Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

HEIA-FR

ISC - IL

2023 - 2024

TB - LLMs for Hotel Booking Bots

Report

Author

Florian Chassot

Professor

Sandy Ingram

Mandator

tripla Co., Ltd.

August 2, 2024

Version 1.0

1. Executive summary

This project involves the integration of advanced sentiment analysis into TriplaBot, a multilingual chatbot software used in the hospitality sector to enhance hotel's client customer interactions. The goal is to get a better understanding of the customer by analyzing their sentiment which will be used in subsequent projects to improve the chatbot behaviour and uses. The project focuses on the elaboration and documentation of the process necessary to perform the sentiment analysis rather than pushing the accuracy of the models.

The project is composed of two main approaches: Message-by-message polarity sentiment analysis and whole conversation sentiment classification analysis. The former tags each message with one of three possibles labels: positive, negative and neutral, while the latter evaluates the general sentiment of entire interaction with the most fitting adjective category describing the sentiment.

In the message-by-message sentiment analysis, we explored various sentiment analysis models, including fine-tuned BERT models and corporate LLM APIs like ChatGPT. We addressed the challenge of multilingual support as well as the lack of example data for some labels. This model allows to isolate messages with positive or negative sentiment and doesn't require entire conversation.

For whole conversation sentiment analysis, we generated sentiment keywords freely using an LLM and categorized them through clustering in order to create possible labels for the prediction. We then used those categories to develop models predicting the most fitting for each conversation. This used the power of embedding LLMs to simplify the input of the classification. This approach aims to use the entire sequence with the pattern of messages to get a more complete understanding of the user sentiment throughout their messages to the chatbot.

In conclusion, the project successfully demonstrated the feasibility and effectiveness of sentiment analysis in a multilingual chatbot context. We explored many techniques and possibility of LLMs, such as ICL, finetuning, embedding generation and usage. The results indicate significant potential of using such analysis to improve Tripla's chatbot services. Future work includes refining the models, integrating the use cases of the model within Tripla's software and exploring more cost-effective alternatives.

Contents

2. Context

Tripla is an international IT company based in Japan that provides SaaS (Software as a Service) solutions for the travel industry. Their solutions help businesses in the hospitality sector reduce staff workload, streamline operations, and improve guest satisfaction. They simplify the process of booking a hotel, making payments, and facilitating communication between hotels and customers.

One of Tripla's services is called TriplaBot. It allows customers to ask queries about a hotel through a chatbot in one of Tripla's four supported languages: Japanese, English, Korean, and Chinese. This service enables customers to receive faster responses and reduces the customer support workload for hotel managers.

A crucial aspect of the chatbot is understanding the user. This includes not only identifying the intent of the current message but also recognizing the user's emotions. Are they irritated or frustrated? Is there an urgent issue? Are they satisfied with their stay? By addressing these questions, the system can better adapt the conversation flow to prevent any negative feelings that could lead to bad reviews or lost sales.

Deep learning technologies have been advancing and are gaining strong interest among many tech companies worldwide. These technologies allow for the gathering and processing of large amounts of data. One application of such technology is sentiment analysis, which extracts the sentiment expressed by the writer from a small portion of text.

The aim of this project is to use deep learning technologies such as LLMs (Large Language Model) and machine learning to process chatbot interactions, starting with sentiment analysis to add sentiment tags to chat traces. Following this, the project will attempt a more comprehensive analysis of entire conversations between users and the chatbot. This analysis will enable Tripla to modify the conversation flow when it detects that a user is dissatisfied with either the hotel services or the chatbot itself. Actions could include connecting the user to a human operator or proposing alternatives to address their issues.

Figure 1 – Tripla logo

3. Objectives

3.1. Main Objectives

The following objectives are the primary focus of the project:

3.1.1. State-of-the-art sentiment analysis on chatbot traces

This sentiment analysis will add an extra attribute to each message in a chat trace, describing the current feeling of the user. This analysis must be compatible with all four languages in which Tripla offers services.

3.1.2. Prototype for analyzing the entire sequence of exchanges in the chatbot

After adding these extra tags to each message, we aim to gain a broader understanding of user interactions. This involves assessing whether the customer is satisfied, whether they might need help, or if they are likely to book the hotel. This analysis will use the sentiment tags from individual chat messages as well as patterns in the sequence of messages (e.g., repetition, similar queries, message length).

3.2. Secondary Objectives

The following objectives will be pursued depending on time constraints and the accuracy of the implementation of the main objectives:

3.2.1. Real-time actions based on chat analysis

We aim to use the results of the second main objective to take actions during chatbot interactions to improve the customer experience if they are not satisfied. We aim to first have a design and possibly a implementation of such actions.

3.2.2. Integration within Tripla's pipeline

If the analysis of the sequence provides accurate results, we will proceed with integrating it into Tripla's pipeline.

4. Methodology

The chosen methodology for this project is agile. The project is split with three main iterations, and with each iteration, the current system is improved. This choice is based on two main reasons:

- **Risk Management and Uncertainty**: The latter part of the project is more experimental, and it is uncertain whether the results of each step will be accurate enough to be used in the subsequent steps. Using an agile methodology allows us to quickly develop a first prototype and detect possible issues, providing opportunities to find solutions early on.
- **Flexibility**: Agile methodology offers a more open and flexible workflow, especially in the later stages, where we might need to shift our focus to either improving the analysis or designing actions based on this analysis.

