MASTER'S THESIS 2024

Using Small LLMs to Assess and Enhance Skill Management Documents

Maxime Pakula

ISSN 1650-2884
LU-CS-EX: 2024-61

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2024-61

Using Small LLMs to Assess and Enhance Skill Management Documents

Använda små LLM:er för att utvärdera och förbättra kompetenshanteringsdokument

Maxime Pakula

Using Small LLMs to Assess and Enhance Skill Management Documents

Maxime Pakula
ma7013pa-s@student.lu.se

September 27, 2024

Master's thesis work carried out at Takima (Bagneux, FRANCE).

Supervisors: Pierre Nugues, Pierre.Nugues@cs.lth.se
François-Pierre Chalopin, fpchalopin@takima.fr

Examiner: Jacek Malec, Jacek.Malec@cs.lth.se

Abstract

In many companies, employees document their experiences and skills in *Records of Expertise*. These individual documents are updated continuously and often follow a specific structure and guidelines. Such documents are essential to match the business tasks to the right competences. Nonetheless, this process requires careful attention to both content and form. Experienced professionals often lack the time to help first-time writers. Recent advancements in natural language processing (NLP) and *large language models* (LLM) can help automate quality assurance and enhancement of those documents, enabling writers to focus more on content rather than formatting.

This Master's thesis focuses on using small LLMs and prompt engineering to evaluate the quality and enhance French *Records of Expertise*. In this work, I focused on a specific section of the *Records of Expertise* and stuck to particular guidelines. I tested a wide range of small models and used various prompts, experimenting with different instruction languages, guideline types, and prompt lengths. This approach allowed me to identify how to effectively prompt small LLMs for optimal performance in the desired tasks.

In particular, this Master's thesis demonstrates that models with approximately 7 billion parameters can achieve convincing quality evaluation, particularly when instructions are provided in French and adhere to a specific prompt structure. In particular, even if the approach was quite simplistic an **accuracy of 0.87** with an f_1 **score of 0.73** was achieved for the classification task. However, this approach encounters limitations when applied to the most complex guidelines. In addition to that, this research highlights the significance of utilizing those models to enhance document quality. While the methodology employed in this study does not consistently yield high-quality results, it paves the way for methodological refinements aimed at achieving more consistent improvements in document quality.

Keywords: Natural Language Processing, Large Language Models, Transformers, Text classification, Text enhancement, Prompt engineering, Skill Management Documents

Acknowledgements

To begin with, I would like to express my sincere gratitude to my supervisor from LTH, Pierre Nugues, for his unwavering support and guidance throughout the course of my Master's thesis. His professionalism and expertise were instrumental in shaping my understanding of the research process. With his deep knowledge in the field, he directed my work with precision and efficiency, helping me to formalize and articulate my findings effectively. His expert advice on writing was particularly valuable, enabling me to structure my ideas clearly and coherently. I am deeply thankful for his dedication and commitment throughout this journey.

I would also like to express my gratitude to my supervisor and CEO of Takima, François-Pierre Chalopin, for his support throughout my research journey. His efforts in providing all the necessary materials greatly facilitated my work, making the research process smoother. Despite the challenges of integrating into the company, he ensured that I could develop in the best possible conditions. His supervision and consistent interest in my progress were instrumental to my development. Lastly, the trust he placed in me served as a significant source of motivation and personal growth.

Contents

Chapter 1

Introduction

This Master's thesis investigates the application of large language models (LLMs) to assess the compliance of *Records of Expertise* with specific drafting guidelines. The *Record of Expertise* is a crucial document within a company, where employees detail their skills and experiences. This file serves as a central resource in skill management, providing insight into the knowledge and expertise available within the organization. Crafting these documents requires a degree of self-reflection and introspection, making the process challenging for those responsible for writing them. Furthermore, the task is often complicated by the need to adhere to specific formatting and content guidelines, posing an additional challenge, particularly for first-time writers who must navigate both the substance and structure of the document.

Creating a solution to verify compliance with corporate guidelines presents several significant challenges. Firstly, these guidelines vary from company to company, which complicates the development of a universal solution. Additionally, the data available for training such a tool is scarce, as it typically consists of only one document per employee. Furthermore, these documents are often unannotated, and the limited time resources in corporate environments make the application of traditional machine learning techniques particularly difficult.

This Master's thesis focuses on addressing the problem through the use of small LLMs with a typical size of a few billion parameters. These models are small enough to be hosted and operated within a company's premises, thereby ensuring data confidentiality. Furthermore, LLMs have the ability to capture the syntax of human language, eliminating the need to train traditional machine learning models on such limited datasets. The effectiveness of these models lies in their adaptability, as they can be guided to perform specific tasks through prompting and prompt engineering.

This Master's thesis was conducted in collaboration with Takima, a company based in France. One of the key challenges encountered during the development of the solution was that the documents were primarily in French. This presented an added difficulty, as most LLMs are predominantly trained on English data.

1.1 Background and Motivation

Today's capacity of storing natural language samples such as books, articles, and transcriptions digitally favors NLP advancements significantly. Proof of this is the wide application of NLP techniques in recent years in most everyday applications, for example, spelling corrections in standard mobile phones, machine translation engines like Google Translate, speech engines like Apple's Siri, and today's mass development of interactive virtual agents like chatbots [Ferrario and Nägelin, 2020].

Among recent advancements, LLMs stand out as a significant development in artificial intelligence. These models are trained on vast datasets, enabling them to effectively capture the nuances and intricacies of human language. For example, the Llama-1 models were trained on approximately 4.5 TB of data [Touvron et al., 2023a], allowing them to acquire a deep understanding of linguistic patterns. As a result, LLMs exhibit remarkable adaptability, capable of being directed to perform complex tasks without the need for additional retraining.

Furthermore, recent advancements have led to the development of models that are both efficient and lightweight, while also being open-source. A notable example is the Mistral 7B model [Jiang et al., 2023]. These models are particularly well-suited for tool development in constrained environments, making them ideal candidates for the specific requirements of this study.

1.2 The Company

This Master's thesis was conducted at Takima which is a French computer engineering consulting company. It seeks to match the profiles of its consultants with the needs of its clients. To do this, it uses an internal application in which each consultant completes a *Record of Expertise*. Consultants are responsible for completing their expertise files and salespeople use the expertise files to present consultant profiles to clients.

The company is exploring the potential of leveraging LLMs to automate specific *Record of Expertise* processing tasks and provide assistance to various users involved with these documents. The primary users include consultants and commercial partners, as well as the personnel responsible for ensuring the quality of the documents.

The envisioned applications include the detection of guideline violations, the suggestion of corresponding corrections, and the development of a research tool for sales personnel. In this Master's Thesis, the focus was on implementing the capability to detect guideline violations, with initial steps taken towards generating associated correction proposals.

1.3 Project Aims and Main Challenges

This research seeks to evaluate the feasibility of using prompt engineering with LLMs for classification and generation tasks, with a particular context that puts the focus on ensuring compliance with guidelines in the writing of *Records of Expertise*. The approach relies exclusively on the use of prompts, without any additional model training. Ultimately, the

study aims to propose a reproducible method for applying this strategy to similar cases and to evaluate its effectiveness.

Throughout this research project, several significant challenges were encountered. These included the limited volume of available data, an unlabeled datasets, issues related to data sovereignty, constraints in computing power, and the need to work with data in French.

Limited volume of available data: This study centers on *Records of Expertise*, with one existing document per consultant in the company. As a result, I had access to approximately one hundred documents. Handling a dataset of this magnitude introduces complexities in the use of metrics, as the associated uncertainties are more pronounced. Consequently, this necessitated the implementation of data augmentation techniques.

Unlabeled dataset: The *Records of Expertise* mentioned previously were not labeled, making it initially impossible to establish metrics. Therefore, I decided to base my work on a subset of reports deemed "good" by the quality assurance agent and intentionally degraded them to create a labeled dataset. This method had certain shortcomings, the impact of which I assessed by conducting a human evaluation of the final dataset.

Data sovereignty issues: The documents involved in this study are sensitive company records, containing not only the extent of the company's internal expertise but also the names of various clients. Due to data sovereignty concerns, the study had to be conducted on a computing server located within the company. This requirement introduced a significant challenge: limited computing power.

Limited computing power: In this study, I had access to a computing server equipped with two NVIDIA GeForce RTX 2080 Ti graphics cards, providing a total of approximately 22 GB of VRAM. This limitation influenced my choice of models and, in some instances, necessitated the use of quantized versions.

Data in French: Given that the company is based in France, the documents analyzed in this study are written in French. This raised an additional question: In this context, is it more appropriate to provide instructions in French as well, even though the models are predominantly trained in English?

1.4 Contributions of this Work

The main contributions of this master's thesis are as follows:

- Development and Evaluation of a Methodology for Test Data Set Creation: Proposes and assesses a method for generating a test data set from unlabeled texts through automated text degradation.
- Development and Evaluation of a Multi-Label Text Classification Methodology: Proposes and evaluates a methodology for multi-label text classification utilizing prompt engineering and LLMs.
- Performance Review of Various Models: Reviews the performance of several models in both text classification and text generation tasks.

Chapter 2
Previous Work

The tasks of classifying *Records of Expertise* according to guidelines and generating improved versions are distinct problems that should be addressed separately.

Classifying *Records of Expertise* according to guidelines: This section addresses the classification of *Records of Expertise* based on writing guidelines. *Records of Expertise* are crucial for the company as they not only help in training teams with skills aligned with specific missions but also serve to introduce consultants to clients. Consequently, it is natural for the format of these documents to be standardized to uphold a consistent quality standard that benefits all readers.

These guidelines prescribe the format for the various sections of the document, specifying the appropriate language level, the nature of the information to include, lexical fields to avoid, and common style errors to be prevented.

Consider a scenario where we focus on a section of a document that must adhere to two guidelines, A and B. We assign the labels VIOLATES_A and VIOLATES_B to these guidelines, respectively. Our goal is to apply these labels to the consultants' sections based on their compliance with the guidelines during text creation. This results in several possible situations.

- If the two guidelines are respected then no labels are associated with the text;
- If guideline A is respected but guideline B is not then the label VIOLATES_B is associated to the text;
- If guideline B is respected but guideline A is not then the label VIOLATES_A is associated to the text;
- If none of the two guidelines are respected then both VIOLATES_A and VIOLATES_B are associated with the text.

This issue is characterized as a multi-label problem because each text can be associated with none, one, or multiple labels. This contrasts with multi-class problems, where each text is assigned to only one category from several possible categories.

Generating improved *Records of Expertise* when the guidelines are not respected: Once non-compliance with a guideline is detected, the objective is to propose, where feasible, a revised version of the original text that adheres to the guideline. This aims to facilitate the consultant's work by providing a compliant alternative.

2.1 Multi-label Problems

A multi-label problem is one in which each instance may be associated with multiple labels. There are typically two ways of approaching these problems in the literature: Binary Relevance and Label Powerset. Those approaches are detailed in the following sections.

Binary Relevance (BR)

According to Zhang et al. [2018], the most intuitive solution for addressing this type of problem is the **Binary Relevance** principle.

The Binary Relevance principle involves decomposing the multi-label problem into several independent binary classification problems, one for each label. This approach has the advantage of simplifying the learning process by breaking it down into multiple, more manageable tasks. However, it also has the disadvantage of potentially ignoring inter-label dependencies, which may be crucial for accurately predicting the relationships between labels.

In his article, Zhang et al. [2018] explores the possibility of aligning the principle of Binary Relevance with the exploitation of inter-label dependencies. He specifically examines strategies such as Chaining Structure, Stacking Structure, and Controlling Structure.

Chaining Structure: In the Chaining Structure for multi-label classification, a sequence of binary classifiers is trained based on a predefined order of class labels. Each classifier in the chain predicts a label using the predictions from all preceding classifiers as additional features. This technique requires a predefined order for the labels and has the drawback of error propagation.

Stacking Structure: The Stacking Structure for multi-label classification involves two distinct layers of classifiers. Base-level classifiers are binary classifiers trained independently on the multi-label dataset using the Binary Relevance approach, each focusing on a single class label. Meta-level classifiers are then trained on a meta-level dataset that incorporates the outputs of all base-level classifiers as additional features. This technique does not require a predefined order of labels and reduces the risk of error propagation.

Controlling Structure In the Controlling Structure, two layers of binary classifiers are used, divided into base-level and meta-level classifiers. Base-level classifiers are trained using Binary Relevance, while meta-level classifiers are trained using a pruned version of these base-level predictions, based on a Bayesian network or directed acyclic graph that captures label correlations. This approach models conditional dependencies among

labels to predict relevancy, simplifying the learning process by focusing on key label correlations and managing computational complexity.

Zhang et al. [2018] also introduces two challenges that happen in multi-label learning: **class imbalance** and **label importance** and some potential solutions. These two problems hinder the effectiveness and accuracy of multi-label learning models.

Class imbalance: Class imbalance occurs when certain labels have disproportionately fewer positive instances compared to negative ones, which can lead to biased model predictions and reduced performance.

Label importance: Label importance arises from the fact that not all labels carry the same level of relevance in describing an instance. Standard approaches often assume equal importance for all labels, neglecting the varying degrees of relevance that different labels may have.

Label Powerset (LP)

Label Powerset (LP) is another method used in multi-label classification and described by Tsoumakas and Vlahavas [2007] that approaches the problem by treating each unique combination of labels from the label set as a distinct label in itself. Instead of predicting multiple labels independently, LP creates a single-label classifier that maps input features to the powerset of the label set which encompasses all possible subsets of labels. This method allows the classifier to directly predict the exact combination of labels for a given instance.

One of the key advantages of the LP method is its ability to consider correlations between labels. By treating label combinations as distinct entities, the LP approach captures the relationships and dependencies between labels, potentially leading to more accurate predictions, especially when labels are interdependent.

However, LP also comes with significant challenges. The primary disadvantage is the large number of possible label subsets, especially as the size of the label set increases. Many of these subsets may correspond to very few examples in the training data, making it difficult for the model to learn effectively. This can lead to issues with model generalization and scalability, particularly in cases with a large label space or limited data.

To overcome this limitation, Tsoumakas and Vlahavas [2007] introduces RAKEL (RAndom k-labELsets) which is an ensemble method built upon the Label Powerset (LP) approach. Instead of using all labels, each LP classifier in RAKEL is trained on a small, randomly selected subset of labels (referred to as "k-label sets"). By reducing the label space for each classifier, RAKEL manages to maintain the ability to capture label correlations while avoiding the data sparsity problem associated with traditional LP. The final classification decision in RAKEL is made by aggregating the predictions from all the individual LP classifiers in the ensemble, typically using a voting mechanism.

2.2 Label Extraction

Label extraction is a fundamental component of sentiment analysis. This section reviews the evolution in techniques in that field.

2.2.1 Lexicon and Machine Learning Based Extractions

Taboada et al. [2011] highlights the two predominant approaches in the field of sentiment analysis: lexicon-based extraction and machine learning–based extraction.

Lexicon-based extraction: Lexicon-based extraction relies on dictionaries that are either manually curated by experts or generated automatically, where words are assigned specific sentiment values. The label for a given text is determined by aggregating the sentiment values of all the constituent words. This method necessitates the prior existence or creation of appropriate lexicons to be effective.

Machine learning–based extraction: Machine learning–based extraction, on the other hand, depends on datasets of labeled text. Traditional machine learning models are trained to identify patterns associated with specific labels, such as n-grams, grammatical tags, or embeddings. This approach requires large amounts of labeled data to accurately learn and generalize the patterns related to sentiment labels.

2.2.2 Transformer Based Extraction

More recent studies such as [Miah et al., 2024] instead use approaches based on the Transformers architecture and trained versions of BERT as well as LLMs like GPT-3.

Twitter-RoBERTa-Base-Sentiment-Latest: A pre-trained sentiment analysis model based on RoBERTa architecture, specifically fine-tuned for sentiment analysis on Twitter data.

BERTweet-Base-Sentiment-Analysis: Another pre-trained model designed for sentiment analysis, based on the BERTweet architecture, which is fine-tuned on Twitter data.

GPT-3: An LLM developed by OpenAI, known for its ability to generate human-like text and perform a variety of language tasks, including sentiment analysis.

In particular this is what Miah et al. [2024] says about their usage of GPT-3:

> The GPT-3 model is leveraged to generate the sentiment of the given text based on a fixed prompt “provide the sentiment of the given text in a single class from positive, negative and neutral”.

2.3 Tactical and Strategic Feedback

According to Drewery et al. [2022] it appear that AI can leverage tactical feedback way more than strategic feedback. Their paper investigates the use of an AI-based resume review tool, distinguishing between tactical and strategic feedback. Tactical feedback focuses on improving phrasing and addressing grammar and syntax errors, whereas strategic feedback involves guidance on what information should be included. The study finds that students utilizing AI

tools achieved more significant tactical learning outcomes but did not gain enhanced strategic insight compared to those receiving traditional critiques. The role of AI appears to be limited to identifying writing mechanics, which may not be as beneficial for students seeking strategic guidance on content summary. The findings suggest that AI is more effective for students who already understand their objectives and require specific feedback, rather than for those needing comprehensive strategic advice.

2.4 Takeaways

The literature review has led me to adopt a "Binary Relevance" approach for addressing the multi-label classification problem by converting it into several binary classification tasks. Given the limited data available, employing a "Label Powerset" approach would be impractical. Additionally, due to time constraints, this study will not address issues related to inter-label correlation, class imbalance, or label importance.

Recent advancements suggest that label extraction methods utilizing Transformer architectures and in particular LLMs have demonstrated convincing performance compared to traditional approaches based on lexicons and classical machine learning techniques. This indicates that leveraging the capabilities of language models is advantageous when data constraints preclude the use of more conventional methods. Thus this study will use LLMs as the basis for its classifiers.

Furthermore, evidence suggests that language models perform better in providing tactical rather than strategic guidance. Consequently, the development and evaluation of the tool will emphasize tactical recommendations over strategic ones.

Chapter 3
Theoretical Background

Building on the decision to utilize LLMs as the foundation for the classifiers, this chapter delves into their operation by exploring key components, including embedding, tokenization, and the Transformer architecture.

