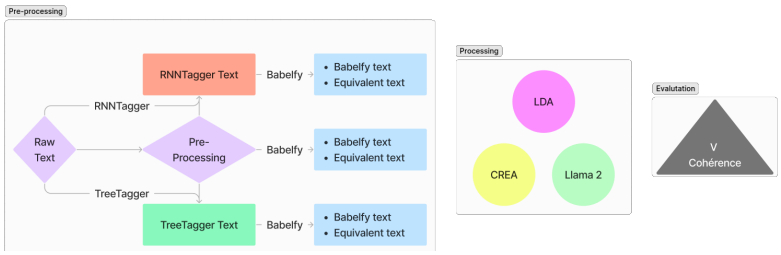

Figure 1.1: Overview of the implemented steps and options of the topic modeling protocol

Chapter 2

Context and State of the Art

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

- The **scenario_slides** corpus made of texts taken from slides.
- The **scenario_texts** corpus made of texts taken from textual formats.
- The **scenario_0** corpus made of the union of the two others.

The used documents are the same that in F. Boissier's thesis Boissier (2022):

Scenario/Corpus	Documents
scenario_slides	C1; C2; C4; C7; C8; C9; C12; C13; C14; C16; C18
scenario_texts	C3; C5; C6; C11; C15; C17; C19

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 (4). **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. 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**.

To evaluate the topics resulting from the methods studied, the use of **coherences** is necessary. **Consistencies** 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**.

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.

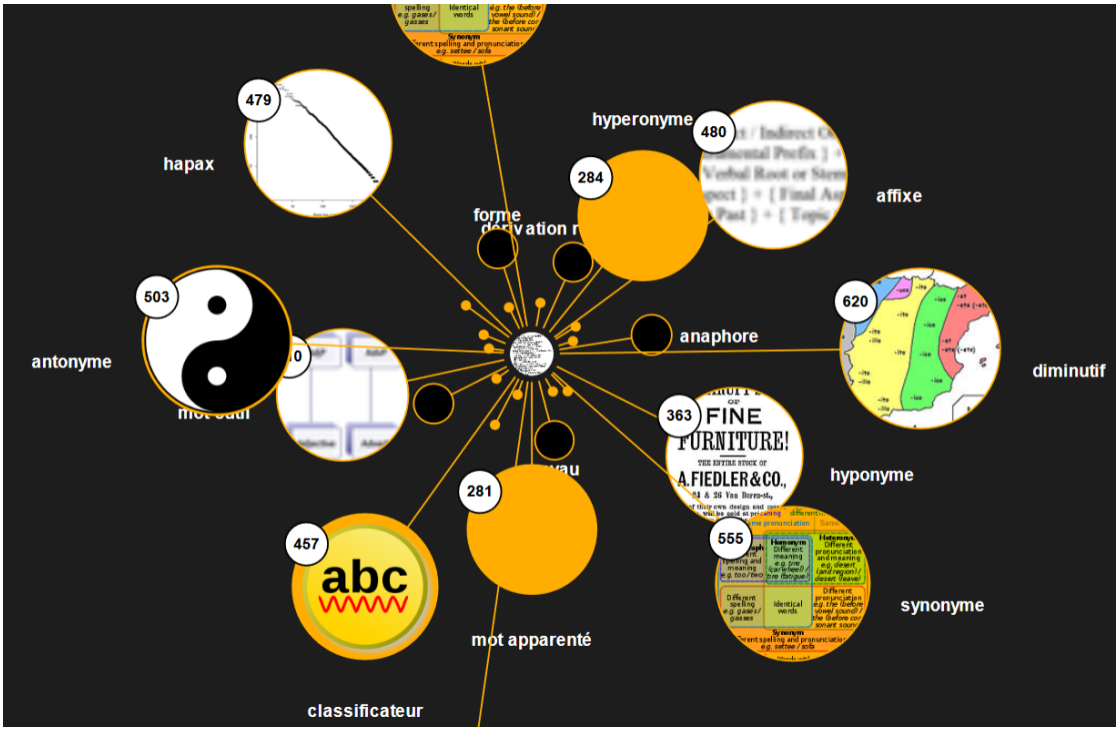

Figure 3.2: Illustration of the BabelNet **semantic network** of the french word "*mot*".