The three iterations have the following objectives:

1. By the end of the first iteration, we aimed to achieve the first primary objective, which involves developing a sentiment analysis capable of tagging every message in a chat trace.
2. By the end of the second iteration, we planned to have a prototype for the entire chat sequence analysis.
3. The final iteration initial goal was to either improve the analysis or on designing and possibly implementing actions based on the chat sequence analysis. The content of this iteration was dependant on the results of the previous iterations. During the project, we focused on thinking about the use cases for the model, creating good documentation, presentation and UI for a demo as well as continuing to do some experiments with the models.

5. Global Process Overview

The final goal of sentiment analysis is to gain insights from conversations by identifying keywords related to the user's sentiment. We will use two different approaches to achieve this. The first approach involves performing sentiment analysis on a message-by-message basis, detecting negative, neutral, and positive sentiments. The second approach is a deeper analysis of the entire flow of conversation with sentiment keywords.
This chapter provides an overview of the steps undertaken to conduct each of these two analyses.

5.1. Message-by-Message Polarity Analysis

The steps used to create the Message-by-message polarity are illustrated in Figure 2. The initial step involved reviewing the state-of-the-art sentiment analysis models. From this, we explored two different possibilities: a fine-tuned BERT model and the use of a powerful corporate LLM API, which we then compared. Both approaches require labeling the existing dataset. The fine-tuned BERT model has challenges with multiple languages and requires data augmentation and balancing to achieve good results. The corporate LLM API solution needs iterative improvement of prompts and comparison with different models.
For more details see chapter 6.

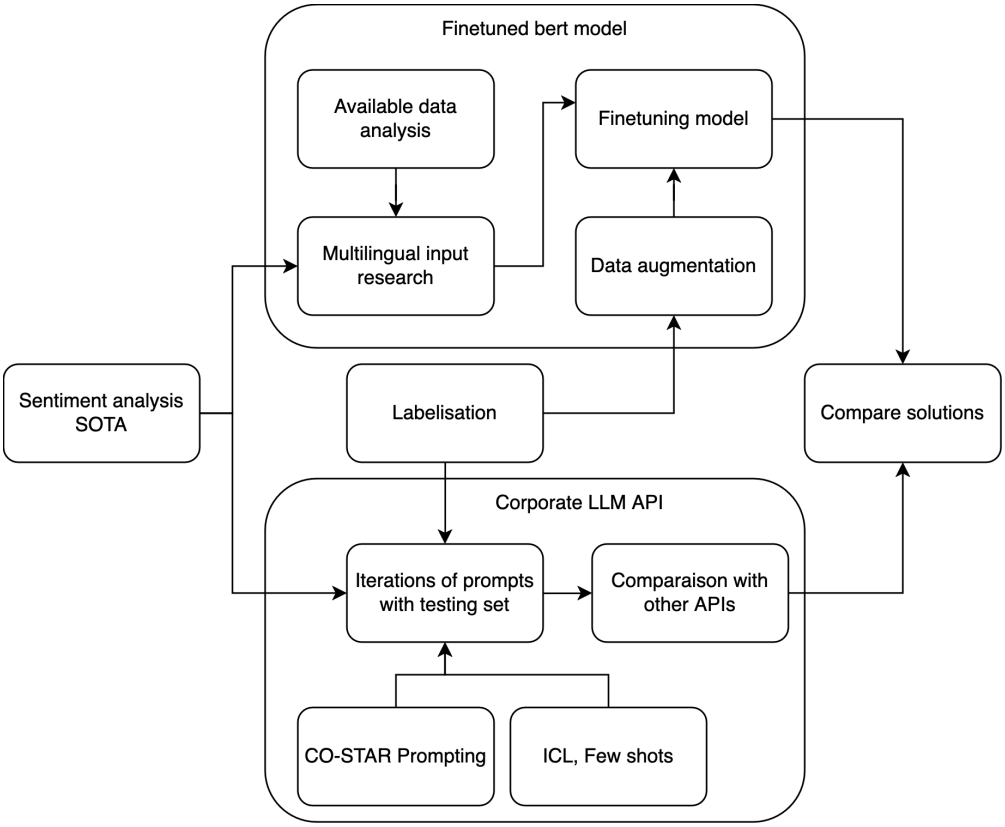

Figure 2 – Steps undertaken for the message-by-message polarity analysis

5.2. Conversation Sentiment Keyword Analysis

This second approach focuses on determining the sentiment of an entire conversation. Its ellaboration is separated into 3 main parts illustrated in Figure 3. The first one involves generating keywords related to sentiment for each conversation without restriction. To do this, we need to create an effective prompt that will generate the keywords and evaluate it on a sample of conversations. After that, we use this prompt on every conversation to generate a list of possible adjectives.

The second part involves creating categories for the keywords. The keywords from the previous step are embedded, and their vectors are dimensionally reduced to improve the results of the next step: clustering. The clustering process will provide us with categories, which we will then validate using Cohen's weighted kappa to determine if the sentiment can be reliably extracted.

The final part is the actual model creation. We use the validated categories from the previous step to label the training and validation datasets. Then, we vectorize each conversation as input for a model that will be trained to predict the labels. Once the model is trained, we evaluate it using the validation set.

For more details see chapter 7.