3.1 Word Embedding

In the field of NLP there is a need for representing efficiently words and documents. As highlighted by Khem et al. [2023] there are two word representation: word encoding and word embedding. They each serving distinct purposes in transforming textual information into a machine-readable format.

Word encoding: Word encoding involves converting text into unique numerical representations. This method primarily focuses on enabling machines to handle text through numerical values, preserving basic patterns and relationships within the text but lacking in-depth semantic understanding.

Word embedding: Word embedding provides a more sophisticated and continuous vectorial representation of words. Each word is mapped to an N-dimensional vector, where the dimensions capture latent semantic features. This allows for a richer representation where the proximity of vectors in the space reflects semantic similarity.

3.1.1 One Hot Encoding

One of the core components of word embedding is the one hot encoding method. It is used to encode words as vectors so that they can be manipulated by ML algorithms but it does not capture syntactic and semantic information about words.

Given a fixed vocabulary set of size V, this encoding method can be applied to represent words as a sparse vector of size V in which each dimension corresponds to a word and the value one codes for the word that is being represented.

3.1.2 Neural Network Language Model (NNLM)

The Neural Network Language Model was introduced by Bengio et al. [2000]. It is a model that introduces the distributed representation of words to avoid the curse of dimensionality.

The model is composed of the following layers:

- **The input layer**: It consists of the N words preceding the word to guess. Each one of them is represented by its one-hot encoded vector of size V (the vocabulary size);
- **The projection layer**: Each one of the input vectors is mapped on a feature vector of size D (dimension of the feature vector) using the same shared projection matrix of size $V \times D$;
- **The hidden layer**: It is a classic feed-forward layer of size H followed by a $\tanh$ activation function, representing an internal state of the model;
- **The output layer**: It is a vector of size V that outputs the probabilities for each word of the vocabulary to be the word to guess. It consists of a classic feed-forward layer and a soft max function.

The goal of the model is to guess the next word based on the preceding N words.

During training the model learns the distributed representation (feature vector) via the projection matrix as well as the probability function.

The distribution representation gave much better results than the previous n-gram models but the author recognized that the architecture as well as speeding up techniques could be investigated to improve performances. The author especially highlighted that RNN could be used.

Based on this work, variations and improvements have been proposed, notably with the Recurrent Neural Net Language Model (RNNLM) by Kombrink et al. [2011] and the Word2vec models by Mikolov [2013] but the principle remains similar.

3.2 Tokenization

In practice, the encoding process is applied to tokens rather than words due to challenges such as large vocabulary sizes and out-of-vocabulary words. Tokenization allows for the segmentation of any text, though this segmentation is not necessarily performed on a word level.

Zhao et al. [2023]'s article outlines three common tokenization techniques:

1. **Byte-Pair Encoding (BPE)**: Originally developed as a compression technique, BPE creates tokens by iteratively merging the most frequent pairs of tokens.
2. **WordPiece**: This technique involves the iterative fusion of tokens to enhance the likelihood of training data coverage.

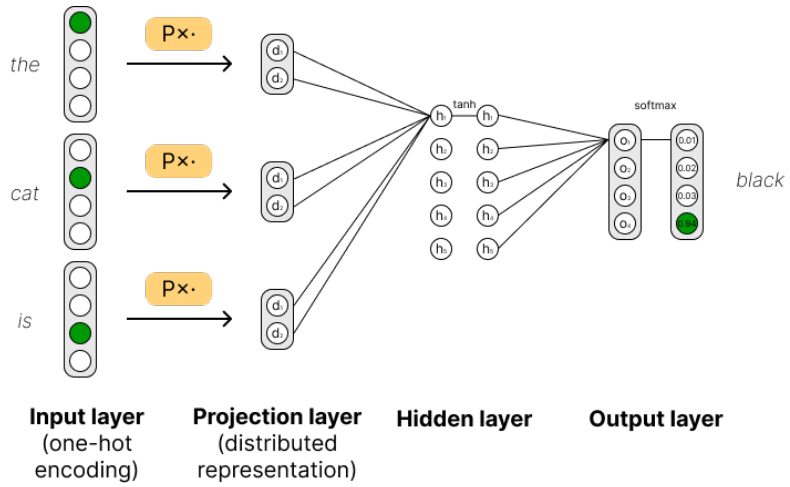

Figure 3.1: NNLM example

3. **Unigram**: This method involves removing the least useful tokens from a large set of hypothetical tokens, based on their impact on the training data.

3.3 Transformers Architecture

The transformer architecture, introduced by Vaswani et al. [2017], represents a groundbreaking advancement in natural language processing. This model has replaced traditional language models that relied on recurrent or convolutional networks. Its key innovation is the attention mechanism, which fundamentally enhances the model's ability to process and generate language.

3.3.1 Attention Mechanism

Historically, LLMs have relied on sequence-based or convolution-based approaches. In these models, the input sequence is processed token by token, with information stored in a memory cell within the encoder. The output sequence is generated incrementally by interpreting the stored information.

However, this approach has several significant limitations:

- Difficulty in capturing long-range dependencies
- Lack of suitability for parallelization
- Potential for vanishing or exploding gradients

This is where the attention mechanism comes into play. It employs a layer that produces a weighted sum of various input values for each output.

The variability in weights for different outputs allows the attention mechanism to distribute focus differently over the inputs, hence the term "attention layer".

At the core of the attention mechanism there are three concepts:

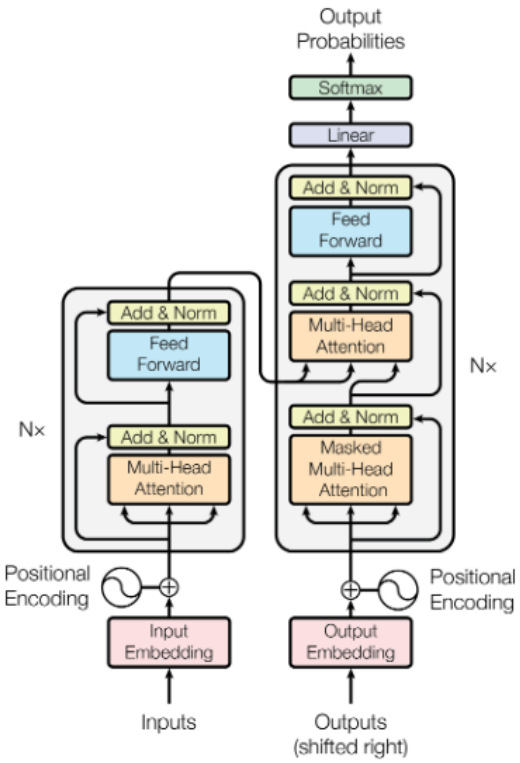

Figure 3.2: The Transformer model architecture – *Attention is all you need*

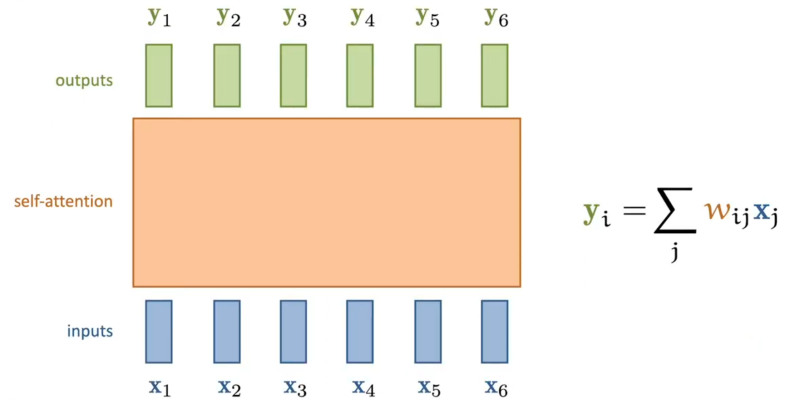

Figure 3.3: Attention layer. After DLVU, *Lecture 12.1 Self-attention*, `https://www.youtube.com/watch?v=KmAISyVvE1Y`

Figure 3.4: The attention mechanism allows a focus on the values. After Halfling Wizard, *Attention Mechanism In a nutshell*. `https://www.youtube.com/watch?v=oMeIDqRguLY`

- The **values** are the inputs to the layer and contribute to the weighted sum for each output. The weights are calculated using the other two concepts: query and key;
- **Queries** correspond to the inputs. For each input, a query is matched with the output's key to determine the attention level assigned to that input in the output computation;
- **Keys** relate to the outputs. For each output being generated, the relevant key is matched against the queries to compute the weights for the weighted sum.

Self attention mechanism

In the self-attention variant of the attention mechanism the values, keys and queries are all based on the inputs. The following value, key and query transformation are introduced.

$$\begin{aligned} \mathbf{k}_i &= \mathbf{K}\mathbf{x}_i + \mathbf{b}_k \\ \mathbf{q}_i &= \mathbf{Q}\mathbf{x}_i + \mathbf{b}_q \\ \mathbf{v}_i &= \mathbf{V}\mathbf{x}_i + \mathbf{b}_v \end{aligned} \tag{3.1}$$

Multi-head attention

In the multi-head attention variant of the attention mechanism, the different dimensions of the inputs are splitted and each head focuses on a subpart of these dimensions with their own smaller attention layer. The ouputs are then extracted by concatenating the outputs of

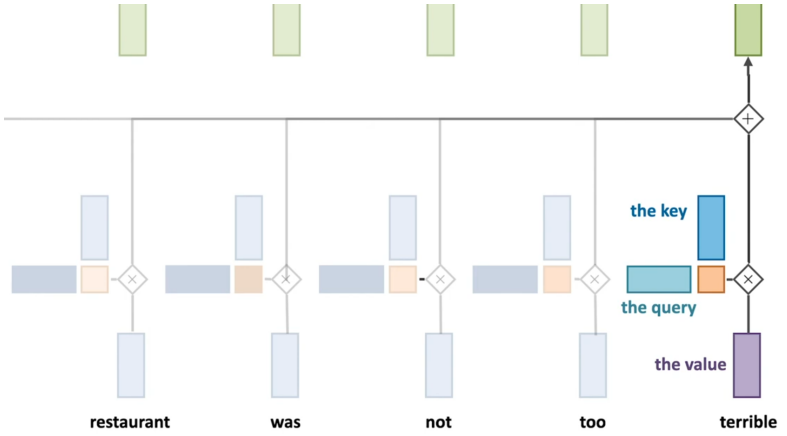

Figure 3.5: The query-key-value trio. After DLVU, *Lecture 12.1 Self-attention*, `https://www.youtube.com/watch?v=KmAISyVvE1Y`

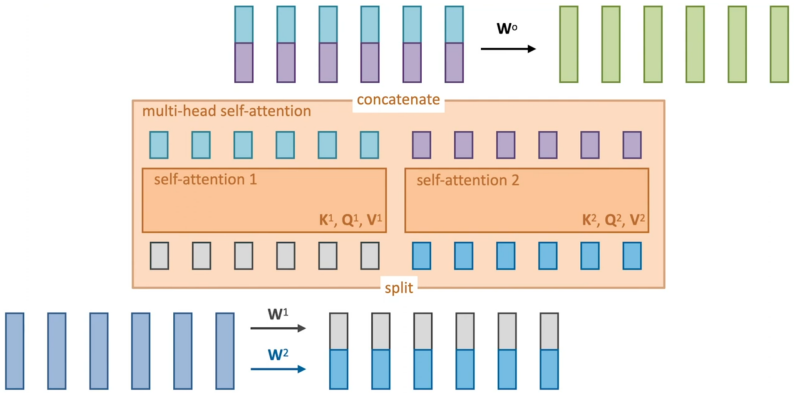

Figure 3.6: Multi-head attention with 2 heads. After DLVU, *Lecture 12.1 Self-attention*, `https://www.youtube.com/watch?v=KmAISyVvE1Y`

the different heads. This has no impact on the number of parameters but helps to capture different kinds of relations between the inputs.

3.3.2 Positional Encoding

Positional encoding is a technique employed in transformer models to integrate information about the order of tokens within a sequence. This method involves augmenting input embeddings with additional vectors that encode positional information. Specifically, sinusoidal functions of varying frequencies are used to generate these encodings, with sine and cosine functions of different wavelengths. By facilitating the model's ability to extrapolate to sequence lengths beyond those encountered during training, positional encoding significantly enhances the model's capacity to process and learn from sequential data despite its non-sequential architecture.

Chapter 4
Methodology

In this chapter, I provide an overview of the methodology employed in this research. The chapter begins with a summary of the initial data processing steps and the literature review conducted. Following this, I describe the project's infrastructure, software architecture, and structure to ensure that the approach is reproducible. Lastly, I discuss the rationale behind the selection of models, the formulation of prompts, and the choice of metrics.

4.1 Data Processing

Initially, I conducted an exploratory data analysis (EDA) to better understand the characteristics of the available data. Based on the insights gained from this analysis, I selected a specific section of the dataset, referred to as the *Synthetic Profile*, and narrowed the focus to three key guidelines related to this section. See detailed information in Section 5.2.

4.2 Literature Review

Following the exploratory data analysis (EDA), I undertook a literature review with a focus on multi-label classification challenges, label extraction, and the application of language models to documents similar to the *Records of Expertise*. This research guided me towards employing a "Binary Relevance" approach with LLMs as classifiers. Additionally, I opted to focus on tactical rather than strategic guidelines, given that language models have demonstrated greater efficacy in tactical applications. Subsequently, I examined the various components of LLMs, with particular emphasis on the Transformer architecture.

4.3 Infrastructure and Software Architecture

For performing inference with language models, I utilized a computing server equipped with two NVIDIA GeForce RTX 2080 Ti graphics cards, providing approximately 22 GB of VRAM. Based on this setup, the exploration of several architectures was carried out:

MlFlow: Initially, an attempt to use a MlFLow architecture to load and use models was made, without success. It seems that the failure is due to compatibility issues between versions of the different drivers.

LM Studio: Simultaneously, I conducted inference using LM Studio, where I achieved promising initial results with this architecture. However, to achieve better scalability, I eventually transitioned to a different architecture.

Dockerized Ollama: The final architecture employed is based on Ollama and is containerized using Docker. This setup enabled efficient server configuration and the ability to load multiple models simultaneously. Specifically, the Ollama image used was *ollama/ollama:0.3.4*.

4.4 Project's Structure

A pipeline structure was designed to perform the various measurements. This pipeline consists of several steps:

Test set creation using data augmentation: This step involves generating the test datasets for two tasks: text classification and text improvement. The datasets are created using input texts deemed "high quality" that adhere to all guidelines, along with degradation prompts applied to these high-quality texts to produce versions that violate specific guidelines. This process is detailed in Section 5.3. Following this process, the pipeline diverges into two distinct branches: one dedicated to the classification task and the other to the improvement task.

Testing the model on the classification task: This branch corresponds to testing the tool on the classification task. It consists of two successive steps: label prediction and metric calculation.

- **Prediction using an LLM:** In this step, the LLM is queried using a prompt to classify the texts in the test dataset. The model is instructed to respond in JSON format. The response is then truncated between the first opening brace and the first closing brace, and interpreted as JSON using Python's *json* package. Finally, the prediction is extracted from this JSON format.
- **Computation of the metrics:** In this step, the predictions are compared to the actual labels in the test set, and the corresponding metrics are calculated and exported.

Testing the model on the enhancement task: This branch is dedicated to testing the tool on the text improvement task. It involves two sequential steps: text improvement and metric calculation.

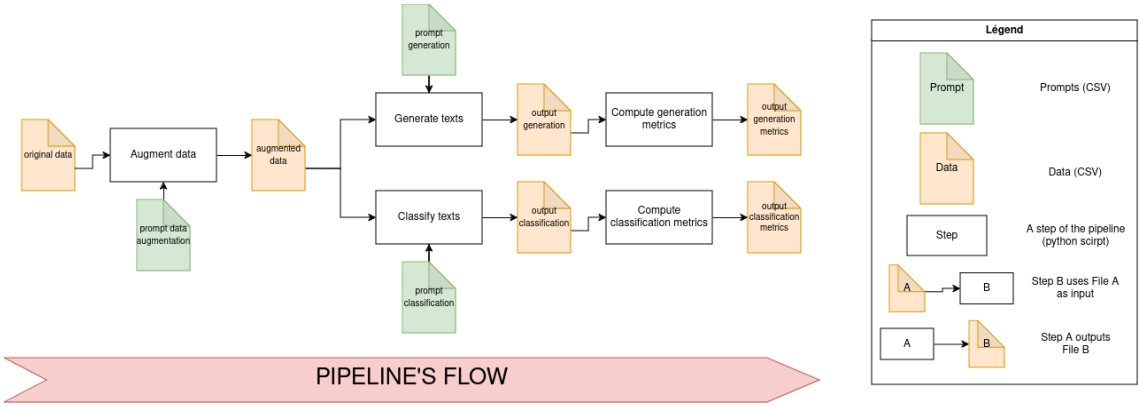

Figure 4.1: Pipeline's flow

Enhancement using an LLM: In this step, the LLM is queried using a prompt with the test texts that have been labeled. The model is tasked with generating an improved version of each text that addresses the guideline associated with the label.

Computation of the metrics: In this step, the improved texts are compared to the target texts in the test set, and the corresponding metrics are calculated and exported.

It is important to note that the pipeline structure is configurable, allowing for the execution of only specific parts as needed. For instance, I generated the test datasets just once and subsequently reused them across the classification and improvement branches, employing different prompts and models for each task. This approach is crucial as it ensures that all results are based on the same test dataset, facilitating meaningful comparison of the obtained values.

4.5 Models, Prompts and Metrics

The selection of language models was influenced by the choice of software architecture, as the models needed to be compatible with *Ollama*. Additionally, the available computing power, approximately 22 GB of VRAM, constrained the selection. In addition to that, the focus was on large and recent model families, with particular attention given to testing models of various sizes. Further details are provided in Chapter 6.

The prompts were crafted according to best practices identified in the literature review. Specifically, a taxonomy of prompts was employed to ensure that the results contribute to the evaluation and enhancement of prompting techniques within the research community. Additional details can be found in Chapter 8.