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), or LDA Blei et al. (2003)).

3.3.1 CREA

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 - β) (set to false), medium frequencies ($\geq$ 0.5 - β and $\leq$ 0.5 + β) (set to false), high frequencies (> 0.5 + β) (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	$\rightarrow$	**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.

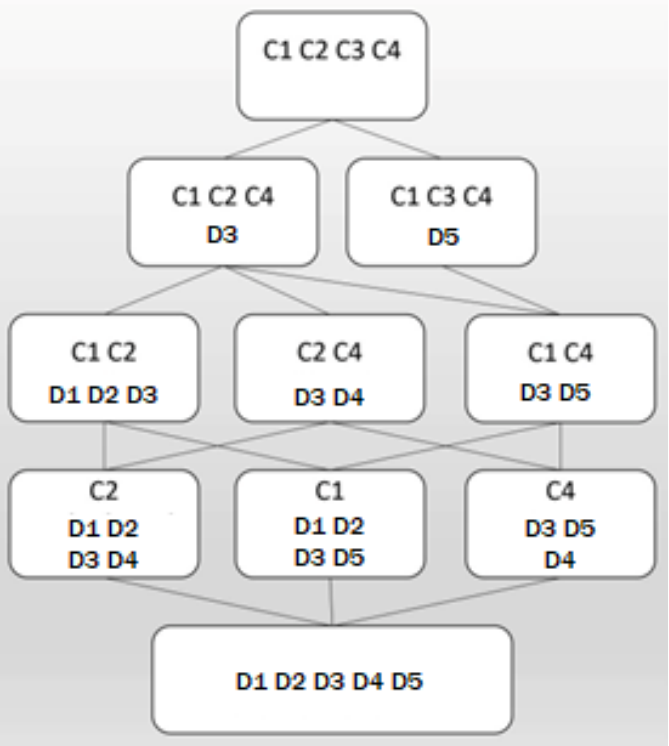

Figure 3.3: Illustration of a Galois lattice.

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

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 LLM

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 request size**.

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 S^{one}_{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) :

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

  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.

Chapter 4

Measures

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.

Therefore, at present, the results do not allow to draw definitive conclusions. It is crucial to "evaluate the evaluators" to ensure the reliability of the results before they can be fully analyzed.

The scores produced by Umass coherence will be analyzed in comparison with their equivalent produced by V coherence, the idea being to use this coherence as a metric for controlling V coherence results.

4.1 Pre-processing for LDA

4.1.1 V coherence

The evaluation of LDA results using V coherence shows a consistent trend across all studied scenarios, specifically regarding the impact of pre-processing on the score.

Two distinct groups of pre-processings can be identified: one lower group with a score around 0.35, which includes pre-processings by Babelfy and RNNTagger+Babelfy, and one upper group with a score around 0.45, which includes pre-processings by RNNTagger, TreeTagger, TreeTagger+Babelfy, and simple pre-processing via NLTK tools only. Interestingly, within the same **all slides** scenario (A.3), two Babelfy-based pre-processings yielded the best and worst LDA evaluation scores: TreeTagger+Babelfy (0.51) and RNNTagger+Babelfy (0.30), respectively.

Detailed results can be found in the appendix (Appendix A).

4.1.2 UMASS coherence

The evaluation of LDA results using Umass coherence shows a less clear-cut trend than for coherence V, although it does exist.

For the top group of V coherence results, preprocessing via RNNTagger ranks right at the top, always scoring at or very close to the top. TreeTagger and TreeTagger+Babelfy, on the other hand, give rather mediocre results, with the latter even achieving the worst score on the **all files** scenario (A.5). Simple pre-processing using only NLTK tools scores well, but collapses in the **all slides** scenario (A.7), where it comes in last place.