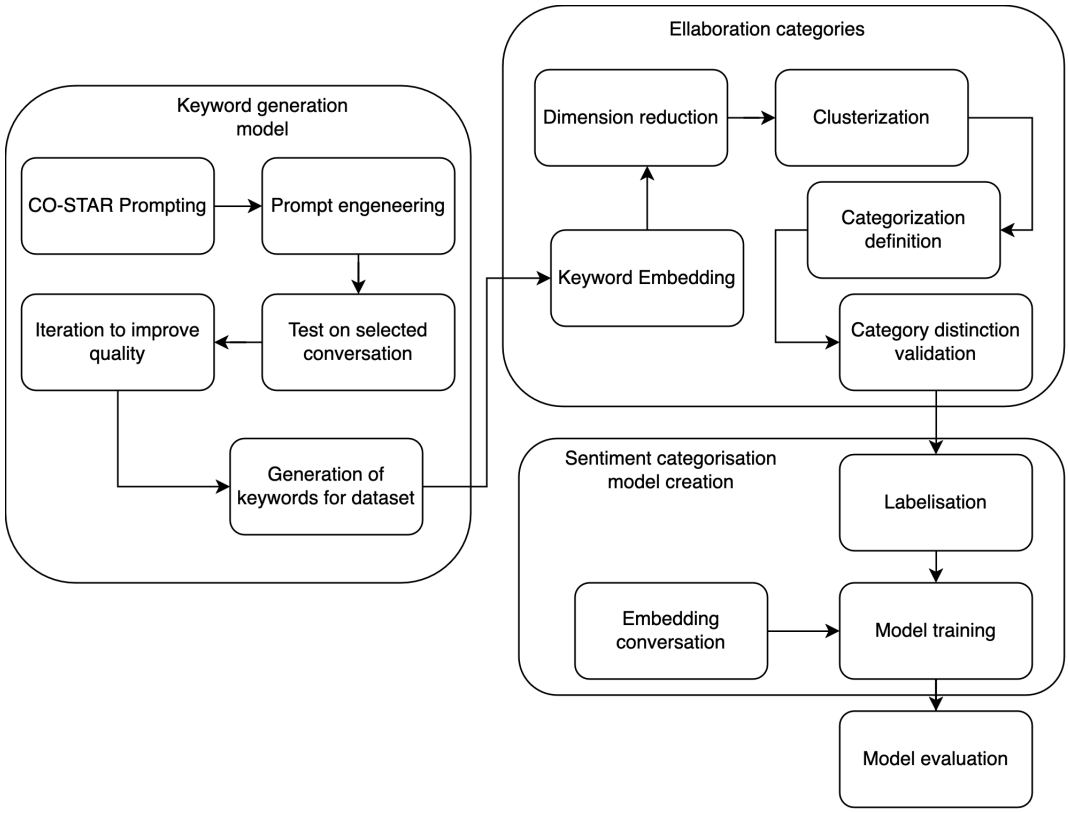

Figure 3 – Steps undertaken for the conversation sentiment keyword analysis

6. Message-by-Message Polarity Analysis

This first approach creates a model that predicts the polarity (positive, negative, neutral) of messages sent by the user to the chatbot. This chapter contains all the steps presented in section 5.1.

6.1. Available Data Analysis

During a conversation between a user and the chatbot, the conversation traces are stored in a JSON file with the following format:

```
{
    "user": {
        "0": "User message #1",
        "1": "User message #2"
    },
    "bot": {
        "0": "Bot reply #1",
        "1": "Bot reply #2"
    }
}
```

The bot replies are chosen automatically based on the user input, with intents such as queries about hotel facilities opening hours or different room offers. The bot replies are unique to each hotel. The bot often repeats itself, caused by the user asking the same or similar questions. On average, 25% of the messages sent by the chatbot are repetitions (this percentage excludes the very first message of each conversation, which is always new).

It is important to note that as Tripla provides services across multiple countries and languages, the data from these chats are in four possible languages: Japanese, English, Korean, and Chinese (Traditional or Simplified). The different proportions of languages used are shown in Figure 4. The large majority of the messages come from Japanese-speaking customers, so it is important to prioritize accuracy in Japanese while maintaining good results in the other available languages.

The amount of data actually used for training is a sample of 300 conversations, with 4'951 user messages in total, provided by Tripla at the start of the project. The conversations are not labeled and will need to be manually labeled to be usable.

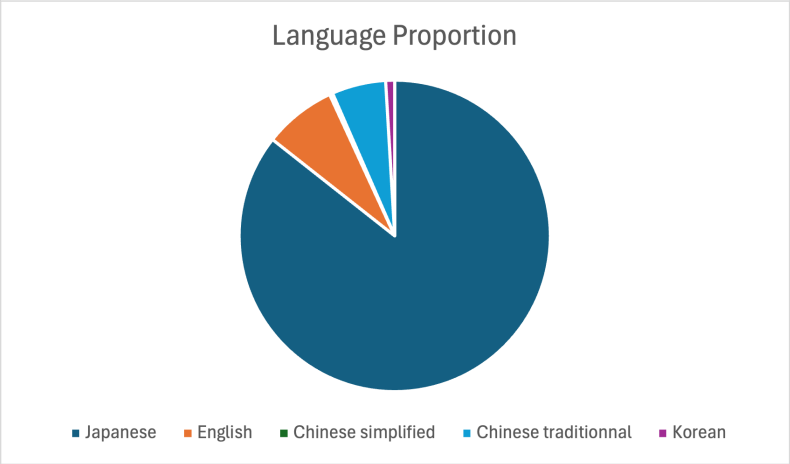

Figure 4 – Proportion of languages in Tripla chatbot messages

The size of each message varies but is mostly short, with an average of 9.68 and a median of 7 characters. While this is low, we need to keep in mind that most messages use Japanese, which uses fewer characters than English because each character conveys more information. The low median indicates many very short messages, such as: ランチ (Lunch), チェックイン (Check-in), 朝食 (Breakfast).