The selection of metrics was guided by insights from the literature review. Emphasis was placed on using established and widely accepted metrics to facilitate the evaluation and comparison of results both within this study and with findings from other research. Further details are provided in Chapter 7.

Chapter 5

Dataset

The dataset is composed of *Records of Expertise*. A *Record of Expertise* is a document owned by a consultant in which the different skills and experiences of the consultant are described. It is crucial for the company as it does not only help in composing teams with skills aligned with specific missions but also serve to introduce consultants to clients.

The dataset comprises 327 *Records of Expertise*, which belong to current and former consultants of the company. The documents vary in quality and maturity, since the structure of the document evolved and was gradually standardized over time.

5.1 The Structure of a *Record of Expertise*

A *Record of Expertise* is organized into several sections. The following two sections of the paper provide a detailed examination of each section of the *Record of Expertise*. First, we will focus in-depth on the *Synthetics Profile* section, which will be the primary focus of the remainder of the paper. For completeness, the other sections will also be described.

5.1.1 Synthetic Profile

The *Synthetic profile* is the first section of a consultant's *Record of Expertise*. It is a section that aims to introduce the consultant's personality, professional journey and expertise.

Here is a example of a Synthetic Summary in French and an English translation:

> Plongé dans le monde passionnant de l'expérience utilisateur (UX), je trouve ma véritable essence. Mon parcours est une symphonie d'empathie, de créativité et de rigueur, orchestrée pour créer des expériences numériques qui transcendent le simple usage pour devenir des moments inoubliables. Je suis animé par la quête constante de comprendre les besoins profonds des utilisateurs et de traduire ces insights en interfaces élégantes et fonctionnelles. Avec une palette

de compétences comprenant la conception centrée sur l'utilisateur, le prototypage interactif et les tests utilisateurs, je sculpte des expériences qui résonnent avec les cœurs et les esprits.

'Immersed in the exciting world of user experience (UX), I find my true essence. My journey is a symphony of empathy, creativity and rigor, orchestrated to create digital experiences that transcend simple use to become unforgettable moments. I am driven by the constant quest to understand deep user needs and translate these insights into elegant and functional interfaces. With a skill set including user-centered design, interactive prototyping, and user testing, I sculpt experiences that resonate with hearts and minds.'

5.1.2 Other Sections

The rest of the research will be based on the *Synthetic Profile* section but the other sections are listed here for completeness.

Career synthesis: The *Career synthesis* section is a paragraph that aims to go into more details about the career path of the consultant.

Activities: The *Activities* section contains a paragraph about the different activities that the consultant has been part of.

The hard skills: The *Hard skills* section lists the main hard skills of the consultant. Typically software development technologies.

Soft skills: The *Soft skills* section lists the main soft skills of the consultant.

Mission: The *Missions* section contains a list of missions the consultant has done inside the company or at client companies. A mission contains:

- The title of the mission;
- The start and end date;
- The role that the consultant has had;
- A few paragraphs (usually four) explaining the context of the mission, the goals, the tasks achieved and the outcomes;
- A list of the associated hard and soft skill.

Diplomas: The *Diplomas* section contains the latest diploma of the consultant as well as other certifications that the consultant have passed. Each entry contains:

- The title of the diploma or certification;
- The associated school or organism;
- The grade;
- The year graduation.

Trainings: The *Trainings* section contains a list of the diffenrent trainings that the consultant have done. A training entry includes:

- A title;
- A high level description of the training;
- A list of associated hard and soft skills;
- The year that the training happened.

Languages: The *Languages* sections contains a list of the languages the consultant can speak and the level of fluency associated.

References: The last paragraph of the *Record of Expertise* is called *References*. It contains a list of contributions of the consultant. These contribution can be but are not limited to:

- Conferences;
- Blog posts;
- Technology watch;
- Personal projects;
- Contribution to other projects.

5.2 Exploratory Data Analysis

As explained by Data et al. [2016], Exploratory Data Analysis (EDA) is a crucial step in research, as it helps in understanding the features and potential issues within a dataset. The primary goals of EDA are to:

Examine the data: This includes analyzing data distribution, identifying outliers, and detecting anomalies.

Understand the data: This is achieved through graphical representations that provide insights into the underlying patterns and relationships.

In this analysis, I aimed to examine the distribution of word counts across different sections. Following the recommendations of Data et al. [2016] for continuous and univariate data, I employed a histogram representation. The results are illustrated in Figure 5.1.

I used the results of this analysis to narrow the focus to a single section of the document for this research paper. The analysis revealed that certain sections, such as *Training*, *Activity Synthesis*, and *Career Synthesis*, are frequently left empty, making them less relevant candidates. Among the remaining sections, the *Synthetic Profile* appears to be the most consistently completed. Its distribution resembles a Gaussian bell curve, suggesting that significant attention is given to filling out this section. Therefore, I chose to focus on the *Synthetic Profile* section for the remainder of the study.

5.3 Test Set Crafting

To effectively evaluate the quality of responses generated by a selected model and prompt, a test dataset is essential. For the classification task, this requires a labeled dataset. In

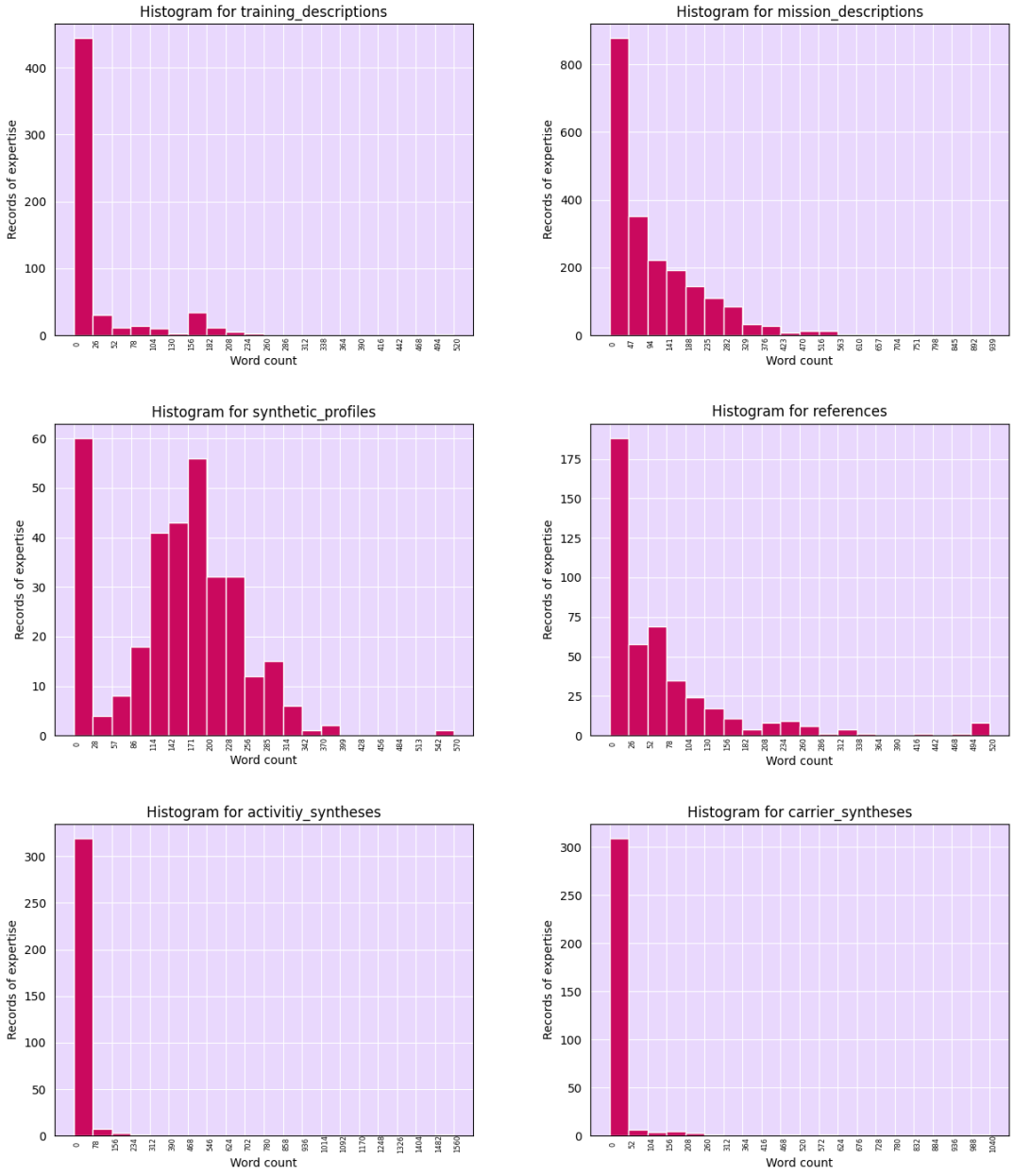

Figure 5.1: Number of words per document in the different sections

contrast, for the task involving text generation or enhancement, a dataset comprising text pairs—consisting of an input text and a corresponding target or reference text—is necessary.

For the classification task, the test set is used to compare the labels predicted by the model with the actual labels of the texts. This allows for the calculation of metrics that indicate how accurately the model assigns each label. Detailed information about the selected metrics can be found in Section 7.1. Consequently, it is necessary to manually label a sufficient portion of the available texts, ideally with the assistance of domain experts, to create an adequately sized test set.

For the generation or improvement task, the test set is designed to compare the target text with the variation generated by the tool based on the input text. This comparison can be evaluated using metrics such as BLEU and ROUGE scores, as discussed in Section 7.2. While it is possible to utilize previously labeled data for guideline labels, it is also necessary to create objective target texts. This process requires significant time and the assistance of a domain expert.

Given the time constraints and limited availability of experts for this task, I developed an alternative method based on data augmentation to create a test dataset for both tasks. This approach significantly reduces the time required from experts compared to the manual methods previously mentioned. The method is outlined below in Subsection 5.3.2 and its effectiveness is evaluated in Subsection 5.3.3.

5.3.1 Choice of Labels

From the list of guidelines provided, I have selected three to prioritize for this study, focusing on their application in the methodology. Due to time constraints, the methodology was not implemented across all the guidelines in the end.

Don't talk about passion all the time: The consultant should not overuse the lexical field of passion. The label **OVERUSE_OF_PASSION (O_Pa)** was derived from it.

Do not use the verb "permit" too often: The consultant should not overuse the verb "permit". The label **OVERUSE_OF_PERMIT (O_Pe)** was derived from it.

Let your personality shine through: The consultant should let its personality shine through the paragraph. The label **NOT_ENOUGH_PERSONALITY (N_E_P)** was derived from it.

5.3.2 Custom Method for Test Set Crafting

The method is based on the following assumptions:

H1: There are high-quality *Synthetic Profiles* that adhere to all or nearly all of the guidelines;

H2: An expert is available to select the best *Synthetic Profiles* and provide an example of how to intentionally degrade a DE to violate a specific guideline for each guideline;

H3: Given such an example, an LLM can plausibly degrade a high-quality *Synthetic Profiles* to violate the intended guideline.

The method I applied involves three steps:

Selection of High-Quality *Synthetic Profiles*: An expert selects the best documents based on their judgment. For one of these *Synthetic Profiles*, the expert provides examples of how each guideline can be violated individually.

Degradation of High-Quality *Synthetic Profiles*: Using the examples of guideline violations, prompts are created to intentionally degrade the *Synthetic Profiles* according to specific guidelines. These prompts, along with an LLM, are used to generate degraded versions of the high-quality *Synthetic Profiles*.

Creation of Test Sets: Test sets are created from the original and degraded data:

- For the classification task test set, the high-quality *Synthetic Profiles* remain unlabeled, as they are assumed to adhere to all guidelines. The degraded *Synthetic Profiles* receive labels corresponding to the specific guidelines they were designed to violate, resulting in a labeled dataset.
- For the improvement task test set, the degraded *Synthetic Profiles* serve as the input texts, while the original high-quality *Synthetic Profiles*, which are considered to meet all guidelines, serve as the target texts. This results in a dataset with labeled input texts and corresponding target texts.

An additional step was considered to expand the test dataset by using data augmentation techniques, specifically through successive translations, also known as back-translation. This was inspired by [Sennrich et al., 2016].

The expert provided 21 texts deemed "high-quality." By applying the method described above, I obtained the following test sets:

Test Set for the Classification Task: This set consists of 84 labeled texts, including:

- 21 original texts, with no labels;
- 21 degraded texts labeled with the O_Pa label;
- 21 degraded texts labeled with the O_Pe label;
- 21 degraded texts labeled with the N_E_P label.

Test Set for the Generation/Improvement Task: For each fo the 3 correction prompt, 21 pairs were created, each consisting of an input text and a corresponding target text

5.3.3 Evaluation of the Test Set

Given the strong assumptions underlying the method described in the previous paragraph, I decided to assess the quality of the test set through a human evaluation done by an expert. The following protocol was conducted for this evaluation.

Texts were selected successively from the entire collection to create a set of 12 texts with the following distribution: 3 original texts, 3 texts labeled with O_Pa, 3 texts labeled with O_Pe, and 3 texts labeled with N_P_E. If a randomly selected text was the fourth in its category or had already been included, it was discarded. The expert then evaluated these 12 texts without knowledge of their labels, using the following scales:

Q1: Plausibility of human written text Sur une échelle de 1 à 5, selon toi ce texte a-t-il été écrit par un humain ou est une variation générée par un modèle d'IA ? La note 1 corespond à un texte qui est clairement une variation générée par IA et la note 5 corespond à un texte qui est clairement écrit par un humain.

'On a scale of 1 to 5, do you think this text was written by a human or is it a variation generated by an AI model? A score of 1 corresponds to a text that is clearly an AI-generated variation and a score of 5 corresponds to a text that is clearly written by a human.'

Q2: Reasonable label association Sur une échelle de 1 à 5, à quel point es-tu en accord avec les affirmations suivantes ? Une note de 1 est un désaccord total et une note de 5 est un accord total.

1. L'auteur ne laisse pas transparaître sa personnalité;
2. L'auteur parle de manière trop passionnée;
3. L'auteur utilise trop souvent le verbe "permettre" et ses dérivés.

'On a scale of 1 to 5, how much do you agree with the following statements? A score of 1 is total disagreement and a score of 5 is total agreement.

1. The author does not let his personality shine through;
2. The author speaks too passionately;
3. The author uses the verb "allow" and its derivatives too often.

'

The results for *Q1* are presented in Table 5.1. The expert's evaluation reveals that there is no clear separation between original texts and those generated by an LLM, suggesting that the generated texts are relatively credible. However, it is important to note that AI-generated texts are still rated lower on average than the original texts, indicating that further improvements could be made to enhance their credibility.

	1	2	3	4	5
Original texts	0	0	1	1	1
LLM generated variations	2	3	1	1	2

Table 5.1: Test set evaluation: Plausibility of human written texts

The results for Q2 are detailed in Table 5.2. This table shows the label tested in the column *Tested label*, along with the expert's evaluations based on whether the label was assigned to the text or not within the test set.

It is observed that for the O_Pa and O_Pe labels, there is a correlation between the labels assigned using the method and the expert's judgment. While this correlation is not perfect, it suggests that the results of this study are meaningful and interpretable. In contrast, for the N_E_P label, the assigned labels appear to be uncorrelated with the expert's assessment.

Tested label	Assigned label	1	2	3	4	5
O_Pa	True	0	0	0	0	3
	False	3	2	1	1	2
N_E_P	True	2	0	0	0	1
	False	4	1	0	2	2
O_Pe	True	1	0	0	0	2
	False	7	0	0	0	1

Table 5.2: Test set evaluation: Reasonable label association

In conclusion, it appears that some of the assumptions made for creating the test set are not fully validated. Specifically, the application of the N_E_P label does not align with human judgment, suggesting that results related to this label should be interpreted with caution. However, the application of the other labels generally corresponds well with human judgment, which supports a relative interpretation of the results obtained.

Chapter 6
Models

The selection of an appropriate LLM is critical for research and tool development, particularly when dealing with specific linguistic and computational and data sovereignty constraints. Our focus is on identifying models that are open source, ensuring privacy and the ability to customize the models for specific needs. Additionally, for practical deployment considerations, we also prioritize models that are lightweight, enabling efficient execution on local servers and minimizing inference costs. These criteria are essential to balance the performance with the operational efficiency of language models in our research and tool development.

The following sections offer a comprehensive overview of some of the largest families of language models, focusing specifically on those relevant to this study. For each model family, the producing company is introduced, followed by a detailed description of the various models, including their architectures, licenses, and any pertinent additional information. An overview of all the selected models is available in Table 6.1.

6.1 The Mistral Family

According to [Wikipedia, 2024c] and [AI, 2024], Mistral AI is a French company founded in April 2023 by former employees of Meta Platforms and Google DeepMind. It has quickly gained prominence in the AI sector. The company focuses on developing open-source large language models, emphasizing the importance of free software as an alternative to proprietary solutions.

Mistral 7B (2023): Mistral 7B is a language model comprising 7 billion parameters. It is based on the Transformers architecture. It incorporates grouped-query attention (GQA) to enhance inference speed and utilizes sliding window attention (SWA) to efficiently manage sequences of varying lengths while minimizing inference costs. Additionally, there is a fine-tuned variant, Mistral 7B – Instruct, designed specifically for instruction-

following tasks. Both models are available under the Apache 2.0 license. [Jiang et al., 2023]

Mixtral 8x7B (2023): Mixtral 8x7B is a Sparse Mixture of Experts (SMoE) language model based on the Mistral 7B architecture. Each layer consists of eight "experts," with a router network selecting two to process each token. A fine-tuned variant, Mixtral 8x7B – Instruct, is also available. Both models are released under the Apache 2.0 license. [Jiang et al., 2024]

Mixtral 8x22B (2024): Mixtral 8x22B is similar to Mixtral 8x7b in its architecture. Released under the Apache 2.0 license.

Codestral 22B (2024): Codestral 22B is Mistral's first code focused open weight model. It is a lightweight model specifically built for code generation tasks. Codestral is licensed under the Mistral AI Non-Production License.