For the bottom group of V coherence results, preprocessing via Babelfy achieves excellent results, contrary to what appeared with V coherence. On the other hand, RNNTagger+Babelfy preprocessing confirms its place at the bottom, systematically obtaining a poor result, which is particularly glaring on the **all texts** scenario (A.8) where it is the only one in the bottom.

Detailed results can be found in the appendix (Appendix A).

4.2 Pre-processing for CREA

The CREA method requires specific metrics for subject modeling. As these metrics are only generated by Babelfy, only pre-processing that begins with a babelization phase could result in a score and therefore an analysis.

4.2.1 V coherence

The results of the CREA method's V Coherence preprocessing evaluation are very interesting, because they illustrate how a single document can totally change the trend. The Babelfy and Babelfy+TreeTagger pre-treatments achieve fairly similar results, while Babelfy+RNNTagger stands out.

When **all files** are present, including the *CJA.txt* document containing a lot of noise for topic modeling (being made up of a large amount of code), Babelfy+RNNTagger comes out on top (B.1). Remove this file, however, and the trend reverses: in all other scenarios, the other two preprocessings come out on top.

Detailed results can be found in the appendix (Appendix B).

4.2.2 UMASS coherence

The results of evaluating the CREA method via Umass coherence give opposite results in every respect. This trend seems to be confirmed by the joint evaluation of the CREA and LDA methods: it seems that the subject scores generated by CREA are highly polarized and controversial.

Here, then, we find the exact opposite phenomenon to the V-coherence analysis. This time, when **all files** are present, including the *CJA.txt* document, Babelfy+RNNTagger comes out on bottom (B.1). The trend reverses in all other scenarios where it comes out on top.

Detailed results can be found in the appendix (Appendix B).

4.3 CREA vs LDA results

Again, the CREA method requires specific metrics for subject modeling. As these metrics are only generated by Babelfy, only pre-processing that begins with a babelization phase could result in a score and therefore an analysis. And since a comparative evaluation requires something to compare, the results obtained by LDA for pre-processing not using Babelfy will not be discussed in this section.

It's also important to note that the topics generated by LDA and CREA are not of the same order: while the former generates overlapping topics, i.e. where a word can appear in several topics at once, this is not the case for the latter. This difference is particularly apparent with the term "*PHP*", the main subject of the corpus present in almost all the topics generated by LDA. This difference may partly explain the divergence in results between V coherence and Umass coherence.

Part of the difference in results between the UMass and V coherences is due to the poor rating by Umass of CREA topics well rated by V, as mentioned above, but it could also be due in part to the P parameter of these coherences. Indeed, it's possible that *PHP*, the main topic, appears in all the documents, but not repeatedly within them. For V consistency, this would favor topics that don't particularly highlight *PHP*, and therefore non-overlapping topics, and therefore the CREA method, as opposed to Umass, which would favor *PHP* omnipresence, and therefore overlapping topics, and therefore LDA.

4.3.1 V coherence

The results of the joint evaluation of LDA and CREA using coherence V give clear results in favor of the CREA method.

With the exception of the **all files** scenario giving mixed results, and even including the only case where LDA obtains a better score than CREA with TreeTagger+Babelfy pre-processing (C.1), all other situations give a much higher score to CREA (around 0.6) than to LDA (around 0.4).

Detailed results can be found in the appendix (Appendix C).

4.3.2 UMASS coherence

To make it easier to compare the results of V coherence and Umass coherence for this joint assessment, the scores assigned (originally between $-\infty$ and 0) by Umass coherence have been exponentiated to give scores between 0 and 1. It is important to note, however, that comparing raw scores from different coherences is meaningless, as only trends can be analyzed. The same applies to comparisons between different scenarios.

The results of the comparison between LDA and CREA using Umass consistency show a totally opposite trend, overwhelmingly in favor of LDA. Here, in all the pre-treatments and scenarios studied, the results for the LDA method exceed 0.75, while those for CREA peak at 0.17.