However, there are also longer messages containing entire sentences, mostly questions such as: プレミアムファミリールームからの景色は何が見えますか？ (What is the view from the Premium Family Room?), 近くに夕食をとれる場所はありますか? (Is there a place to have dinner nearby?).

The sampled conversations all contain at least 11 user messages, with an average of 16.3. The complete distribution of message lengths and messages per conversation is shown in Figure 5.

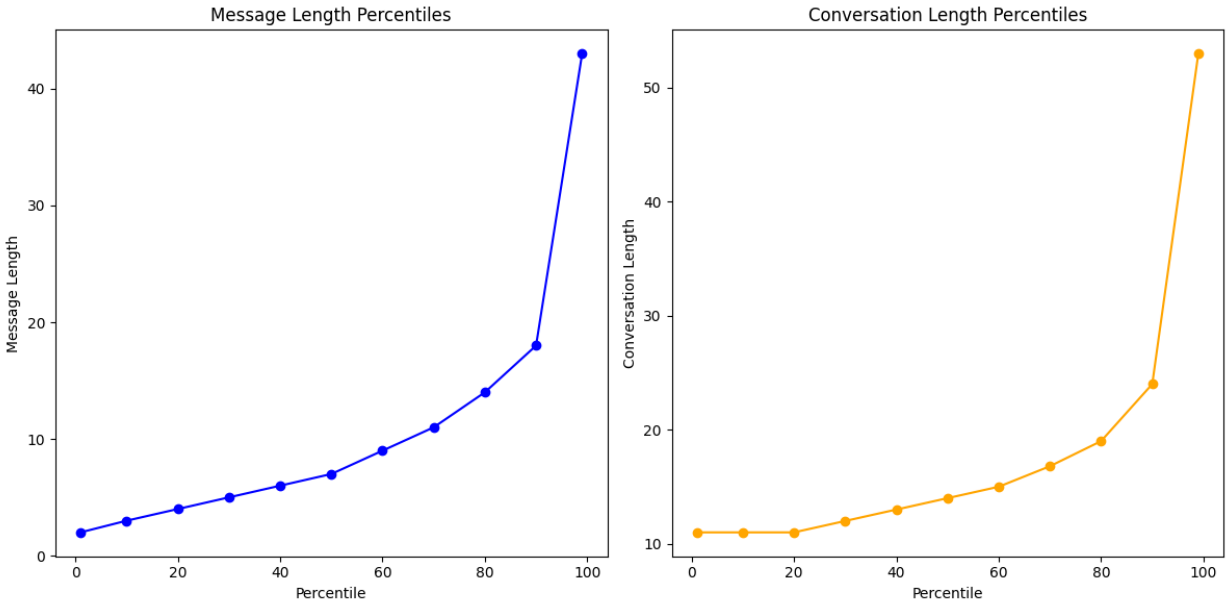

Figure 5 – Distribution of message length and conversation length per 10% (excluding the 1% extremes)

6.2. State of the Art for Sentiment Analysis

For this analysis, we kept in mind that our dataset is limited in terms of size. Currently, one of the most common and well-known method for achieving effective sentiment analysis is by creating or using a model based on BERT[1]. BERT is a model designed to be fine-tuned, retaining its base English understanding while modifying the output for a specific task. This approach allows us to reduce the amount of data required for training compared to creating a model from scratch, making it ideal for smaller datasets. This solution outperforms older methods, such as using a CNN neural network[2], which we will therefore not focus on for this sentiment analysis.

Another recent alternative is using corporate LLMs with a very large number of parameters, such as ChatGPT, to perform sentiment analysis via in-context learning. This has the significant advantage of not requiring a large training dataset, as it leverages the existing knowledge of the LLM. This type of usage can yield results comparable to a BERT model[3].

For these reasons, we will explore both a solution using a fine-tuned BERT model and one using the ChatGPT API. We will also compare the results of ChatGPT with other corporate models to determine if any perform better than others.

6.3. Multilanguages challenge

As specified in the previous Section 6.1, Tripla needs to find a solution to be compatible with all four languages in which Tripla delivers services. While solutions using corporate LLMs such as those from Google or OpenAI already support different languages, a solution using a fine-tuned local model requires some reflection on how to resolve this issue. This chapter will discuss three possible solutions, compare each of them, and explain which option was chosen for this project. Those solutions are described from input to the sentiment label. The way the actual labels are going to be used is out of scope for this chapter.

One model per language

The first solution is the easiest one in terms of concept but might be the most costly in terms of implementation. It involves training four different models, one for each language, as shown in Figure 6.

Here is a list of tasks to implement such a solution:

1. Analysis, choice, and implementation of a language detection module. This module will have to run at the start of the pipeline to direct the message to the appropriate model.
2. Analysis and choice of a trainable text classification model that yields good results in sentiment analysis benchmarks for all four languages.
3. Finding datasets containing content labeled for sentiment analysis in all four languages with a similar methodology of labeling.
4. Training all four models with their respective data and optimizing each one individually.