Mathstral 7B (2024): Mathstral 7B is a 7-billion-parameter model specifically designed to enhance the solving of advanced mathematical problems that require complex, multi-step logical reasoning. Developed in collaboration with Project Numina, Mathstral 7B is available under the Apache 2.0 license.

Codestral Mamba 7B (2024): Codestral Mamba 7B is built on the Mamba 2 architecture, providing linear time inference and the theoretical capability to model sequences of infinite length. Unlike the base Codestral model, only the instruct version of Codestral Mamba 7B was released, under the Apache 2.0 license.

NeMo 12B (2024): NeMo 12, developed in collaboration between Mistral AI and NVIDIA, features an architecture similar to that of Mistral 7B. The model is available under the Apache 2.0 license.

The company also develops proprietary models, which are listed here for completeness. These include **Mistral Small (2024)**, **Mistral Medium (2024)**, **Mistral Large (2024)**, and **Mistral Large 2 (2024)**.

In selecting models for testing, I opted for those available on Ollama that were both small enough to run on the two GPUs at my disposal and relevant to the research objectives. The models chosen were Mistral 7B, Mixtral 8x7B (Q3-K-L version), and Codestral 22B.

6.2 The Llama Family

Llama (Large Language Model Meta AI) is a family of autoregressive LLMs released by Meta AI. [Wikipedia, 2024b]

LLaMA-1 (2023): LLaMA is a suite of LLMs built on the Transformer architecture, with parameter sizes ranging from 7 billion to 65 billion. Although the models are released under a non-commercial license, access to the model weights is restricted and managed through an application process.

Llama2 (2023): Llama 2 is a suite of LLMs developed by Meta AI in collaboration with Microsoft. The models are available in three sizes—7, 13, and 70 billion parameters—and come in both foundation and chat versions. The architecture of Llama 2 is similar to that of the original LLaMA models with increased context length and grouped-query attention (GQA) [Touvron et al., 2023b]. These models are released under the Llama 2 Community License Agreement (L2CLA), which provides access to the model weights and allows for use in many commercial applications, albeit with some restrictions. Additionally, Meta AI released Code Llama, a variant of Llama 2 fine-tuned on code-specific datasets. Code Llama is available in 7B, 13B, and 34B parameter versions.

Llama3 and Llama3.1 (2024): Llama 3 is a suite of LLMs available in two sizes: 8 billion and 70 billion parameters. These models were released in both text and instruct versions. The models are released under the Meta Llama 3 Community License Agreement (ML3CLA). This license allows the release of model weights and permits usage in many commercial applications, albeit with certain restrictions. Llama 3.1, an updated suite of LLMs, expands the range with three sizes: 8 billion, 70 billion, and 405 billion parameters. Like Llama 3, these models are available in both text and instruct versions. The models are released under the Meta Llama 3.1 Community License Agreement (ML3.1CLA). This license allows the release of model weights and permits usage in many commercial applications, albeit with certain restrictions. Both Llama 3 and Llama 3.1 are based on the Transformer architecture [Dubey et al., 2024]. and are released under the Meta Llama 3 Community License Agreement (ML3CLA). This license allows the release of model weights and permits usage in many commercial applications, albeit with certain restrictions.

In selecting models for testing, I opted for those available on Ollama that were both small enough to run on the two GPUs at my disposal and relevant to the research objectives. The models chosen were **Llama2 7b**, **Llama2 13b**, **Llama3 8b** and **Llama3.1 8b**.

6.3 The Gemma Family

Gemma is a family of free and open-source multimodal LLMs released by Google DeepMind. Google DeepMind first released their proprietary suite Gemini. [Wikipedia, 2024a]

Gemma 1 (2024): Gemma 1, commonly referred to as Gemma, is a lightweight suite of LLMs available in two sizes: 2 billion and 7 billion parameters. These models are based on the Transformer architecture [Team et al., 2024b] and are distributed under the Gemma Terms of Use. Additionally, there are two code generation versions, known as CodeGemma, which are fine-tuned models also available in 2 billion and 7 billion parameter sizes.

Gemma 2 (2024): Gemma2 is a suite of lightweight and open models that range in scale from 2 billion to 27 billion parameters. Like the models from Gemma 1, they are based on the Transformer architecture but incorporate additional techniques such as interleaving local-global attentions and group-query attention [Team et al., 2024a]. Notably, the 2 billion and 9 billion parameter models were trained using knowledge distillation rather

than next-token prediction. These models are available under the Gemma Terms of Use.

RecurrentGemma (2024): RecurrentGemma is an open language model that utilizes the Griffin architecture rather than the traditional Transformer architecture. The Griffin architecture combines linear recurrences with local attention mechanisms [Botev et al., 2024]. This model has 2.7 billion parameters and is available under an Apache 2.0 license.

PaliGemma (2024): PaliGemma is an open Vision-Language Model (VLM) that integrates a vision encoder with the Gemma-2B language model [Beyer et al., 2024]. This model is available under an Apache 2.0 license.

In selecting models for testing, I opted for those available on Ollama that were both small enough to run on the two GPUs at my disposal and relevant to the research objectives. The models chosen were **Gemma 2b**, **Gemma 7b**, **Gemma2 2b**, **Gemma2 9b** and **Gemma2 27b**.

6.4 The Phi Family

Phi is a family of small-sized multimodal LLMs developed and released by Microsoft. These models are free, open-source, and based on the Transformer architecture.

Phi-1 (2023): Phi-1 is a suite of Transformer-based large language models designed for code generation. The models are trained on "textbook quality" data and fine-tuned on a dataset of coding exercises. Phi-1 includes a 1.3 billion parameter model, as well as Phi-1-small, a smaller variant with 350 million parameters. Both models are available under the MIT License. [Gunasekar et al., 2023]

Phi-1.5 (2023): Phi-1.5 is a 1.3 billion parameter model with a similar architecture to Phi-1, also available under the MIT License. [Li et al., 2023]

Phi-2 (2023): Phi-2 is a 2.7 billion-parameter language model, similar in architecture to Phi-1 and Phi-1.5. It is also available under the MIT License. [Hughes, 2024]

Phi-3 (2024): Phi-3 is a suite of LLMs offered in various sizes: phi-3-mini (3.8 billion parameters), phi-3-small (7 billion parameters), and phi-3-medium (14 billion parameters). These models are based on the Transformer architecture and are distributed under the MIT License. The suite also includes Phi-3-Vision, a 4.2 billion parameter model based on Phi-3-mini, with strong reasoning capabilities for both image and text prompts. [Abdin et al., 2024]

In selecting models for testing, I opted for those available on Ollama that were both small enough to run on the two GPUs at my disposal and relevant to the research objectives. The models chosen were **Phi3 3.8b** and **Phi3 14b**.

6.5 The Qwen Family

Qwen is a multimodal LLM family developed and released by Alibaba Group. It includes a range of open-source models in various sizes.

Qwen (2023): QWEN is a comprehensive language model series featuring models with varying parameter counts: 1.8 billion, 7 billion, 14 billion, and 72 billion. These models are built on a modified Transformer architecture similar to Llama-1. The series includes QWEN, the base pretrained models, and QWEN-CHAT, which are fine-tuned for chat applications using human alignment techniques. It also offers specialized models such as CODE-QWEN and CODE-QWEN-CHAT for coding tasks, MATH-QWEN-CHAT for mathematics, and QWEN-VL and QWEN-VL-CHAT, which are capable of processing both visual and language instructions. All models are released under the Tongyi Qianwen Research License Agreement. [Bai et al., 2023] [Team, 2024a]

Qwen1.5 (2024): Qwen1.5 is a comprehensive language model series that includes open-source base and chat models in various sizes: 0.5 billion, 1.8 billion, 4 billion, 7 billion, 14 billion, 32 billion, 72 billion, and 110 billion parameters. Like the previous models, they are based on the Transformer architecture. The series also features a Mixture of Experts (MoE) model, Qwen1.5-MoE-A2.7B. All models are released under the Tongyi Qianwen Research License Agreement (TQRLA). [Team, 2024b]

Qwen2 (2024): The Qwen2 series includes large language models and multimodal models, offering a comprehensive suite of foundational and instruction-tuned models based on the Transformer architecture. These models range from 0.5 to 72 billion parameters and include both dense models and a Mixture-of-Experts model. The series is primarily released under the Apache 2.0 license, with the exception of the 72 billion parameter model, which is distributed under the Tongyi Qianwen Research License Agreement. [Yang et al., 2024]

In selecting models for testing, I opted for those available on Ollama that were both small enough to run on the two GPUs at my disposal and relevant to the research objectives. The models chosen were **Qwen1.5 32b** and **Qwen2 7b**.

Model	License	Parameters (active)	Model Size
Mistral 7b	Apache 2.0	7B	4.1GB
Codestral 22b	MNPL	22B	13GB
Mixtral 8x7b Q3-K-L	Apache 2.0	47B (13B)	20GB
Llama2 7b	L2CLA	7B	3.8GB
Llama2 13b	L2CLA	13B	7.4GB
Llama3 8b	ML3CLA	8B	4.7GB
Llama3.1 8b	ML3.1CLA	8B	4.7GB
Gemma 2b	Gemma Terms of Use	2B	1.7GB
Gemma 7b	Gemma Terms of Use	7B	5.0GB
Gemma2 2b	Gemma Terms of Use	2B	1.7GB
Gemma2 9b	Gemma Terms of Use	9B	5.4GB
Gemma2 27b	Gemma Terms of Use	27B	16GB
Phi3 3.8b	MIT	3.8B	2.2GB
Phi3 14b	MIT	14B	7.9GB
Qwen1.5 32b	TQRLA	32B	18GB
Qwen2 7b	Apache 2.0	7B	4.4GB

Table 6.1: Characteristics of the selected models

Chapter 7
Metrics

In this chapter, the metrics employed in the experiments are outlined according to the two distinct tasks performed. The first section addresses the evaluation metrics for the classification task, focusing on precision, recall, F1 score, and other relevant measures. The second section, on the other hand, details the metrics used for the text generation task, specifically discussing the BLEU and ROUGE scores.

7.1 Classification metrics

When working a classification task, we want to evaluate the quality of the classification. In order to do so we rely on a test set that is labeled for the different features we are working on. Once the classifier has predicted values for the test set, the following metrics can be used to evaluate the quality of the classifier.

For every feature, a test sample has two values associated with it:

True value: The true value is the real value for the feature, unknown to the classifier;

Predicted value: The predicted value is the value predicted by the classifier for this feature.

Because we are working with boolean features here the values will be *true* or *false*.

It is common to fill a confusion matrix with the true values represented by the columns and the predicted values represented by the rows. We give the following names to the four cells of the confusion matrix:

True positive (TP): A true positive is a sample that the classifier correctly classified for the boolean feature as true;

False negative (FN): A false negative is a sample that the classifier incorrectly classified for the boolean value as false when it was actually true;

False positive (FP): A false positive is a sample that the classifier incorrectly classified for the boolean value as true when they were actually false;

True negative (TN): A true negative is a sample that the classifier correctly classified for the boolean feature as false.

Building on the previous measures—True Positive, False Negative, False Positive, and True Negative—we derive the following classic metrics. Unlike the absolute nature of the previous measures, which count the occurrences of specific outcomes, these metrics are relative. They quantify the classifier's performance on a scale from 0 to 1, reflecting its ability to achieve the classification task on various aspects.

Accuracy: The accuracy represents the fraction of correctly classified samples for a given feature. It is a measures how well the classifier is performing on a high level. Accuracy is computed as follows.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}$$

Positive Predictive Value (PPV) / Precision: The precision, also known as positive predictive value, represents the fraction of correctly classified samples for a given feature among the samples classified as positive / true. It measures how confident one can be in the result of the classifier when the sample is classified as positive / true. It is computed as follows.

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Negative Predictive Value (NPV): The negative predictive value represents the fraction of correctly classified samples for a given feature among the samples classified as negative / false. It measures how confident one can be in the result of the classifier when the sample is classified as negative / false. It is computed as follows.

$$\text{NPV} = \frac{\text{TN}}{\text{TN} + \text{FN}}$$

Sensitivity / Recall: The recall, also known as sensitivity, represents the fraction of correctly classified samples for a given feature among the samples classified as positive / true. It measures how confident one can be that the samples that actually are positive / true will be classified as so. It is computed as follows.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Specificity: The specificity represents the fraction of correctly classified samples for a given feature among the samples classified as negative / false. It measures how confident one can be that the samples that actually are negative / false will be classified as so. It is computed as follows.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

F1 score: The F1 score is a high-level metric for evaluating predictive performance in binary classification, effectively combining precision and recall into a single measure. As the harmonic mean of precision and recall, it provides a balanced assessment of a model's ability to distinguish between true positive and false negative samples. It is computed as follows.

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}.$$

7.2 Generation metrics

To evaluate the quality of text generated by language models, it is common practice to measure the similarity between the generated text and a reference target. In this section, we will focus on two classic similarity metrics: the BLEU and ROUGE scores.

7.2.1 BLEU Score

The BLEU (for Bilingual Evaluation Understudy) score is a historic metric to asses the quality of translation models. It was introduced in 2002 by Papineni et al. [2002]. Its goal is to provide a decent evaluation of a translation model so that there is no need to do a human evaluation as human evaluations are costly and cannot be reused.

On a high level the BLEU score basically compares a source sentence against one or more target sentences and measures statistically how close the source sentence is to the target sentences. The bleu score tends to categorize sentences as close when they have the same length, use the same words and have the same fluency.

To do so BLEU score combines several algorithms and measures:

- Short sentences are penalized by a brevity penalty factor
- Long sentences are penalized because they will have fewer matches on their n-grams for low values of n due to the exhaustion of reference n-grams
- A source sentence that have a fluency similar to the target sentence will have a higher score due to more matches on the n-grams for higher values of n.

The BLEU score is advantageous due to its quick computation, low cost, and language independence, and it often aligns well with human judgment. However, it has high variance, making it less reliable for small corpora and most effective when used with larger datasets where its results are more stable and consistent.

The Brevity Penalty

The brevity penalty is introduced to avoid short sentences to have high scores because their n-grams are more likely to match.

Let's denote C the candidate sentence and R the reference sentence. Let's denote c the number of words of the candidate sentence and r the number of words of the reference sentence. Then the brevity penalty is computed as follows.

$$BP(C,T) = \begin{cases} 1 & \text{if } c > r \\ \exp(1 - \frac{r}{c}) & \text{if } c \leq r \end{cases}$$

If there are multiple target sentences then the sentence which has its length closest to the length of the candidate is used.

If the BLEU score is measured on a corpus made of several sentences then the brevity penalty can be computed at the corpus level instead of the sentence level in order to give more freedom for the model at the sentence level. Therefore the length of the candidate corpus and the length of the reference corpus are compared.

Modified N-gram Precision

An n-gram is a sequence of n adjacent words in particular order. The basic n-gram precision is the proportion of n-grams in the candidate sentence (with multiplicity) that match any n-gram of the target sentence.

The issue with the basic n-gram precision is that it gives higher scores to candidates using overly very common words. For example, in the following example, every single occurrence of 'the' in the candidate sentence matches the single occurrence of 'the' in the reference sentence leading to a perfect 1-gram precision of 1.

Candidate sentence: The the the the.

Reference sentence: The man is eating an apple.

To solve this issue [Papineni et al., 2002] introduced the modified n-gram precision denoted p_n. It adds the notion of exhaustion of reference n-grams. This means that an n-gram of the reference sentence can match at most one occurrence of the n-gram in the candidate sentence. In the previous example, it means that only one of the four 'the' has a match as 'the' appears only once in the reference sentence and is exhausted after the first match. This means the modified precision score for 1-grams has the following value.

$$p_1 = \frac{1}{4} = 0.25$$

Computing the BLEU Score

Based on the previous sections the BLEU score is computed as follows.

$$BLEU = BP \times \exp(\sum_{n \in [\![1;N]\!]} \omega_n log(p_n))$$

where:

- N is the maximum length of the n-gram that are being considered, Papineni et al. [2002] uses $N = 4$
- ω_n are the weights for the weighted arithmetic mean of the modified n-gram precision, Papineni et al. [2002] uses a uniform distribution (ie. $\omega_n = \frac{1}{N}$)

1-gram	count in source	count in target	number of matches
the	2	1	1
man	1	1	1
eats	1	0	0
apple	1	1	1
total	**5**	N/A	**3**

Table 7.1: 1-gram counts in the example

2-gram	count in source	count in target	number of matches
the man	1	1	1
man eats	1	0	0
eats the	1	0	0
the apple	1	0	0
total	**4**	N/A	**1**

Table 7.2: 2-gram counts in the example

Example

In this example we use $N = 2$ and $\omega_n = \frac{1}{N}$. Considering the following sentences:

Source sentence: The man eats the apple.

Target sentence: The man is eating an apple.

We can compute the Tables 7.1 and 7.2 and get the following modified precision values.

$$p_1 = \frac{3}{5} = 0.6$$

$$p_2 = \frac{1}{4} = 0.25$$

We can also compute the brevity penalty value. Because the source sentence is shorter than the target sentence we get the following value.

$$BP = \exp(1 - \frac{6}{5}) = 0.82$$

Finally we get the value for the BLEU score.

$$BLEU = BP \times \exp \sum \omega_n log(p_n) = 0.32$$

Chosen BLEU Implementation

To calculate the blue scores, I relied on the *sentence_bleu* function from the Python package *nltk.translate.bleu_score*.

7.2.2 ROUGE Score

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is a measure that was introduced in [Lin, 2004]. It aims to evaluate the quality of machine generated summaries. It was also used in translation evaluation. The ROUGE score has several strength such as being quick and inexpensive to compute, language independent and it correlates to human judgement. However just like the BLEU score, it has a high variance and is therefore relevant on big corpus. The article introduces several types of ROUGE scores that the next paragraphs will go through.