Detailed results can be found in the appendix (Appendix C).

Chapter 5

Discussion and Conclusion

5.1 Discussion

5.1.1 Related Work

The work carried out within this project is mainly based on the "*Technical Report no202306-techrep-goncalves, June 2023, revision c636a38*", 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.

5.1.2 Future Work

Perspectives for this work include resolving the conflict between the V coherence and the Umass coherence evaluations (it can be done by adding other secondary coherences to this work, to understand the resulting trends), including the LLM to the protocol by resolving the prompt size limitation problem, and applying the various tools to other data corpus where the topics are less one-sided (here with php). The development of other pre-processing subprotocols could also be included in these perspectives.

5.2 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 6

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 5)

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022. (pages 5 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 20)

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 5)

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

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 11)

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 15)

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 5)

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. (page 6)

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 15)

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.

Appendix A

Pre-processings for LDA

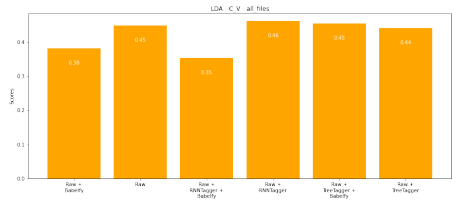

Figure A.1: Evaluation of pre-processings for LDA with C_V (Scenario: all files)

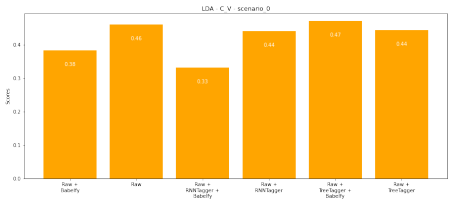

Figure A.2: Evaluation of pre-processings for LDA with C_V (Scenario: all files except CJA.txt)

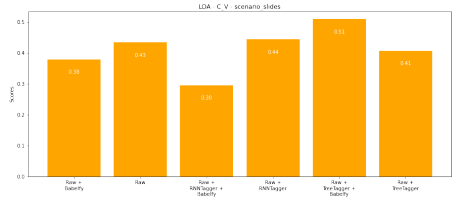

Figure A.3: Evaluation of pre-processings for LDA with C_V (Scenario: all slides)

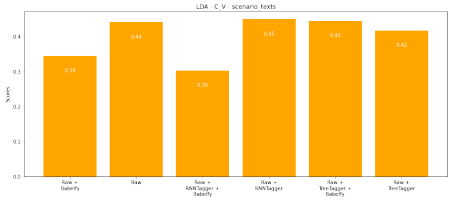

Figure A.4: Evaluation of pre-processings for LDA with C_V (Scenario: all texts)

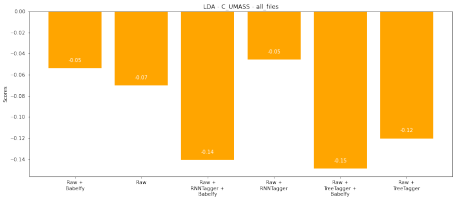

Figure A.5: Evaluation of pre-processings for LDA with C_{Umass} (Scenario: all files)

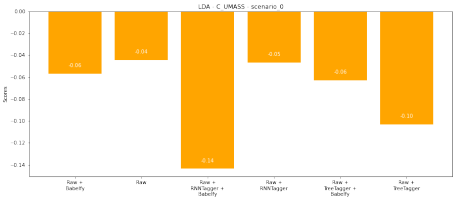

Figure A.6: Evaluation of pre-processings for LDA with C_{Umass} (Scenario: all files except CJA.txt)

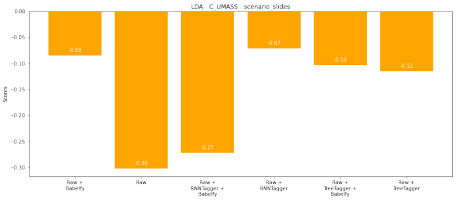

Figure A.7: Evaluation of pre-processings for LDA with C_{Umass} (Scenario: all slides)