Having a model for each language allows it to be specialized in a specific language. This is a significant advantage because not only do languages have different vocabularies, grammars, alphabets, etc., but each language is also related to a different culture and group of people, which will have differences in how they interact and, most importantly in our context, in how they express their sentiments and emotions.[4]

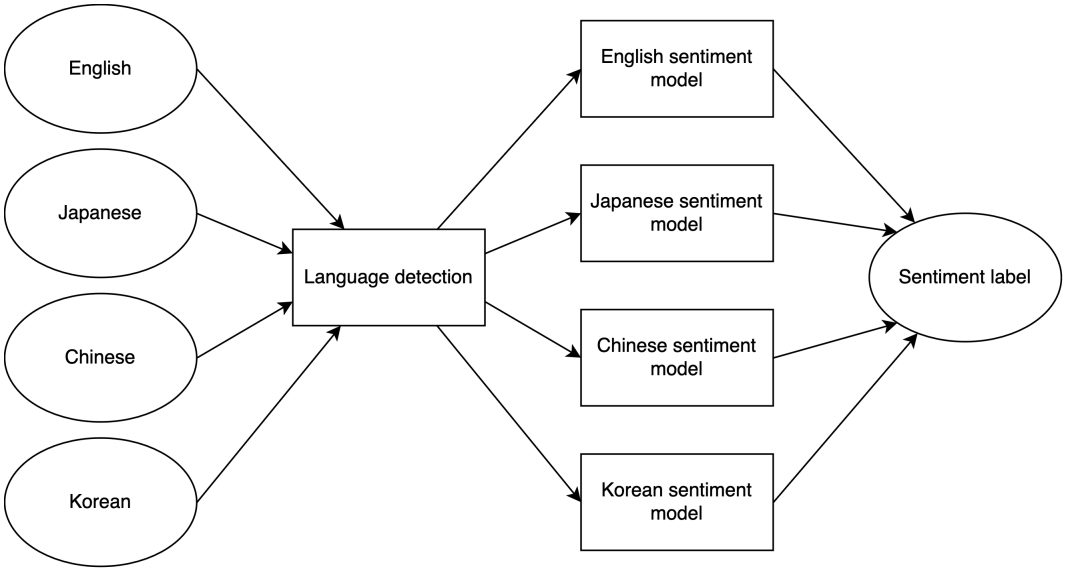

Figure 6 – Multi-models solution

One model that supports multiple languages

The second solution would be to use a single model trained with a dataset containing data from all four languages. This greatly simplifies the usage of sentiment analysis as we only have one model, allowing us to input all our data without any extra steps, as shown in Figure 7. Here is the list of tasks to implement this solution:

1. Analysis and choice of a trainable text classification model that yields good results in sentiment analysis benchmarks for all four languages.
2. Finding datasets containing content labeled for sentiment analysis in all four languages with a similar methodology of labeling.
3. Combining all the datasets to form a multilingual dataset with data from all four languages. Ensure to resolve any compatibility issues between datasets and balance the number of entries for each language.
4. Training the model and optimizing it.

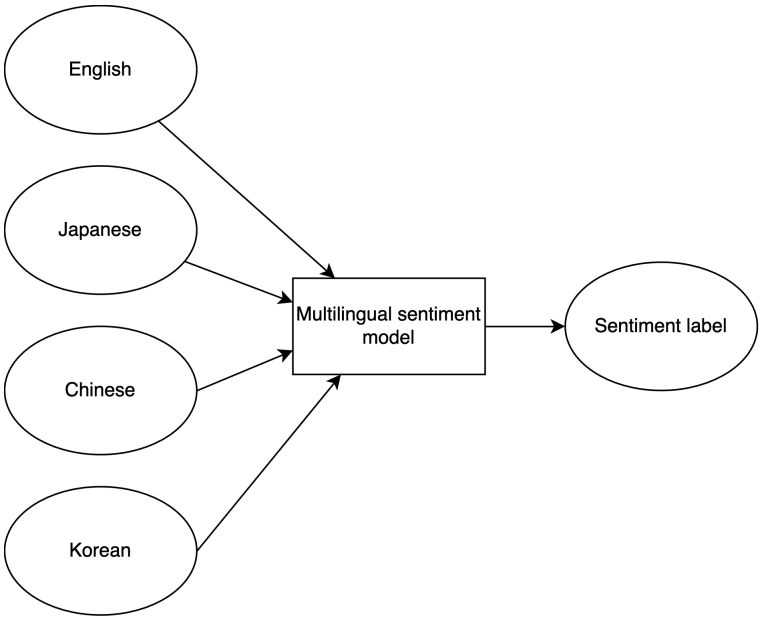

Figure 7 – Multilingual model solution

For the usage of Large coorporate LLM pretrained on multiple languages such as ChatGPT, this is typically the architecture we are going to use as the model is already able to handle multiple languages.

Use translation with monolingual model

The last solution is to translate text into a single language and use a single model trained on that language to perform sentiment analysis, as shown in Figure 8. We might lose some context and information due to the translation, but a recent study[5] found "that machine translation and sentiment analysis models can accurately analyze sentiment in foreign languages." This study used five languages, including English, Chinese, and Arabic, which are very different in their semantics and origins. Another study proved that translation from Japanese to English can be done with good accuracy[6]. With all this in mind, we can say such a technique is viable for the sentiment analysis of our chatbot. There are two languages that make sense for us to use as our main language. The first is Japanese, as most of our data is in that language. The second is English, as it has the most resources available, such as models, datasets, and research. English has been prioritized as it seems like the safest option for the project. It does mean that the label will only be in english and might need to be translated back depending on the use case of the label.