ROUGE-N

Computing a ROUGE-N score: When comparing a candidate and a reference sentence ROUGE-N basically measures the fraction of the n-grams of the reference text matches an n-gram of the candidate text.

$$\frac{\sum_{\text{gram}_n \in R} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{\text{gram}_n \in R} \text{Count}(\text{gram}_n)}$$

where:

- gram_n represent an n-gram;
- $\boldsymbol{R}$ is the set of all unique n-grams for the reference sentence;
- $\text{Count}_{\text{match}}$ computes the number of matches between the reference sentence and the candidate sentence for a given n-gram and using exhaustion on the candidate text n-grams;
- Count computes the number of occurrences of a given n-gram in the reference text.

Example: In this example, the following source and target sentences will be used.

Source sentence: The man eats the apple.

Target sentence: The man is eating an apple.

In this example, 3 out of 6 of the unigram of the target sentence match a unigram in the source sentence ("The", "man" and "apple") so the ROUGE-1 score is $\frac{3}{6} = 0.5$. Only 1 out of the 5 bigrams of the target sentence match a unigram in the source sentence ("The man") so the ROUGE-2 score is $\frac{1}{5} = 0.2$.

Strength and weaknesses: One of the primary strengths of ROUGE-N lies in its simplicity and ease of implementation. By measuring the overlap of n-grams (sequences of words) between a candidate summary and reference summaries, ROUGE-N provides a straightforward method for evaluating summarizing tasks. This simplicity has contributed to its widespread adoption as a popular metric in the field of automatic summarizing.

However, ROUGE-N also has several notable drawbacks. The metric is highly sensitive to the exact wording and structure of the text. Consequently, a candidate summary

that accurately captures the meaning of the reference summary but employs different wording or phrasing may still receive a low ROUGE-N score. Additionally, because ROUGE-N is recall-based, it tends to favor longer summaries that include a greater number of n-grams from the reference summaries. This bias may result in higher scores for summaries that are less concise or focused, potentially overemphasizing the inclusion of less important or relevant information.

ROUGE-L

Computing a ROUGE-L score: When comparing a candidate sentence to a reference sentence, ROUGE-L primarily measures the fraction of the texts covered by the Longest Common Subsequence (LCS). The LCS is defined as the longest sequence that can be derived from both texts by removing some or none of the elements, without altering the order of the remaining elements. The ROUGE-L score is calculated by dividing the length of the LCS by the length of one of the texts. Choosing the reference text as the denominator yields a recall measure, while selecting the candidate text results in a precision measure.

$$\frac{\text{Length of the LCS}}{\text{Length of one of the text}}$$

Example: In this example, the following source and target sentences will be used.

Source sentence: The man eats the apple.

Target sentence: The man is eating an apple.

The LCS is *The man apple* and has a length of 3. From there a precision measure can be derived by dividing the length of the LCS by the length of the source sentence: $P_{LCS} = \frac{3}{5} = 0.6$. A recall measure can also be derived by dividing the length of the LCS by the length of the target sentence: $R_{LCS} = \frac{3}{6} = 0.5$. An F-score can also be derived from the previous measures: $F_{1,LCS} = 2\frac{0.5 \times 0.6}{0.5+0.6} = 0.55$.

Strength and weaknesses: ROUGE-L has several strengths that make it a valuable metric in text evaluation. One of its primary advantages is its ability to capture sentence-level structure by focusing on in-sequence matches, which reflect the word order within sentences. This sensitivity to structure and flow differentiates ROUGE-L from n-gram-based metrics, which do not consider word order as strictly. Additionally, ROUGE-L automatically includes the longest in-sequence common n-grams, eliminating the need for a predefined n-gram length. This flexibility allows it to effectively capture varying lengths of matching sequences.

However, ROUGE-L has several notable limitations. Firstly, it considers only the LCS, which may overlook shorter but still meaningful subsequences. This emphasis on longer sequences can introduce a bias, potentially underestimating the significance of shorter, yet important, n-grams or word matches that contribute to the overall meaning of the text. Secondly, ROUGE-L does not differentiate between LCSs of the same length based on their consecutive matches, even though consecutive matches should be favored as they are more indicative of meaningful content alignment.

ROUGE-W: To address this limitation, the paper introduces ROUGE-W, where "W" stands for *Weighted LCS*. ROUGE-W builds on the ROUGE-L framework by incorporating consecutive matches, which are assigned greater importance. This enhancement allows ROUGE-W to better capture the significance of contiguous sequences, thus providing a more nuanced evaluation of text similarity.

ROUGE-S

Computing a ROUGE-S score: Computing a ROUGE-S score is similar to computing a ROUGE-N score but instead of using n-grams, skip-n-grams are used. A skip-n-gram is any sequence of words in their sentence order, allowing for arbitrary gaps. The ROUGE-S score is then computed by dividing the number of matches between the skip-n-grams of the source and target texts by the number of skip-n-gram in one of the texts. Choosing the reference text as the denominator yields a recall measure, while selecting the candidate text results in a precision measure.

$$\frac{\text{Number of matching skip-n-grams}}{\text{Number of skip-n-grams of one of the texts}}$$

Example: In this example, the following source and target sentences will be used. Skip-bigrams will be used.

Source sentence: The man eats the apple.

Target sentence: The man is eating an apple.

The different matching skip-bigrams are “the man”, “the apple”, and “man apple”. From there a precision measure can be derived by dividing the number of matches by the number of skip-bigrams of the source sentence: $P_{skip2} = \frac{3}{10} = 0.3$. A recall measure can also be derived by dividing the number of matches by the number of skip-bigrams of the target sentence: $R_{skip2} = \frac{3}{15} = 0.2$. Finally an F-score can be derived from the previous values $F_{1,skip2} = 2\frac{0.3 \times 0.2}{0.3+0.2} = 0.24$.

Strengths and weaknesses: ROUGE-S offers notable advantages in evaluating translation quality by retaining sensitivity to word order while allowing for gaps between words. This flexibility provides a more nuanced assessment compared to metrics that require exact, consecutive matches. However, ROUGE-S also has its drawbacks. It can count spurious matches, such as non-informative pairs like “the the” or “of in,” due to its lack of initial restrictions on word distance. This can lead to inflated scores. It also does not differentiate candidates that have no unigram matches from those who have.

Refinements: Refinements are proposed to the ROUGE-S score in the paper.

Introducing a maximal skip distance: A refinement parameter d_{skip} can be introduced to help control the distance between words in skip-n-grams to reduce spurious matches. By setting d_{skip}, users can limit how far apart words can be to form valid skip-n-grams, improving the accuracy of the metric.

ROUGE-SU: ROUGE-SU is an enhanced version of ROUGE-S designed to address a key limitation of the original metric. While ROUGE-S can assign a score of zero when no skip-bigram matches are found between a candidate and reference sentence, ROUGE-SU incorporates unigram counts to overcome this issue. By introducing a begin-of-sentence marker and considering individual word occurrences, ROUGE-SU captures partial similarities and word overlaps that ROUGE-S would miss.

Chosen ROUGE Implementation

To calculate the red scores, I chose the ROUGE-N score with bigrams and used the Python module *red_scorer* for that.

Chapter 8
Prompt Engineering

LLMs are designed to respond to prompts—queries or instructions formulated in natural language. The accuracy and relevance of the responses provided by these models are significantly influenced by the specific wording and structure of the prompts. This variability in performance highlights a critical challenge: crafting prompts that consistently yield precise and reliable answers.

Prompt engineering aims to address this issue by developing systematic techniques and best practices for constructing prompts that optimize the model's output quality, thereby enhancing the overall utility and dependability of LLMs in practical applications. In this chapter, I outline the the literature regarding techniques and best practices in prompt engineering.

8.1 Content

In a literature review, Zhao et al. [2023] highlight three crucial components that should be incorporated into a prompt to enhance the performance of LLMs. These components are: a task description, input data, and contextual information. By including these elements, prompts can be better tailored to elicit more accurate and contextually appropriate responses from LLMs.

8.1.1 Task Description

Task descriptions provide instructions that LLMs are expected to follow. To maximize the chances of obtaining the desired output from an LLM, one should clearly define the goal and avoid using vague or ambiguous terms [Santu and Feng, 2023]. For complex tasks, it is advisable to break them down into simpler sub-tasks, organizing them with bullet points or numbered items. This approach aids the LLM in identifying and executing each individual sub-task [Santu and Feng, 2023].

8.1.2 Input Data

Some prompts that do not provide input data rely on the LLM's background knowledge to complete their tasks. To address more specific tasks, one can include relevant data that the model should use to solve the problem. It is advisable to clearly state whether such data is provided and to separate this data from the task description [Santu and Feng, 2023].

8.1.3 Contextual Information

Contextual information refers to supplementary data that, although not the input data, can aid the LLM in accurately solving the given task. This includes additional sources obtained through Information Retrieval Techniques, enabling the LLM to utilize up-to-date information. Additionally, contextual information includes examples of (input-desired output) pairs for the task, known as few-shot examples [Santu and Feng, 2023].

8.2 The TELeR Prompt Taxonomy

An LLM prompt taxonomy is a structured classification system that categorizes various types of prompts used to interact with LLMs. Such a taxonomy organizes prompts based on their characteristics, and aims to establish a standard that facilitates the evaluation and comparison of LLM performance across different categories of prompts. Such comparisons will help draw meaningful conclusions about the performances of LLMs.

The TELeR prompt taxonomy [Santu and Feng, 2023] propose a classification for complex task prompts based on 4 dimensions:

1. **Turn**: This dimension is based on the number of interactions needed to complete a task. Prompts can be:
    - Single-turn: The task is completed in one prompt and response cycle
    - Multi-turn: The task requires multiple prompt and response cycles
2. **Expression**: This refers to the style in which the prompts are expressed. Prompts can be:
    - Question-style: The prompt is framed as a question
    - Instruction-style: The prompt is framed as an instruction or command
3. **Role**: This dimension considers whether a specific role is defined for the LLM before prompting. Prompts can be:
    - System-role defined: A role is specified for the LLM via system prompt
    - Undefined: No specific role is assigned
4. **Level of Detail**: This dimension categorizes prompts based on the amount of detail provided. There are seven levels (0-6), with:
    - Level 0: No directive, Just Data

- Level 1: Simple one-sentence directive expressing the high-level goal
- Level 2: Multi-sentence (paragraph-style) directive expressing the high-level goal and the sub-tasks that need to be performed to achieve the goal
- Level 3: Complex (bulleted-list-style) directive expressing the high-level goal along with a detailed bulleted list of sub-tasks to be performed
- Level 4: Level 3 + guideline on how LLM output will be evaluated/Few-Shot Examples
- Level 5: Level 4 + additional relevant information gathered via retrieval-based techniques
- Level 6: Level 5 + an explicit statement asking LLM to explain its own output

8.3 Prompt Writing

Building upon the best practices outlined in the first section and utilizing the taxonomy introduced in the second section, I developed my prompts through a series of successive components.

System Role: I began by defining a system role for the language model to direct it toward generating more accurate responses.

Task description: Next, I outlined the expected task in general terms and specified the output format when necessary.

Examples: Then, depending on the context, I provided between 0 and 6 examples of expected input and output.

Input data: Finally, I provided the input text on which the task was to be performed.

Examples of prompts I used are available in Appendix A.

Chapter 9
Results and Discussions

This chapter describes the results I obtained in my two tasks: **text classification** and **text enhancing**. The first section goes through the results in the text classification task. The second section goes through the text enhancing task.

9.1 Classification Task

This section in separated in four subsections, each one of them having their own goal:

- **Model comparison**: the *Model comparison* section aims to compare a wide range of models on the classification task by evaluating them on the same prompt type, same instruction language and same label, as a result only a few model will be selected as candidates to be used in the following sections;
- **Language comparison**: the *Language comparison* section aims to quantify the impact of the instruction language that is chosen between English and French;
- **Prompt type comparison**: the *Prompt type comparison* section aims to quantify the impact of the prompt type on the classification task using the TELeR prompt taxonomy;
- **Label comparison**: the *Label comparison* aims to compare the results gotten on the different labels.

I am using the following notation:

- **acc**: accuracy of the model of the classification task;
- **P+**: Precision on the positive class;
- **R+**: Recall on the positive class;

Model	acc	P+	R+	f_1	P-	R-	c.rate	avg time
Mistral 7b	0.73	0.47	0.90	0.62	0.95	0.67	**1.00**	0.45
Codestral	0.82	0.80	0.38	0.52	0.82	0.97	**1.00**	1.32
Mixtral 8x7b Q3	0.80	0.75	0.29	0.41	0.80	0.97	**1.00**	6.43
Llama2 7b	0.77	1.00	0.10	0.17	0.77	1.00	**1.00**	1.99
Llama2 13b	0.45	0.31	1.00	0.48	1.00	0.27	**1.00**	1.09
Llama3 8b	0.77	0.53	0.76	0.63	0.91	0.77	0.99	2.48
Llama3.1 8b	0.79	0.56	0.67	0.61	0.88	0.83	**1.00**	0.41
Gemma 2b	0.39	0.24	0.67	0.35	0.73	0.30	**1.00**	0.24
Gemma 7b	0.76	0.56	0.24	0.33	0.79	0.94	**1.00**	0.46
Gemma2 2b	0.81	0.73	0.38	0.50	0.82	0.95	**1.00**	2.52
Gemma2 9b	0.76	0.51	1.00	0.68	1.00	0.68	**1.00**	1.09
Gemma2 27b	0.80	0.56	0.95	0.70	0.98	0.75	**1.00**	9.43
Phi3 3.8b	0.50	N/A	0.00	0.00	0.50	1.00	0.05	N/A
Phi3 14b	0.66	0.41	0.81	0.55	0.90	0.61	0.99	1.15
Qwen2 7b	**0.86**	0.71	0.71	**0.71**	0.90	0.90	**1.00**	0.38
Qwen1.5 32b	0.85	0.70	0.67	0.68	0.89	0.90	**1.00**	4.92

Table 9.1: Result per model

- f_1: f_1 score on the positive class;
- **P-**: Precision on the negative class
- **R-**: Recall on the negative class
- **avg time**: average time needed to generate the answer.

9.1.1 Model Comparison

In this section, the different models are tested on the same specific prompt that aims to specify a unique label. The prompt can be found in Appendix A.1.1. The full results can be found in Table 9.1.

It can be seen that most of the models have a classification rate of 1 which means a vast majority of the responses from the models were successfully parsed thanks to the parsing strategy. One exception to this observation is the Phi3 3.8b model which had a classification rate of **0.05** due to unexpectedly formatted responses. This model will be counted out during further analyses.

Looking at the relation between the f_1 score and the average generation time of the response in Figure 9.1 it can be seen that one of the model out performs the others: Qwen2 7b. It had the best f_1 score out of all the models with a score of **0.71**. It also had the best accuracy of all the models with a value of **0.86**. This model is a medium sized model with its 7 billion parameters but it surprisingly performs evenly with the bigger models such as Gemma2 27b and Qwen1.5 32b. It also out performs the best models of the same size such as Mistral 7b and Llama3.1 8b.

Regarding those results and due to time constraint the rest of Section 9.1 will focus on the following 4, medium size, well performing, models:

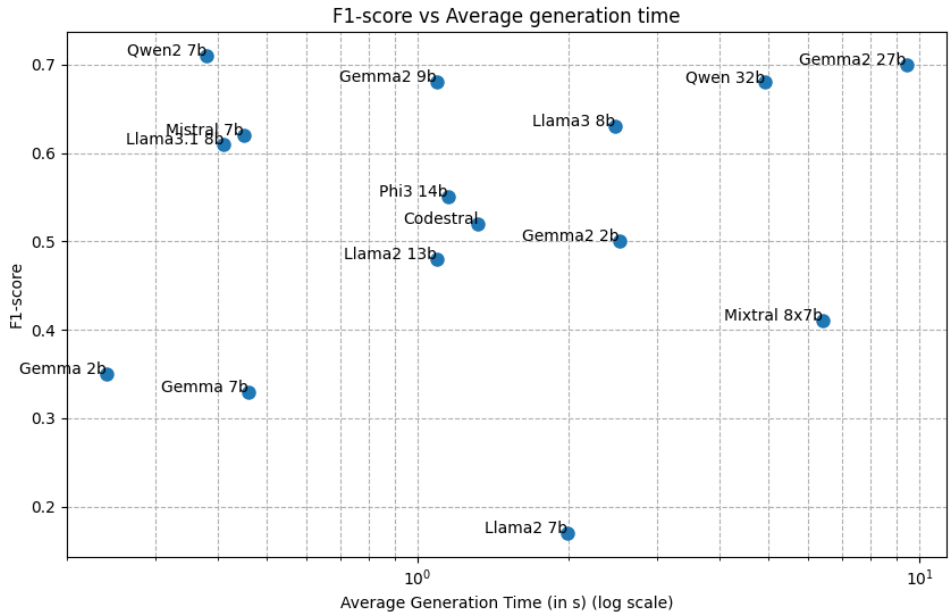

Figure 9.1: F1-score vs average generation time

- Qwen2 7b
- Mistral 7b
- Llama3.1 8b
- Gemma2 9b

9.1.2 Results per Language

The results in this section aims to decided whether the language in which the instruction are given (French or English) has an impact on the quality of the classification task. As a reminder the 4 models chosen in Section 9.1.1 (Qwen2 7b, Mistral 7b, Llama3.1 8b and Gemma2 9b) have been evaluated on the two following prompts.

1. The first prompt is the same prompt that was used in Section 9.1.1. It is a 2-shot prompt with instructions given in French. The prompt can be found in Appendix A.1.1.

2. The second prompt is the equivalent of the first one with the instructions translated into English. The prompt can be found in Appendix A.1.2

The results per models and the average values per instruction language can be found in Table 9.2. On average it seems that giving instructions in French gives a better accuracy, f_1 score and computational time. The classification rate seems to stays the same. However there are some differences based on the model that help to put things in perspective.