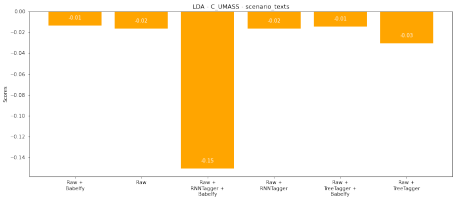

Figure A.8: Evaluation of pre-processings for LDA with C_{Umass} (Scenario: all texts)

Appendix B

Pre-processings for CREA

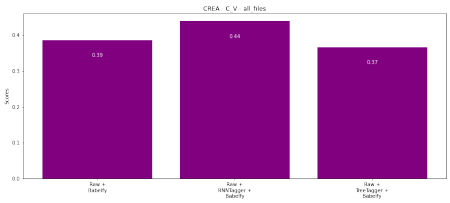

Figure B.1: Evaluation of pre-processings for CREA with C_V (Scenario: all files)

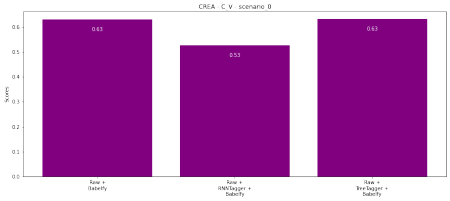

Figure B.2: Evaluation of pre-processings for CREA with C_V (Scenario: all files except CJA.txt)

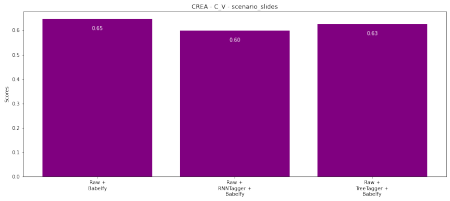

Figure B.3: Evaluation of pre-processings for CREA with C_V (Scenario: all slides)

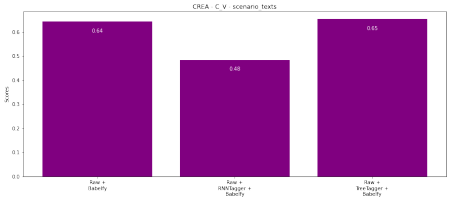

Figure B.4: Evaluation of pre-processings for CREA with C_V (Scenario: all texts)

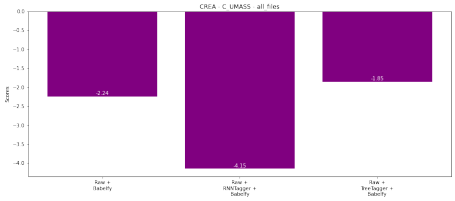

Figure B.5: Evaluation of pre-processings for CREA with C_{Umass} (Scenario: all files)

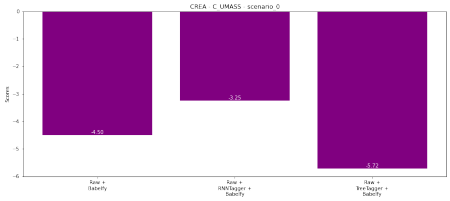

Figure B.6: Evaluation of pre-processings for CREA with C_{Umass} (Scenario: all files except CJA.txt)

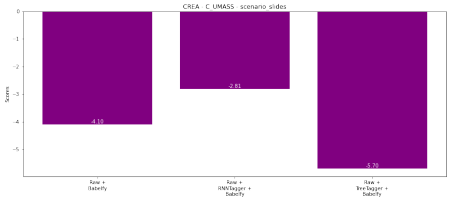

Figure B.7: Evaluation of pre-processings for CREA with C_{Umass} (Scenario: all slides)