Here is the list of tasks for this method:

1. Analyze and compare different translation models.
2. Implement the translation module.
3. Analyze benchmarks for English models.
4. Find datasets in English with good sentiment annotation for our use case.
5. Train the model on the dataset and optimize it.

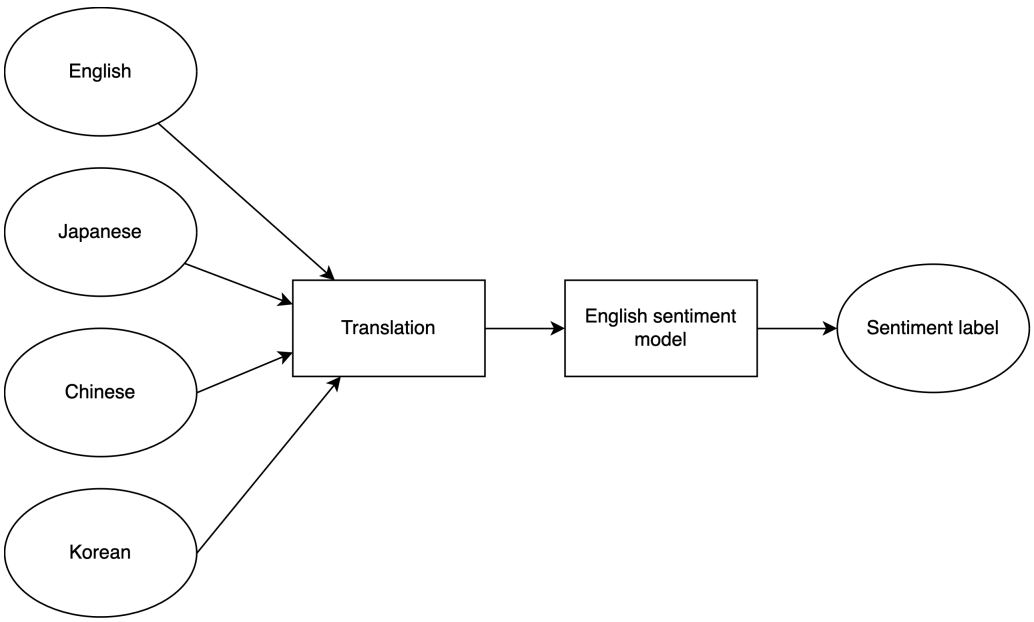

Figure 8 – monolingual model with translation solution

Comparison and choice

To choose which of the three techniques the project will use, let's compare several criteria (see Figure 9).

- Compute time: Detecting one of the four languages should be a trivial task, especially with only four languages each using a different alphabet. Therefore, the multi-model technique will be the fastest, involving computation for small models trained on individual languages. Next, the single multilingual model will have slower computation time due to handling all four languages. Finally, the translation method is the slowest as translating each message will consume more time than the actual sentiment analysis. Faster computation time leads to lower operational costs, which is crucial for handling large datasets such as the one we are working with.
- Documentation/models/dataset availability: The majority of current sentiment analysis systems are designed for a single language, typically English [7]. This limitation is also reflected in available benchmarks and research papers [8]. Although there is some documentation on multiple languages, it may not cover all four languages. For instance, a study like "Massively Multilingual Corpus of Sentiment Datasets and Multi-faceted Sentiment Classification Benchmark" [9] covered 27 languages but lacked data for Korean, with only one Japanese dataset compared to 17 English datasets. Given the time and resources available for a bachelor's project, having a robust base of documentation and resources is crucial.
- Ease of implementation / development time: Acquiring datasets for all four languages, training, and optimizing models will be time-consuming. Simplifying the problem to a single language and using translation will be much easier in comparison.
- Expected accuracy: A multilingual model will inherently sacrifice prediction accuracy because parameters are shared across languages. The translation method, while viable, introduces potential inaccuracies due to cultural context loss and translation errors. The highest accuracy would be achieved by using separate models for each language.
- Low risks: This criterion assesses the risk of encountering development issues such as time constraints, dataset availability, or model training difficulties. A single English model presents the lowest risk, given extensive prior projects and fewer potential points of failure compared to multi-model approaches.

	one model multilingual	multi model	Translation
Compute time	2	3	1
Existing model/dataset /docs	1	2	3
Ease of implementation / dev time	2	1	3
Expected accuracy	1	3	2
Low Risks	2	1	3
Result / 18	8	10	12

Figure 9 – Comparison of all three techniques. Each technique is ranked from 3 (best) to 1 (worst) for each criterion. The rankings are summed to provide a final score out of 18.

Considering these criteria, the preferred technique to train a model is using english translation. However, if issues arise such as execution time concerns or significant loss of context during trans-

lation, we will consider developing a second model specifically for Japanese, given the abundance of chatlogs in that language.

Translation model

As we have decided to use translation to English, it is important to analyze the currently available translation techniques.

To select the model, we considered several Japanese to English services, including corporate APIs like DeepL and locally available LLM models such as those from Helsinki-NLP available on Hugging Face, to determine which one best suits our use case.