When looking in depth at the average accuracy it can be seen that most of the models achieve comparable accuracy given the size of the test set. Qwen2 7b achieves an 0.87 accuracy

Language	model	acc	P+	R+	f_1	P-	R-	c.rate	avg time
	Gemma2 9b	0.75	0.50	1.00	0.67	1.00	0.67	1.00	1.33
	Llama3.1 8b	0.82	0.67	0.57	0.62	0.86	0.90	1.00	0.72
French	Mistral 7b	0.73	0.41	0.37	0.39	0.81	0.84	0.96	0.80
	Qwen2 7b	0.87	0.75	0.71	0.73	0.91	0.92	1.00	0.65
	Average	**0.79**			**0.60**			**0.99**	**0.85**
	Gemma2 9b	0.80	0.55	1.00	0.71	1.00	0.73	1.00	1.74
	Llama3.1 8b	0.75	0.50	0.38	0.43	0.81	0.87	1.00	0.72
English	Mistral 7b	0.59	0.29	0.56	0.38	0.82	0.60	0.95	0.77
	Qwen2 7b	0.85	1.00	0.38	0.55	0.83	1.00	1.00	0.60
	Average	**0.75**			**0.52**			**0.99**	**0.96**

Table 9.2: Results per language

in French against an 0.85 accuracy in English. Gemma2 9b Llama3.1 8b achieves a slightly better accuracy in French with a value of 0.82 against an 0.75 accuracy in English. Finally achieves a slightly better accuracy in English with a value of 0.80 against an 0.75 accuracy in French. Only Mistral 7b stands out with a significantly better accuracy in French than in English. However Mistral 7b achieves the same f_1 score regardless of the instruction language. This means that, when given instructions in French the model achieved a better accuracy because it tended to classify most of the samples as being from the main (negative) class but was not better at classifying. Therefore, even though giving instructions in French gives on average a better accuracy, it is not sufficient to state that it helps to build a better classifier.

When looking in depth at the f_1 score it can be seen that one half of the models achieve comparable f_1 scores regardless of the instruction language whereas the other half achieves significantly better scores with French instructions. The two models achieving comparable f_1 scores regardless of the instruction language are Gemma2 9b and Mistral 7b. Mistral 7b achieves a slightly better score with French instruction with a score of 0.39 against a score of 0.38 with English instructions. And Gemma2 9b achieves a slightly better score with English instruction with a value of 0.71 against a score of 0.67 with French instructions. The two other models, Llama3.1 8b and Qwen2 7b, both achieve significantly better scores when instructions are given in French. Llama3.1 8b achieves a score of 0.62 with French instructions against a score of 0.43 with English instructions. Qwen2 7b achieves a score of 0.73 with French instructions against a score of 0.55 with English instructions. Therefore, it can be concluded that giving instructions in French, which - as a reminder - is the language of the texts dynamically inserted in the prompts, helps achieve better f_1 score and therefore better classifiers. In the worst cases the scores are comparable to the scores achieved when the instruction are given in English. This could be further analyzed by taking into account the proportion of French documents used to train the different models.

When looking in depth at the classification rates, it can be seen that only Mistral 7b fails to consistently give a response with a format that could be parsed. However it fails comparably regardless of the instruction language with a classification rate of 0.96 with French instructions and a classification rate of 0.95 with English instructions. Therefore can be concluded that the instruction language has in this case no significant impact on the formatting of the answer.

Finally, looking in depth at the average generation time, it can be seen that most of the

models achieve comparable values but Gemma2 9b. All the models among Llama3.1 8b, Mistral 7b and Qwen2 7b achieve comparable values for the average generation time. Llama3.1 8b achieves an average duration of 0.72s for both languages. Mistral 7b achieves an average generation time of 0.80 with French instructions against an average of 0.77 with English instructions. Finally Qwen2 7b achieves an average generation time of 0.65s with French instructions against an average of 0.60 with English instructions. Gemma2 9b on the contrary achieves a significantly better average generation time with French instruction with a value of 1.33 against an average time of 1.74s with English instructions. However this seems mainly due to the number of generated tokens which values are not given in Table 9.2 but are 42.67 with French instructions against 66.55 with English instructions. This difference could explain the difference in generation time but the origin of it remains unclear since the answers have the same format so should have the exact same number of tokens.

In conclusion the instruction language seems to have a significant and meaningful impact on the f_1 score in favor of the French instructions but not on either of the accuracy, the classification rate and the average generation time. Overall the classifier yields better results if the instructions are given in French.

9.1.3 Results per Prompt Type

The results in this section aims to quantify the impact of the prompt type on the quality of the classification task. As a reminder the 4 models chosen in Section 9.1.1 (Qwen2 7b, Mistral 7b, Llama3.1 8b and Gemma2 9b) have been evaluated on 7 different prompts. The prompts will be described using the TELeR prompt taxonomy described in Section 8.2. The prompts were all Single-turn, Instruction-style, System-role defined prompts. The key difference between the prompts is the Level of Detail of the prompt. The 7 prompts had the following Level of Detail:

1. A level 4 prompt with 6-shot prompting;
2. A level 4 prompt with 5-shot prompting;
3. A level 4 prompt with 4-shot prompting;
4. A level 4 prompt with 3-shot prompting;
5. A level 4 prompt with 2-shot prompting;
6. A level 4 prompt with 1-shot prompting;
7. A Level 3 prompt (Complex directive expressing the high-level goal along with a detailed bulleted list of sub-tasks to be performed) that will be referred as the 0-shot prompt.

The level 4 prompt with 6-shot prompting can be found in Appendix A.1.3 and the other prompts can be obtained by iteratively removing the last examples given in the prompt.

The results per models and the average values per prompt type can be found in Table 9.3. On average it seems that giving 2 to 3 examples gives better f_1 scores and generation times. On average it also looks like using 4 or more examples decreases the classification rate. Finally

Prompt type	model	acc	P+	R+	f_1	P-	R-	c.rate	avg time
	Gemma2 9b	0.77	0.50	0.53	0.51	0.85	0.84	0.98	3.38
	Llama3.1 8b	0.66	0.42	0.88	0.57	0.94	0.58	0.80	2.11
6-shot	Mistral 7b	0.82	0.57	0.50	0.53	0.87	0.90	0.90	0.91
	Qwen2 7b	0.76	0.50	0.05	0.09	0.77	0.98	0.99	0.77
	Average	**0.75**			**0.43**			**0.92**	**1.79**
	Gemma2 9b	0.51	0.33	0.95	0.49	0.96	0.37	1.00	1.47
	Llama3.1 8b	0.72	0.45	0.68	0.54	0.88	0.74	0.95	2.45
5-shot	Mistral 7b	0.68	0.44	0.81	0.57	0.90	0.63	0.96	1.10
	Qwen2 7b	0.70	0.36	0.25	0.29	0.78	0.85	0.96	0.97
	Average	**0.65**			**0.47**			**0.97**	**1.50**
	Gemma2 9b	0.76	0.51	1.00	0.68	1.00	0.68	0.99	0.91
	Llama3.1 8b	0.77	0.51	0.95	0.67	0.98	0.71	0.98	1.18
4-shot	Mistral 7b	0.65	0.32	0.73	0.44	0.91	0.63	0.68	1.75
	Qwen2 7b	0.73	0.47	0.70	0.56	0.88	0.74	0.98	0.65
	Average	**0.73**			**0.59**			**0.91**	**1.12**
	Gemma2 9b	0.69	0.45	1.00	0.62	1.00	0.59	1.00	0.89
	Llama3.1 8b	0.83	0.63	0.81	0.71	0.93	0.84	1.00	0.44
3-shot	Mistral 7b	0.71	0.46	0.90	0.61	0.95	0.64	0.98	0.96
	Qwen2 7b	0.85	1.00	0.38	0.55	0.83	1.00	1.00	0.40
	Average	**0.77**			**0.62**			**1.00**	**0.67**
	Gemma2 9b	0.77	0.53	1.00	0.69	1.00	0.70	1.00	1.25
	Llama3.1 8b	0.79	0.56	0.67	0.61	0.88	0.83	1.00	0.44
2-shot	Mistral 7b	0.73	0.48	0.95	0.63	0.98	0.65	1.00	0.49
	Qwen2 7b	0.80	0.62	0.48	0.54	0.84	0.90	1.00	0.38
	Average	**0.77**			**0.62**			**1.00**	**0.64**
	Gemma2 9b	0.43	0.30	1.00	0.47	1.00	0.24	1.00	1.80
	Llama3.1 8b	0.63	0.40	0.95	0.56	0.97	0.52	1.00	0.62
1-shot	Mistral 7b	0.67	0.42	0.90	0.58	0.95	0.59	1.00	0.47
	Qwen2 7b	0.86	0.68	0.81	0.74	0.93	0.87	1.00	0.38
	Average	**0.65**			**0.59**			**1.00**	**0.82**
	Gemma2 9b	0.77	0.53	1.00	0.69	1.00	0.70	1.00	1.65
	Llama3.1 8b	0.79	0.64	0.35	0.45	0.82	0.94	0.98	1.40
0-shot	Mistral 7b	0.83	0.73	0.52	0.61	0.86	0.94	1.00	0.43
	Qwen2 7b	0.74	0.00	0.00	0.00	0.75	0.98	1.00	0.36
	Average	**0.78**			**0.44**			**1.00**	**0.96**

Table 9.3: Result per prompt type

it seems that on average the accuracy is not impacted by the number of examples. However there are some differences based on the model that help to put things in perspective.

When looking at the accuracy, there is no clear trend indicating that the prompt type influences the accuracy for any of the models. Gemma2 9b achieved accuracies between 0.43 and 0.77. Llama3.1 8b achieved accuracies between 0.63 and 0.83. Mistral 7b between 0.65 and 0.83. Qwen2 7b achieved accuracies between 0.70 and 0.86. Therefore, it can be concluded that the prompt type has no impact on the accuracy regardless of the model. It can be noted that Gemma2 9b seems to have a worse accuracy than the other models and Qwen2 7b a slightly better one.

When looking at the f_1 score however it can be noted that, on average, there seem to be an optimum number of examples around 2 to 3 examples. Adding or removing more examples result in a significant decrease of the f_1 score. Gemma2 7b achieves its best scores with 0, 2 and 4 examples with scores of respectively 0.69, 0.69 and 0.68. Llama3.1 8b achieves its best score with 3 examples with a score of 0.71. Mistral 7b achieves its best scores with 2 examples with a score of 0.63. Finally Qwen2 7b achieves its best score with 1 example with a score of 0.74 Therefore it can be concluded that the best prompt varies based on the model. It can also be noted that Mistral 7b seems to under-perform compared to the 3 other models. The best f_1 scores are obtained with a number of example ranging from 0 to 4 so it can be concluded that too many examples harm the f_1 score. On average the best f_1 scores are obtained with 2 to 3 examples so this can be the take away to work with new models.

When looking at the classification rate it can be noted that, on average, it decreases past 3 examples. Mistral 7b and Llama3.1 8b seem to be suffering more from the decrease than the two other models. Mistral 7b achieves its lower classification rate with a value of 0.68 with the 4-shot prompt. Llama3.1 8b achieve its lower rate with a value 0.80 with the 6-shot prompt. Gemma2 9b achieves its lowest classification with a value of 0.98 with the 6-shot prompt. Finally Qwen2 7b achieves its lowest classification rate with a value of 0.96 with the 5-shot prompt. Therefore it can be concluded that the decrease in classification rate as the number of examples go up heavily depends on the model and is not a clear decrease for each model but a general rule is that the rate can be kept maximal for prompts with less than 4 examples.

When looking at the average time of generation it can be noted that, on average, there seem to be an optimum value of example to get the fastest responses from the model around 2 to 3 examples. On average the models generate the answer under 0.64s with 2 examples and under 0.67s with 3 examples. Gemma2 9b achieves its lowest average generation time with a value of 0.89s with 3 examples in the prompt. Llama3.1 8b achieves its lowest average generation time with a value of 0.44s with 2 and 3 examples per prompt. Mistral 7b achieves its lowest average generation time with a value of 0.43s with the 0-shot prompt. Finally Qwen2 7b achieves its lowest average generation with a value of 0.36s with the 0-shot prompt. Some models have higher average generation times for 0-shot and 1-shot prompting but it is correlated to a higher number of generated tokens, this remains unexplained as the responses should have the exact same length. Therefore it can be concluded that the fastest generation that is achieved heavily depends on the model but tend to be obtained with a number of example ranging from 0 to 3. On average, the best generation time are obtained with 2 to 3 examples so this can be the take away to work with new models.

In conclusion the prompt type, and more specifically the number of examples given in the prompt seem to have a significant and meaningful impact on the f_1 score, the classifica-

Label	model	acc	P+	R+	f_1	P-	R-	c.rate	avg time
	Gemma2 9b	0.71	0.46	0.81	0.59	0.91	0.68	1.00	1.54
	Llama3.1 8b	0.71	0.41	0.33	0.37	0.79	0.84	0.99	0.66
O_Pe	Mistral 7b	0.56	0.34	0.81	0.48	0.88	0.48	1.00	0.59
	Qwen2 7b	0.76	0.53	0.38	0.44	0.81	0.89	1.00	0.50
	Average	**0.69**			**0.47**			**1.00**	**0.82**
	Gemma2 9b	0.76	0.51	1.00	0.68	1.00	0.68	1.00	1.28
	Llama3.1 8b	0.82	0.67	0.57	0.62	0.86	0.90	1.00	0.64
O_Pa	Mistral 7b	0.63	0.37	0.84	0.52	0.92	0.56	0.96	0.68
	Qwen2 7b	0.76	0.54	0.33	0.41	0.80	0.90	1.00	0.58
	Average	**0.74**			**0.56**			**0.99**	**0.80**
	Gemma2 9b	0.76	0.53	0.48	0.50	0.83	0.86	1.00	1.88
	Llama3.1 8b	0.81	1.00	0.12	0.21	0.81	1.00	0.95	0.72
N_E_P	Mistral 7b	0.77	0.75	0.14	0.24	0.77	0.98	0.98	0.69
	Qwen2 7b	0.70	0.40	0.38	0.39	0.80	0.81	1.00	0.57
	Average	**0.76**			**0.34**			**0.98**	**0.97**

Table 9.4: Results per label

tion rate and the average generation time but not on the accuracy. The results vary between models but on average the f_1 score is maximized with 2 to 3 examples, the classification rate is maximized with less than 4 examples and the average generation time is maximized with 2 to 3 examples as well. Overall the classifier is expected to yield the best results with 2 to 3 examples given in the prompt when using such models.

9.1.4 Results per Label

The results in this section aims to study the impact of the label to classify on the quality of the classification task. As a reminder the 4 models chosen in Section 9.1.1 (Qwen2 7b, Mistral 7b, Llama3.1 8b and Gemma2 9b) have been evaluated on 3 different prompts. The prompts will be described using the TELeR prompt taxonomy described in Section 8.2. The prompts were all Single-turn, Instruction-style with instructions in French, System-role defined prompts with a Level of Detail 4 with 2-shot prompting. The key difference between the prompts is the label they are trying to classify. The 3 prompts had the following labels to classify:

1. OVERUSE_OF_PERMIT
2. OVERUSE_OF_PASSION
3. NOT_ENOUGH_PERSONALITY

The prompts can be found in Appendix A.1.4.

The results per label and per model and the average values per label can be found in Table 9.4. On average it seems that the labels impacts the f_1 score a lot. On the contrary it does not seem to impact the accuracy that much. On average it seems that the classification rate and generation time stay the same. However there are some differences based on the model that help to put things in perspective.

When looking at the accuracy in depth, it can be seen that on average the classifier has a better accuracy for the label NOT_ENOUGH_PERSONALITY with an accuracy of 0.76 and worse accuracy for the label OVERUSE_OF_PERMIT with a value of 0.69. However it should be noted that it does not correspond with the f_1 scores for the different labels. Especially because the f_1 score for the label NOT_ENOUGH_PERSONALITY is the lowest of the three. This can be explained by the models mostly classifying the the label NOT_ENOUGH_PERSONALITY as the main (negative) class leading to a higher accuracy but not a higher f_1 value. This means that even though the accuracy is higher the models are not better at differentiating between the two classes. Therefore it can be concluded that although the accuracy stays the same, the reality is that in some cases the accuracy is inflated - this might be due to label imbalance. This can be seen as the f_1 scores are really different between the labels. The accuracy alone does not prperly indicate the ability of the model to predict the different labels.

When looking at the f_1 score in depth, it can be seen that on average the classifier has a better score with the label OVERUSE_OF_PASSION with a value of 0.56 and a worse score with the label NOT_ENOUGH_PERSONALITY with a value of 0.34. This trend is followed by Gemma2 9b, Llama3.1 8b and Mistral 7b. Qwen2 7b achieves a slightly better score for the label OVERUSE_OF_PERMIT with a value of 0.44 against a score of 0.41 for the label OVERUSE_OF_PASSION. Therefore it can be concluded that there seem to be general rule that all the models under-perform when classifying the label NOT_ENOUGH_PERSONALITY and that most models perform the best with the label OVERUSE_OF_PASSION.

When looking in depth at the classification rate, it can be seen that on average, almost all responses could be parsed for the label OVERUSE_OF_PERMIT, a few responses could not be parsed for the label OVERUSE_OF_PASSION and a few more for the label NOT_ENOUGH_PERSONALITY. Gemma2 9b and Qwen2 7b both achieve a perfect classification rate. Mistral 7b and Llama3.1 8b both unepectedly formatted some responses. Mistral 7b achieves a classification rate of 0.96 for the label OVERUSE_OF_PASSION and 0.98 for the label NOT_ENOUGH_PERSONALITY. On the other hand Llama3.1 8b achieves a classification rate of 0.95 on the label NOT_ENOUGH_PERSONALITY and 0.99 on the label OVERUSE_OF_PERMIT. Therefore it can be concluded that Mistral7b and Llama3.1 8b in particular are having trouble to answer according to the desired format than the other models. These errors are happening on average more often for the NOT_ENOUGH_PERSONALITY and OVERUSE_OF_PASSION labels.