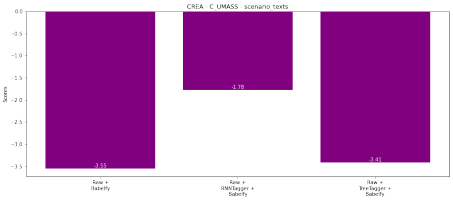

Figure B.8: Evaluation of pre-processings for CREA with C_{Umass} (Scenario: all texts)

Appendix C

Compared evaluation of LDA and CREA

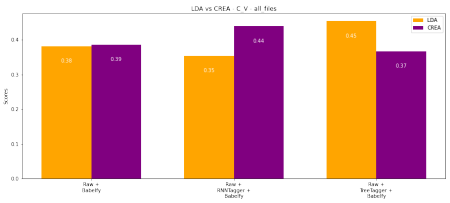

Figure C.1: Compared evaluation of LDA and CREA with C_V (Scenario: all files)

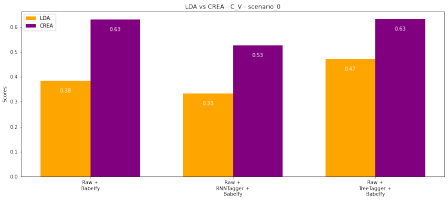

Figure C.2: Compared evaluation of LDA and CREA with C_V (Scenario: all files except CJA.txt)

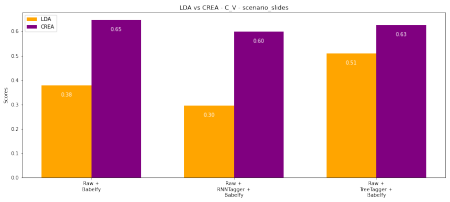

Figure C.3: Compared evaluation of LDA and CREA with C_V (Scenario: all slides)

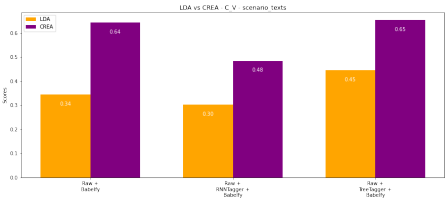

Figure C.4: Compared evaluation of LDA and CREA with C_V (Scenario: all texts)

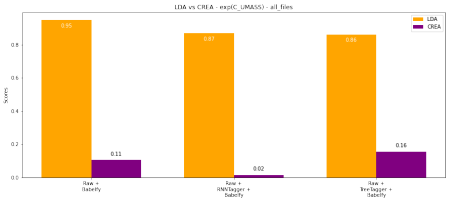

Figure C.5: Compared evaluation of LDA and CREA with exp(C_{Umass}) (Scenario: all files)

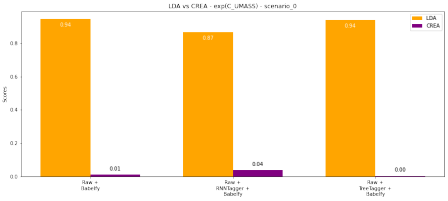

Figure C.6: Compared evaluation of LDA and CREA with exp(C_{Umass}) (Scenario: all files except CJA.txt)

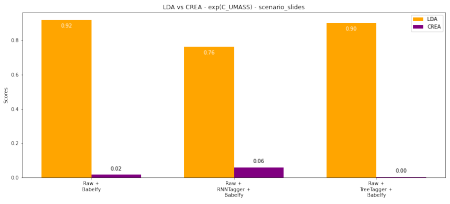

Figure C.7: Compared evaluation of LDA and CREA with exp(C_{Umass}) (Scenario: all slides)

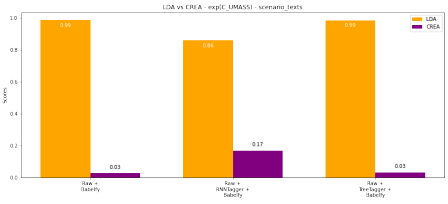

Figure C.8: Compared evaluation of LDA and CREA with exp(C_{Umass}) (Scenario: all texts)