To rate different models, it is possible to evaluate them using the BLEU score, which measures how closely the translation of a model matches accurate human translations[10]. We can evaluate them on a sample of 2,500 sentences taken from the Tatoeba dataset[11], which contains translations in many languages. While this sample isn't fully representative, it can provide a general idea of the accuracy of each model.

Here is a list of models we considered:

- DeepL API: DeepL is considered state-of-the-art in machine language translation, and its accuracy has been validated by studies[6].
- Google Translate API: Google Translate is widely recognized as one of the most popular machine translation services and has been used in various studies[5].
- Helsinki-NLP/opus-mt-ja-en: Helsinki University has a research group on Hugging Face focused on natural language processing. They have developed several high-quality translation models, including one for Japanese to English[12].
- K024/mt5-zh-ja-en-trimmed: This model, based on MT5 and published by a user on Hugging Face, supports Japanese, English, and Chinese. It offers potential broad usage for over 99% of the chat content. While not affiliated with a renowned institution, it is worth considering for its capabilities.

We decided not to perform the actual evaluation as it falls somewhat out of the scope of the project and can be easily replaced later if required. Instead, we chose to use the DeepL API, which has already proven its efficacy in previous studies.

6.4. Polar Labeling

The chat traces available for our analysis do not offer any ground truth, so it was necessary to manually label some of them to evaluate the model and provide examples for few-shot learning or fine-tuning. As I do not understand Japanese, Chinese, or Korean, a translation service was required to comprehend the messages. The translation service used was DeepL, which, as mentioned in the analysis phase, provides great results and maintains the sentiment of each message with good accuracy. The labeling chosen for each message was as follows:

- 1 (negative): A label of 1 indicates that the user is dissatisfied with either the hotel service or the conversation with the Tripla bot, or wishes to contact a real human operator.
- 2 (neutral): A label of 2 means that the user is in a neutral state or that the message alone does not provide information about their mood.
- 3 (positive): A label of 3 indicates that the user is satisfied or grateful for the hotel chatbot/services.

Out of the 300 chat logs that were manually labeled, the distribution of labels is as follows:

- 1 (negative): 523
- 2 (neutral): 4,398
- 3 (positive): 30

The majority of messages are neutral. Many messages are simple questions or key phrases, such as チェックイン (Check-in), 風呂 (Bath), or 何台分の駐車場があるの？ (How many parking spaces are available?). These simple queries usually do not provide information about the user's satisfaction, which explains the high number of neutral labels (2).

There is a decent amount of negative labels (1), which usually represent complaints, requests for changes or cancellations, or a desire to switch from the chatbot to a real operator. Examples include: 騒音 (Noise), ログインパスワードと取引パスワードを混同してしまいました (I confused my login password with my transaction password.), 電話での質問は可能でしょうか？ (Is it possible to ask questions over the phone?).

The number of positive labels (3) is very low, occurring in only about one in every ten conversations. Positive sentiment is generally detected in messages where the customer expresses gratitude or appreciation after receiving satisfactory responses from the bot. Examples include: 承知しました。ありがとうございます。 (Noted. Thank you very much.), ありがとう (Thank you.), 請問我們抵達第一天可以先租好雪具 隔天直接使用嗎? 謝謝 (Can we rent our snow gear on the first day we arrive and use it the next day? Thank you.).

The labeling was performed manually using translation services and may contain personal bias. While it is not 100% accurate, it provides a good foundation for further analysis.

6.5. ChatGPT

The initial implementation of binary sentiment analysis utilizes the GPT-3.5-turbo-0125 model from OpenAI. We selected this model over the more advanced GPT-4 due to its significantly lower cost, which is reported to be ten times less expensive [13]. The higher cost of GPT-4 would not be financially advantageous for Tripla and would be difficult to justify. The pricing for GPT-3.5-turbo-0125 is $0.5 per 1,000,000 input tokens and $1.5 per output token. Our primary cost concern will be from the input tokens, as we only require a single character as output. The model was tested on 100 labeled conversations to determine the optimal input prompt, to then have a validation on 100 different conversation to control that the modification of the prompt weren't overfitting to the first 100 conversations.

In-Context Learning (ICL)

In-context learning (ICL) refers to the ability of language models (LLMs) to adapt their behavior and responses based on the context provided during interaction [14]. This context can range from zero-shot, where the model receives no explicit examples or instructions, to few-shot, where a small number of examples or prompts are provided (typically ranging from 1 to 20 examples), and finally to many-shot, also known as ICL+, where the number of examples can extend into the thousands [15].

The ability of LLMs to learn and adapt in this context-dependent manner is a powerful feature that allows these models to handle a wide range of tasks and domains with varying degrees of explicit guidance, making them versatile and adaptable tools for numerous applications.

In the zero-shot scenario, the LLM relies solely on its pre-trained knowledge and capabilities, which represent an impressive amount of information but may not be tailored to specific tasks or domains. To enhance output quality, the few-shot method can be employed, where a few examples are included within the prompt to provide more context to the LLM. This approach improves the accuracy of the LLM for specific tasks. The more examples included, the higher the accuracy of the query, although this also increases the length and the cost of the prompt.

Many-shot learning represents a newer capability available with large commercial models that have very large context windows of up to 1 million tokens. This allows thousands of examples to be included in the prompt, significantly boosting result accuracy by 5% to 35%, depending on the task. However, a drawback of this method is the substantial increase in the number of input tokens used.