When looking in depth at the average generation time, it appears that, on average, the models take longer to answer for the NOT_ENOUGH_PERSONALITY label with an average time of 0.97s compared to 0.82s and 0.80s for the labels OVERUSE_OF_PERMIT and OVERUSE_OF_PASSION respectively. This trend is followed by the 4 models but especially visible for Gemma2 9b but is highly correlated to the average number of tokens generated (63.21 for the OVERUSE_OF_PERMIT label, 42.93 for the OVERUSE_OF_PASSION label, and 78.55 for the NOT_ENOUGH_PERSONALITY label). It is also correlated to the corresponding f_1 score. Therefore it can be concluded that the models on average take more time to classify the NOT_ENOUGH_PERSONALITY label and this is due to the fact that they generate more tokens. Some models seem to be express this behaviour more than others such as Gemma2 7b for example.

In conclusion the label seem to have a significant and meaningful impact on the f_1 score and the average generation time. They might also have an impact on the classification rate but

model	bleu	rouge	avg time
Mistral 7b	0.80	0.84	5.94
Codestral 22b	0.41	0.45	11.77
Mixtral 8x7b Q3	0.81	0.87	33.00
Llama2 7b	0.04	0.06	5.12
Llama2 13b	0.51	0.59	110.65
Llama3 8b	0.36	0.44	7.65
Llama3.1 8b	0.68	0.76	6.43
Gemma 2b	0.63	0.78	2.48
Gemma 7b	0.70	0.76	4.41
Gemma2 2b	0.43	0.61	13.72
Gemma2 9b	0.84	0.88	6.61
Gemma2 27b	**0.90**	**0.92**	14.41
Phi3 3.8b	0.02	0.05	7.83
Phi3 14b	0.48	0.56	9.50
Qwen2 7b	0.61	0.68	5.64
Qwen1.5 32b	0.34	0.44	93.78

Table 9.5: Bleu and rouge scores

not on the accuracy. The results vary a little bit between models but on average the f_1 score is maximal for the OVERUSE_OF_PASSION label and is minimal for the NOT_ENOUGH_PERSONALITY label. This means that the classifier performs better on the OVERUSE_OF_PASSION label and worse on the NOT_ENOUGH_PERSONALITY label. The average generation seem to be correlated to the f_1 score since it is minimal for the OVERUSE_OF_PASSION label and maximal for the NOT_ENOUGH_PERSONALITY label. The classification rate might also be correlated to the f_1 score since the lowest classification rate is observed for the NOT_ENOUGH_PERSONALITY label. Overall the classifier yields the best quality and fastest results for the OVERUSE_OF_PASSION label and the worst and slowest results for the NOT_ENOUGH_PERSONALITY label.

9.2 Generation Task

9.2.1 Model Comparison

In this section, the different models are tested on the same specific prompt that aims to correct a unique label. The prompt can be found in Appendix A.2.1. The full results can be found in Table 9.5.

Some model achieved very low BLEU and ROUGE scores such as Llama2 7b and Phi3 3.8b witch scores below 0.1. These models suffer from terrible hallucinations. Then there are some models with low BLEU and ROUGE scores such as Llama3 8b, Phi3 14b, Gemma2 2b, Codestral, Qwen1.5 32b and Llama2 13b with scores ranging from 0.34 to 0.61. These models provide answers containing parts of the desired text but may not contain the full text, they may also include some other examples given in the prompt and answer in English instead of French. Then there are some models with medium BLEU scores and ROUGE

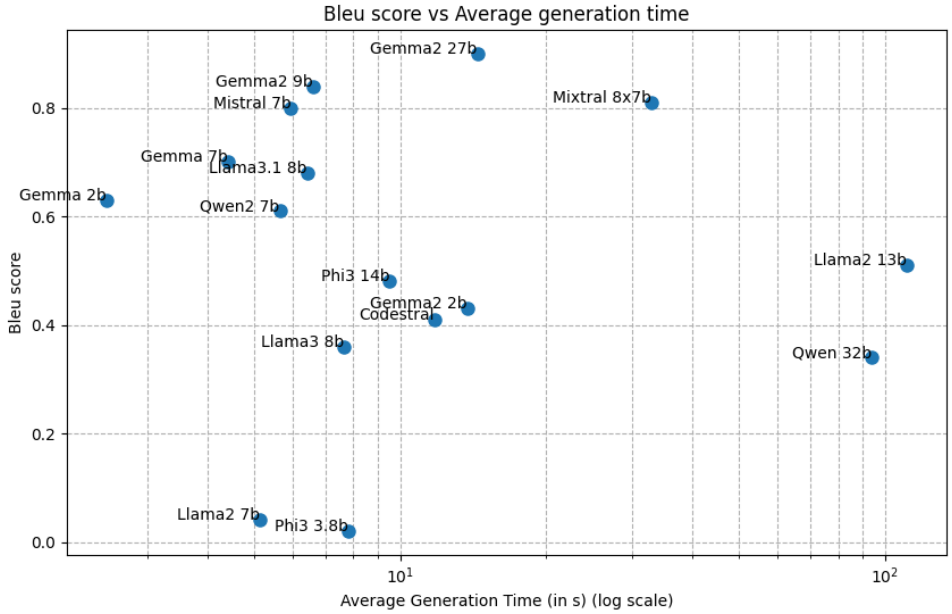

Figure 9.2: bleu score vs average generation time

scores such as Gemma 2b, Gemma 7b, Qwen2 7b, Llama3.1 8b with values from 0.61 to 0.78. Those models are subject to rare hallucinations and mainly provide a variation of the desired text in the correct language. Finally the models with the best BLEU and ROUGE scores are Mistral 7b, Gemma2 9b, Gemma2 27b and Mixtral 8x7b with scores from 0.80 to 0.92. Those models hallucinate even less than the ones from the previous category. They consistently give correctly formatted answers.

Most of the models give their responses under 4 to 15 seconds. One model manages to give its responses under less than 3 seconds: Gemma 2b. Some other models require a lot more time to generate their answers: Mixtral 8x7b requires over 30 seconds, Qwen1.5 32b and Llama2 13b require around 90 and 110 seconds respectively.

Looking at the relations between the BLEU score and the Average generation time in Figure 9.2 and the relation between the ROUGE score and the Average generation time in Figure 9.3, the models with the best Score-over-time ratio were kept for further studies. The following 5 models were kept:

1. Gemma2 9b
2. Gemma2 27b
3. Gemma 2b
4. Gemma 7b
5. Mistral 7b

It can be noted that Gemma models tend to give both good BLEU and ROUGE score along short generation times which make them suitable models for this task.

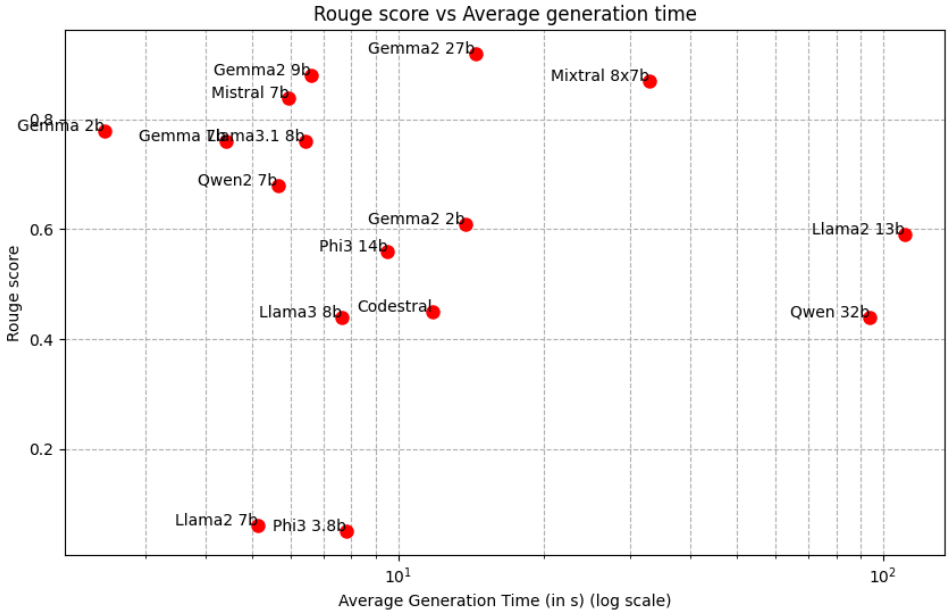

Figure 9.3: rouge score vs average generation time

9.2.2 Results per Label

The results in this section aims to study the impact of the label to classify on the quality of the classification task. As a reminder the 5 models chosen in Section 9.2.1 (Gemma2 9b, Gemma2 27b, Gemma 2b, Gemma 7b, Mistral 7b) have been evaluated on 3 different prompts. The prompts will be described using the TELeR prompt taxonomy described in Section 8.2. The prompts were all Single-turn, Instruction-style with instructions in French, System-role defined prompts with a Level of Detail 4 with 1-shot prompting. The key difference between the prompts is the label they are trying to correct. The 3 prompts had the following labels to correct:

1. OVERUSE_OF_PERMIT
2. OVERUSE_OF_PASSION
3. NOT_ENOUGH_PERSONALITY

The prompts can be found in Appendix A.2.2.

The results per label and per model and the average values per label can be found in Table 9.6. On average it seems that the labels seem to have a significant impact on the BLEU and ROUGE scores. It also seem that it has on average somewhat of an impact on the generation time. However there are some differences based on the model that help to put things in perspective.

When looking at the BLEU score in details it appears that, on average, the models achieve better scores with the OVERUSE_OF_PERMIT and OVERUSE_OF_PASSION labels than with the label NOT_ENOUGH_PERSONALITY with a score of 0.76 against 0.36. The 5 models also follow this trend which indicates that there is something that makes it hard

prompt	model	bleu	rouge	avg time
O_Pa	Gemma2 9b	0.84	0.89	50.50
	Gemma2 27b	0.88	0.90	62.07
	Gemma 2b	0.53	0.69	2.36
	Gemma 7b	0.71	0.78	3.86
	Mistral 7b	0.85	0.88	5.63
	Average	**0.76**	**0.83**	**24.88**
N_E_P	Gemma2 9b	0.26	0.41	40.35
	Gemma2 27b	0.28	0.41	46.72
	Gemma 2b	0.34	0.42	1.63
	Gemma 7b	0.47	0.56	2.82
	Mistral 7b	0.44	0.57	4.18
	Average	**0.36**	**0.47**	**19.14**
O_Pe	Gemma2 9b	0.72	0.85	61.15
	Gemma2 27b	0.84	0.91	61.32
	Gemma 2b	0.68	0.79	2.14
	Gemma 7b	0.83	0.89	4.35
	Mistral 7b	0.73	0.87	5.45
	Average	**0.76**	**0.86**	**26.88**

Table 9.6: Bleu and rouge scores

to get a good BLEU score for the label NOT_ENOUGH_PERSONALITY. Some models achieve a better score with the label OVERUSE_OF_PASSION compared to with the label OVERUSE_OF_PERMIT such as Gemma2 9b (0.84 against 0.72), Gemma2 27b (0.88 vs. 0.84) and Mistral 7b (0.85 against 0.73). The others achieve worse results with the label OVERUSE_OF_PASSION compared to with the label OVERUSE_OF_PERMIT such as Gemma 2b (0.53 vs. 0.68) and Gemma 7b (0.71 against 0.73). Therefore it can be concluded that there its something that makes it harder to achieve a good BLEU score with the label NOT_ENOUGH_PERSONALITY. The two other labels are give an easier time for the models to get a good score and the models manage on average to get the same scores. The scores depends on the model suggesting that some models have a easier time with some labels and less with other labels.

When looking at the ROUGE score in details it appears that, on average, the models achieve the better scores with the label OVERUSE_OF_PERMIT with a score of 0.86, then OVERUSE_OF_PASSION with a score of 0.83 and finally the models achieve the worst scores with the label NOT_ENOUGH_PERSONALITY with a score of 0.47. The 5 models achieve worse scores with the label NOT_ENOUGH_PERSONALITY than with any of the other two labels. It indicates that there is something that makes it hard to get a good ROUGE score for the label NOT_ENOUGH_PERSONALITY. Some models achieve a better score with the label OVERUSE_OF_PASSION compared to with the label OVERUSE_OF_PERMIT such as Gemma2 9b (0.89 against 0.85) and Mistral 7b (0.88 against 0.87). The others achieve worse results with the label OVERUSE_OF_PASSION compared to with the label OVERUSE_OF_PERMIT such as Gemma2 27b (0.90 vs. 0.91), Gemma 2b (0.69 vs. 0.79) and Gemma 7b (0.78 against 0.89). Therefore it can be concluded that there its something that makes it harder to achieve a good ROUGE score with the label NOT_ENOUGH_PERSONALITY. The two other labels

are give an easier time for the models to get a good score and the models manage on average to get comparable scores. The scores depends on the model suggesting that some models have a easier time with some labels and less with other labels.

When looking at the average generation time it can be seen that on average the time to generate the answers is lower with the label NOT_ENOUGH_PERSONALITY (19.14 seconds) than with the labels OVERUSE_OF_PASSION (24.88 seconds) and OVERUSE_OF_PERMIT (26.88 seconds). This trend is followed by all the models. This trend is due to less tokens being generated with the label NOT_ENOUGH_PERSONALITY (252 tokens) than with the labels OVERUSE_OF_PASSION (327 tokens) and OVERUSE_OF_PERMIT (326 tokens).

In conclusion the label seem to have a significant and meaningful impact on the BLEU and ROUGE score and the average generation time. There is a clear gap between scores with the labels OVERUSE_OF_PASSION and OVERUSE_OF_PERMIT and scores with the label NOT_ENOUGH_PERSONALITY, the first ones being a lot higher. Therefore there is something with the label NOT_ENOUGH_PERSONALITY that makes it harder to get good scores. There is also a gap in generation time. The models generate less tokens when working with the NOT_ENOUGH_PERSONALITY label than with the two other labels. Overall the generation yields the best BLEU and ROUGE scores and slowest results for the OVERUSE_OF_PASSION and OVERUSE_OF_PERMIT labels and the worst but fastest results for the NOT_ENOUGH_PERSONALITY label.

9.2.3 Human Scoring

In order to verify that the BLEU and ROUGE scores actually mean that the text has been enhanced, a comparison is made in this sub section between the BLEU and ROUGE scores and a human scoring.

15 generated texts were picked and put side to side with their augmented version. These 15 texts correspond to one text per model and per label to correct. An expert was then asked to give its opinion on the 2 presented texts without knowing which text was the generated one and which text was the augmented one.

The expert was asked the following question.

> Lequel des deux textes ci-dessus est le meilleur ?
>
> 'Which of the two texts above is better?'

The expert could then answer with a number from 1 to 7:

1. Le texte de gauche est largement meilleur que celui de droite
   'The text on the left is much better than the text on the right'
2. Le texte de gauche est meilleur que celui de droite
   'The text on the left is better than the text on the right'
3. Le texte de gauche est un peu meilleur que celui de droite
   'The text on the left is a little better than the text on the right'
4. Les deux textes ont une qualité équivalente
   'The two texts are of equivalent quality'

Correlation	Score
Human scoring / BLEU score	0.50
Human scoring / ROUGE score	0.41

Table 9.7: Correlation between Human Scoring and Evaluation Metrics

5. 'Le texte de droite est un peu meilleur que celui de gauche'
   The text on the right is a little better than the text on the left
6. 'Le texte de droite est meilleur que celui de gauche'
   The text on the right is better than text on the left
7. 'Le texte de droite est largement meilleur que celui de gauche'
   The text on the right is much better than the text on the left

Based on which one of the text was the generated one (right or left) and which one was the augmented one these numbers were then converted into a new scale from 1 to 7:

1. The generated text is **much worse** than the augmented one
2. The generated text is **worse** than the augmented one
3. The generated text is **a little worse** than the augmented one
4. The generated text is **comparable to** the augmented one
5. The generated text is **a little better** than the augmented one
6. The generated text is **better** than the augmented one
7. The generated text is **much better** than the augmented one

Figure 9.4 features the BLEU and ROUGE scores against the previous scale. It can be seen that generated text that were considered a little better, better or much better correspond on average to better BLEU and ROUGE scores than the ones considered a little worse, worse or much worse. However it should be noted that some texts achieved high BLEU and ROUGE scores but were considered a little worse, worse or much worse. Therefore the the score alone cannot guarantee the quality of the generated text but is still correlated to it. The correlation between the BLEU score and human scoring as well as the correlation between the ROUGE score and the human scoring are featured in Table 9.7. It can be seen that the BLEU score is a better indicator than the ROUGE score of the quality of the generated text as it correlates more with the human scoring.

9.3 Further Work

Here is a non-exhaustive list of potential improvements and future work that could be explored:

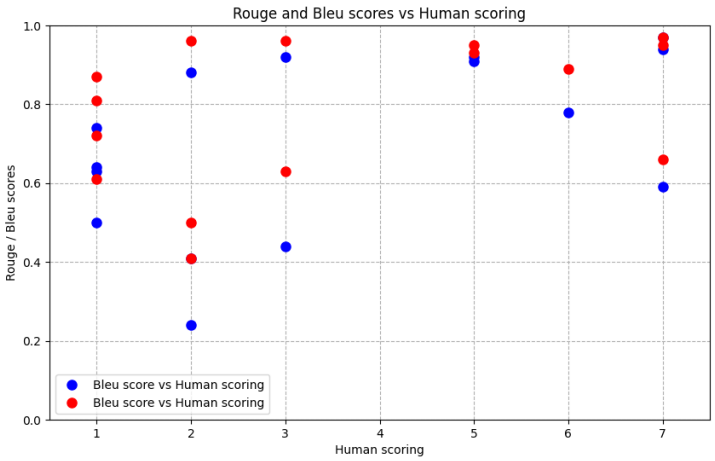

Figure 9.4: bleu rouge scores vs human scoring

Regarding the Test Set:

- Enhance the method for creating a test dataset from unlabeled data by degrading high-quality data. This could involve having an expert label the best texts and using the back-translation technique to generate a larger quantity of higher-quality data.

Regarding the Classification and Improvement Method:

- Implement dynamic insertion of examples into prompts to include text similar to the text to be approved, in order to assess whether this improves performance.
- Experiment with a multi-prompt approach to improve results for more complex guidelines. This could involve initially detecting areas or words of interest, followed by their classification or modification. This method may yield better results and enhanced explainability.
- Investigate whether automatic translation of texts into English prior to classification improves performance.
- Conduct more comprehensive research to identify suitable measures for text improvement.

Regarding the Overall Approach:

- Apply this methodology to other labeled datasets or different domains to compare results.

Chapter 10
Conclusion

This thesis explores the feasibility of using prompt engineering with LLMs for text classification and enhancement tasks, specifically within the context of ensuring adherence to guidelines in the creation of *Records of Expertise*. This approach was driven by the unique nature of the document, limited data availability, and constrained time resources which made traditional machine learning techniques less feasible.

Several significant challenges emerged during this research, including the limited volume of data, the use of an unlabeled dataset, concerns about data sovereignty, constraints on computing power, and the need to work with French language data. To address these issues, the research proposes a method for generating a labeled test set from unlabeled data through data degradation on high-quality sources. Additionally, it focuses on small language models with typically a few billion parameters.

The study evaluated the performance of various models on both classification and enhancement tasks, incorporating human judgment to assess the results. It examined the effects of instruction language, prompt type, and guidelines on classification performance, as well as the impact of guidelines on enhancement tasks.

Key findings from the study include that the Qwen2 7b model was the top performer in text classification, achieving the highest F1 score and accuracy among the models evaluated. The instruction language had a notable effect, with instructions in French yielding better outcomes. The prompt type also influenced performance, with 2 to 3 examples proving most effective, and style-oriented guidelines had a positive impact on the results. In text generation, the Gemma model family outperformed others, with Gemma2 27b achieving the highest BLEU and ROUGE scores. The style-oriented guidelines again played a significant role, and the test set generation method may have influenced these results. Notably, the BLEU score aligned most closely with human judgment.

Based on these findings, several improvements and future research directions are proposed. Enhancements to the test dataset creation method, such as incorporating back-translation and selective expert labeling, could improve the quality and significance of the results. Further exploration of dynamic example insertion, multi-prompt approaches, and automatic

translation may enhance classification and generation performance. Applying this methodology to other labeled datasets and domains could provide additional insights and validate the approach.

Overall, this research demonstrates that prompt engineering with LLMs, even with smaller models, can be effective for both classification and generation tasks in French. While challenges persist, particularly regarding test set labeling and performance with complex guidelines, the results are promising and offer a foundation for future exploration and application of this methodology in various contexts.

References

Marah Abdin et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Mistral AI. Mistral AI | Frontier AI in your hands — mistral.ai. `https://mistral.ai/fr/`, 2024. [Accessed 19-08-2024].

Jinze Bai et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Yoshua Bengio et al. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

Lucas Beyer et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Aleksandar Botev et al. Recurrentgemma: Moving past transformers for efficient open language models. *arXiv preprint arXiv:2404.07839*, 2024.

MIT Critical Data et al. Exploratory data analysis. *Secondary analysis of electronic health records*, pages 185–203, 2016.

David Drewery et al. Artificial intelligence and résumé critique experiences. *Canadian Journal of Career Development*, 21(2):28–39, 2022.

Abhimanyu Dubey et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Andrea Ferrario and Mara Nägelin. The art of natural language processing: classical, modern and contemporary approaches to text document classification. *SSRN's Research Paper Series*, 2020.

Suriya Gunasekar et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

Alyssa Hughes. Phi-2: The surprising power of small language models — microsoft.com. `https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/`, 2024. [Accessed 21-08-2024].

Albert Q Jiang et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Albert Q Jiang et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Dhawal Khem et al. An overview of context capturing techniques in nlp. *International Journal on Recent and Innovation Trends in Computing and Communication*, 11:193–198, 2023.

Stefan Kombrink et al. Recurrent neural network based language modeling in meeting recognition. In *Interspeech*, volume 11, pages 2877–2880, 2011.

Yuanzhi Li et al. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*, 2023.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL `https://aclanthology.org/W04-1013`.

Md Saef Ullah Miah et al. A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and llm. *Scientific Reports*, 14(1):9603, 2024.

Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Kishore Papineni et al. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002. URL `https://aclanthology.org/P02-1040.pdf`.

Shubhra Kanti Karmaker Santu and Dongji Feng. Teler: A general taxonomy of llm prompts for benchmarking complex tasks. *arXiv preprint arXiv:2305.11430*, 2023.

Rico Sennrich et al. Improving neural machine translation models with monolingual data. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1009. URL `https://aclanthology.org/P16-1009`.

Maite Taboada et al. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307, 2011.

Gemma Team et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024a.

Gemma Team et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024b.

Qwen Team. Introducing Qwen — qwenlm.github.io. `https://qwenlm.github.io/blog/qwen/`, 2024a. [Accessed 21-08-2024].

Qwen Team. Introducing Qwen1.5 — qwenlm.github.io. `https://qwenlm.github.io/blog/qwen1.5/`, 2024b. [Accessed 21-08-2024].

Hugo Touvron et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Grigorios Tsoumakas and Ioannis Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *European conference on machine learning*, pages 406–417. Springer, 2007.

Ashish Vaswani et al. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wikipedia. Gemini (language model) - Wikipedia — en.wikipedia.org. https://en.wikipedia.org/wiki/Gemini_(language_model), 2024a. [Accessed 20-08-2024].

Wikipedia. LLaMA — Wikipédia — fr.wikipedia.org. https://fr.wikipedia.org/wiki/LLaMA, 2024b. [Accessed 20-08-2024].

Wikipedia. Mistral AI - Wikipedia — en.wikipedia.org. https://en.wikipedia.org/wiki/Mistral_AI, 2024c. [Accessed 19-08-2024].

An Yang et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Min-Ling Zhang et al. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12:191–202, 2018.

Wayne Xin Zhao et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Appendices

Appendix A
Prompts

A.1 Classification

A.1.1 Model Comparison

```
## INSTRUCTION

Vous êtes un agent d'assurance qualité expérimenté qui
vérifie si un texte répond aux attentes exprimées. Les
attentes sont formulées sous forme de lignes directrices.
Chacune des lignes directrices est associée à un label.
Vous associez soigneusement et précisément chacun des labels
à une valeur booléenne pour indiquer si la consigne est
respectée ou non.

Classez le texte avec les labels données.

Répondez avec un format JSON :

{

    <label> : <booléen>

}

## LABELS

OVERUSE_OF_PASSION : le texte fourni utilise excessivement
```

```
le champ lexical de la passion.

## TEXTE no.1

[Example respecting the guildeline]

## RÉPONSE no.1
{
    "OVERUSE_OF_PASSION": 0
}

## TEXTE no.2

[Example disrespecting the guildeline]

## RÉPONSE no.2

{
    "OVERUSE_OF_PASSION": 1
}

## TEXTE no.3

[Text to classify]

## RÉPONSE no.3
```

A.1.2 Instruction Language Comparison

```
## INSTRUCTION

You are an experienced quality assurance agent who checks
whether a text meets the stated expectations. Expectations
are formulated in the form of guidelines. Each of the
guidelines is associated with a label. You carefully and
precisely associate each of the labels with a Boolean value
to indicate whether or not the guideline is respected.

Classify the following texts with the given labels.
Answer with a JSON format:
{
    <label>: <boolean>
}

## LABELS
```

```
OVERUSE_OF_PASSION: the text provided does not excessively
use the lexical field of passion.

## TEXT no.1

[Example respecting the guildeline]

## RESPONSE no.1

{
    "OVERUSE_OF_PASSION": 0
}

## TEXT no.2

[Example disrespecting the guildeline]

## RESPONSE no.2

{
    "OVERUSE_OF_PASSION": 1
}

## TEXT no.3

[Text to classify]

## RESPONSE no.3
```

A.1.3 Prompt Type Comparison

```
## INSTRUCTION

Vous êtes un agent d'assurance qualité expérimenté qui vérifie
si un texte répond aux attentes exprimées. Les attentes sont
formulées sous forme de lignes directrices. Chacune des lignes
directrices est associée à un label. Vous associez soigneusement
et précisément chacun des labels à une valeur booléenne pour
indiquer si la consigne est respectée ou non.

Classez le texte avec les labels données.
Répondez avec un format JSON :
{
  <label> : <booléen>
}
```

```
## LABELS

OVERUSE_OF_PASSION: le texte fourni utilise excessivement le champ
lexical de la passion.

## TEXTE no.1

[Example respecting the guildeline]

## RÉPONSE no.1

{
    ""OVERUSE_OF_PASSION"": 0
}

## TEXTE no.2

[Example disrespecting the guildeline]

## RÉPONSE no.2

{
    ""OVERUSE_OF_PASSION"": 1
}

## TEXTE no.3

[Example respecting the guildeline]

## RÉPONSE no.3

{
    ""OVERUSE_OF_PASSION"": 0
}

## TEXTE no.4

[Example disrespecting the guildeline]

## RÉPONSE no.4

{
    ""OVERUSE_OF_PASSION"": 1
}
```

```
## TEXTE no.5

[Example respecting the guildeline]

## RÉPONSE no.5

{
    ""OVERUSE_OF_PASSION"": 0
}

## TEXTE no.6

[Example disrespecting the guildeline]

## RÉPONSE no.6

{
    ""OVERUSE_OF_PASSION"": 1
}

## TEXTE no.7

[Text to classify]

## RÉPONSE no.7
```

A.1.4 Label Comparison

Label OVERUSE_OF_PERMIT

```
## INSTRUCTION

Vous êtes un agent d'assurance qualité expérimenté qui vérifie
si un texte répond aux attentes exprimées. Les attentes sont
formulées sous forme de lignes directrices. Chacune des lignes
directrices est associée à un label. Vous associez soigneusement
et précisément chacun des labels à une valeur booléenne pour
indiquer si la consigne est respectée ou non.

Classez le texte avec les labels données.
Répondez avec un format JSON :
{
  <label> : <booléen>
}
```

```
## LABELS

OVERUSE_OF_PERMIT: le texte fourni utilise de façon excessive le
verbe ""permettre" et ses dérivés.

## TEXTE no.1

[Example respecting the guildeline]

## RÉPONSE no.1

{
    ""OVERUSE_OF_PERMIT"": 0
}

## TEXTE no.2

[Example disrespecting the guildeline]

## RÉPONSE no.2

{
    ""OVERUSE_OF_PERMIT"": 1
}

## TEXTE no.3

[Text to classify]

## RÉPONSE no.3
```

Label OVERUSE_OF_PASSION

```
## INSTRUCTION

Vous êtes un agent d'assurance qualité expérimenté qui vérifie
si un texte répond aux attentes exprimées. Les attentes sont
formulées sous forme de lignes directrices. Chacune des lignes
directrices est associée à un label. Vous associez soigneusement
et précisément chacun des labels à une valeur booléenne pour
indiquer si la consigne est respectée ou non.

Classez le texte avec les labels données.
Répondez avec un format JSON :
{
```

```
  <label> : <booléen>
}

## LABELS

OVERUSE_OF_PASSION: le texte fourni utilise de façon excessive
le champ lexical de la passion.

## TEXTE no.1

[Example respecting the guildeline]

## RÉPONSE no.1

{
    ""OVERUSE_OF_PASSION"": 0
}

## TEXTE no.2

[Example disrespecting the guildeline]

## RÉPONSE no.2

{
    ""OVERUSE_OF_PASSION"": 1
}

## TEXTE no.3

[Text to classify]

## RÉPONSE no.3
```

Label NOT_ENOUGH_PERSONALITY

```
## INSTRUCTION

Vous êtes un agent d'assurance qualité expérimenté qui vérifie
si un texte répond aux attentes exprimées. Les attentes sont
formulées sous forme de lignes directrices. Chacune des lignes
directrices est associée à un label. Vous associez soigneusement
et précisément chacun des labels à une valeur booléenne pour
indiquer si la consigne est respectée ou non.
```

```
Classez le texte avec les labels données.
Répondez avec un format JSON :
{
  <label> : <booléen>
}

## LABELS

NOT_ENOUGH_PERSONALITY: le texte ne fait pas assez ressortir la
personnalité de l'auteur.

## TEXTE no.1

[Example respecting the guildeline]

## RÉPONSE no.1

{
    ""NOT_ENOUGH_PERSONALITY"": 0
}

## TEXTE no.2

[Example disrespecting the guildeline]

## RÉPONSE no.2

{
    ""NOT_ENOUGH_PERSONALITY"": 1
}

## TEXTE no.3

[Text to classify]

## RÉPONSE no.3
```

A.2 Generation

A.2.1 Model Comparison

```
## INSTRUCTION

Tu es un agent capable de générer des variations subtiles d'un texte
en fonction des instructions qui te sont données sans changer le sens
```

```
du texte.

Ré-écris les textes suivants en utilisant moins souvent le champ
lexical de la passion. Par exemple en utilisant moins souvent des
tournures comme ""ma passion", ""je suis passionné", ""un projet
passionnant", ""je me passionne pour" etc.

## TEXTE no. 1

[Example to be enhanced]

## REPONSE no. 1

[Enhanced version of the previous example]

## TEXTE no.2

[Text to enhance]

## REPONSE no. 2
```

A.2.2 Label Comparison

Label OVERUSE_OF_PASSION

```
## INSTRUCTION

Tu es un agent capable de générer des variations subtiles d'un texte
en fonction des instructions qui te sont données sans changer le
sens du texte.

Ré-écris les textes suivants en utilisant moins souvent le champ lexical
de la passion. Par exemple en utilisant moins souvent des tournures
comme ""ma passion", ""je suis passionné", ""un projet passionnant",
""je me passionne pour" etc.

## TEXTE no. 1

[Example to be enhanced]

## REPONSE no. 1

[Enhanced version of the previous example]

## TEXTE no.2
```

```
[Text to enhance]

## REPONSE no. 2
```

Label NOT_ENOUGH_PERSONALITY

```
## INSTRUCTION

Tu es un agent capable de générer des variations subtiles d'un texte
en fonction des instructions qui te sont données sans changer le
sens du texte.

Ré-écris les textes suivants en insistant sur les parties qui font
ressortir la personnalité de l'auteur.

## TEXTE no. 1

[Example to be enhanced]

## REPONSE no. 1

[Enhanced version of the previous example]

## TEXTE no.2

[Text to enhance]

## REPONSE no. 2
```

Label OVERUSE_OF_PERMIT

```
## INSTRUCTION

Tu es un agent capable de générer des variations subtiles d'un texte
en fonction des instructions qui te sont données sans changer le
sens du texte.

Ré-écris les textes suivants en utilisant plus moins le verbe
""permettre" et ses dérivés. Par exemple en utilisant moins souvent
des tournures comme ""me permettant", ""m'a permis", ""me permettent",
""permet" etc.

## TEXTE no. 1

[Example to be enhanced]
```

```
## REPONSE no. 1

[Enhanced version of the previous example]

## TEXTE no.2

[Text to enhance]

## REPONSE no. 2
```

INSTITUTIONEN FÖR DATAVETENSKAP | LUNDS TEKNISKA HÖGSKOLA | PRESENTERAD 2024-08-29

EXAMENSARBETE Using Small LLMs to Assess and Enhance Skill Management Documents
STUDENT Maxime Pakula
HANDLEDARE Pierre Nugues (LTH)
EXAMINATOR Jacek Malec (LTH)

Mastering Tool Creation: The Power of Formulating Instructions with LLMs

POPULÄRVETENSKAPLIG SAMMANFATTNING **Maxime Pakula**

Med framväxten av allt mer avancerade och tillgängliga språkmodeller verkar skapandet av verktyg för språkbehandling baserade på dessa teknologier nu vara inom räckhåll för alla. Denna studie bedömer genomförbarheten av att designa ett effektivt verktyg genom att enbart fokusera på utformningen av instruktioner.

Natural Language Processing expanderar snabbt och möjliggör för företag att automatisera textrelaterade uppgifter. Traditionellt krävde detta stora datamängder, vilket kan vara utmanande för små eller nischade företag. Nyligen har kraftfulla, öppen källkod stora språkmodeller introducerats som kan bearbeta text effektivt, men som kräver betydande datorkraft och ofta är värdextern, vilket väcker oro för datastyrning. Små språkmodeller erbjuder ett lovande alternativ. De kräver färre resurser och kan hostas lokalt, vilket ger bättre kontroll över data för företag.

Detta examensarbete, som genomfördes på ett franskt konsultföretag, syftade till att skapa ett verktyg för konsulter att skriva dokument som visar deras färdigheter och erfarenheter. Företaget står inför utmaningar, inklusive den extra svårigheten med att bearbeta data på franska, eftersom de flesta modeller är tränade på engelska. Verktyget som är under utveckling måste kontrollera att texterna överensstämmer med företagets riktlinjer och kunna förbättra dem vid behov. Under min forskning skapade jag först ett testdataset och valde jämförelsemått. Därefter testade jag olika små språkmodeller för att bedöma deras effektivitet baserat på språket som instruktionerna är skrivna på och mängden exempel som ges. Slutligen jämförde jag de erhållna resultaten med en mänsklig bedömning.

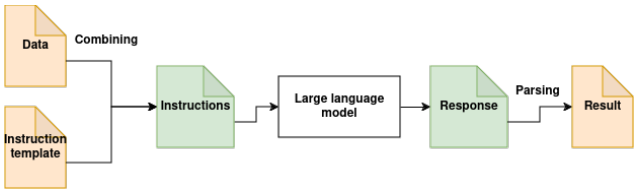

Figure 1: Schematisk vy av verktyget

Även om metoden är ganska enkel, är resultaten mycket uppmuntrande. De visar att vissa små språkmodeller redan är tillräckligt avancerade för att automatisera denna typ av uppgift. Jag fann särskilt att när texterna är på franska, är det bättre att ge instruktionerna på franska, även om modellerna främst har tränats på engelska. Jag observerade också att det finns ett optimalt antal exempel att ge för att uppnå de bästa resultaten, och för texter av några stycken som de i denna studie, är det optimala antalet mellan 2 och 3 exempel. Dessa resultat antyder att vi bör fortsätta utforska mer sofistikerade instruktionstekniker med dessa små modeller för att ytterligare förbättra prestandan hos de utvecklade verktygen.