For in-context learning, achieving the optimal balance between cost and improved accuracy for each example included is crucial.

Basic Prompt

To establish a baseline, we used a very basic prompt:

> "You are a sentiment analysis tool for chat logs from hotel customers. Please rate the statement from 1 to 3, where 1 indicates unsatisfied, 2 indicates neutral, and 3 indicates satisfied. Reply with only the number and nothing else."

Figure 10 displays the confusion matrix for this basic prompt. To evaluate the model, we use the average of the macro F1 score and the weighted F1 score [16], which provides a balanced measure between the importance of heavily represented classes (weighted) and every class (macro). This metric is more informative than overall accuracy alone. For instance, a model that predicts only the neutral class (2) would achieve an overall accuracy of 87%, but an average F1 score of only 57%. The score we refer to in this report as the overall F1 score reflects this balance.

For the basic prompt, ChatGPT achieved an overall F1 score of 64%, with a tendency to predict more positive labels. The results also showed a higher number of false negatives and false positives for label 1 compared to true positives. Those result are rather low and need to be improved.

Basic Prompt		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	57	73	37	34%	44%	38%
	2	73	1083	331	73%	93%	82%
	3	0	3	3	50%	1%	2%
					69%		64%

Figure 10 – Confusion matrix for the basic prompt

CO-STAR Prompt

To improve the performance of the LLM, it is crucial to craft a well-defined prompt that clarifies your intentions and expectations for the task. The CO-STAR framework, introduced by GovTech Singapore's Data Science & AI team [17], provides guidelines for creating effective prompts.

The CO-STAR framework includes six categories:

- **Context**: Provides background information about the scenario and the role the model is expected to play.
- **Objective**: Clearly states the purpose of the task to ensure clarity.
- **Style**: Specifies the desired writing style, whether personal or technical.
- **Tone**: Defines the manner in which the LLM should communicate.
- **Audience**: Describes who will use the output and how it will be used.
- **Response**: Defines the expected format for the answer.

Applying this structured approach helps to provide clear instructions to the model, reducing ambiguity and improving performance. Although some categories, such as tone and style, may not be directly applicable due to the single-word output requirement, the framework's pattern is still useful for enhancing prompt effectiveness.

Context
You are a tool to extract sentiment analysis from chatBot conversation between an hotel's customer and a hotel chatbot, able to answer question about the hotel.

Objective
Give a number from 1 to 3 that will represent the customer satisfaction about the hotel services and the chatbot. This number will be linked with one message of the conversation. a 1 means the client is not satisfied / unhappy with the hotel services or with the chatbot conversation and would like to speak to a real person. A 2 means the client is in a neutral state or that the message doesn't contains much information about the satisfaction of the client. A 3 means the client is satisfied or grateful for the service. The user message will be the statement that needs to be evaluated.

Style
Only reply with a single number 1, 2 or 3. Do not reply anything else than one of this 3 options.

Tone
Only give the answer

Audience
Those number will be used to detect automatically the mood of the user and change the flow of the chatbot when appropriate.

Response
Only an Integer, either 1, 2 or 3

It is important to note that using longer prompts such as this one will greatly increase the amount of input token used by our model, as the context prompt will need to processed for every messages. We see that while the result (Figure 11) did get modified, it is not a big improvement, we mostly have a shift of prediction towards the negative label. The explanation might our basic prompt did not contain that much complexity to begin with, making it easy enough for the LLM to understand.

CO-STAR		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	67	88	10	41%	34%	37%
	2	129	1109	244	75%	92%	83%
	3	0	2	4	67%	2%	3%
					71%		64%

Figure 11 – Confusion matrix with CO-STAR prompt

English Usage

ChatGPT has been trained to understand multiple languages, but are its performance and accuracy consistent across all languages? To investigate this, sentences translated into English were input into ChatGPT instead of the original Japanese text, while keeping the prompt unchanged. The results, shown in Figure 12, reveal that there are barely any differences. Therefore, we can continue to use the original language of the messages while predicting with ChatGPT.

CO-STAR + english		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	62	80	23	38%	44%	40%
	2	80	1112	290	75%	93%	83%
	3	0	1	5	83%	2%	3%
					71%		65%

Figure 12 – Confusion matrix with CO-STAR prompt and English input

Multi-shots with CO-STAR

The next step in improving performance involves providing ChatGPT with examples in the prompt. The examples selected are from cases where the model previously produced incorrect labels.

Here is the list of examples given to ChatGPT in the prompt: メニュー変更：1
電話での質問は可能でしょうか？：1
飲み放題の値段：2
スパは当日受付ですか？：2
承知しました。ありがとうございます。：3
I want to pay now：1
騒音：1
安いお部屋：2
雨の日でも屋外プール遊べる？：2
調味料：2

The results indicate improvement, with the F1 score increasing by 5% and accuracy by 15% (see Figure 13). However, results for label 1 are still suboptimal, with more false positives and false negatives than true positives. This is likely because we are operating with examples that are quite distant from the context of our actual conversations, making it challenging for the LLM to accurately assess the content based on just a few words.

CO-STAR + 10 shots		Predicted			Accuracy	Recall	F1
		1	2	3			
Expected	1	65	99	1	39%	47%	43%
	2	72	1353	57	91%	93%	92%
	3	0	2	4	67%	6%	12%
					86%		69%

Figure 13 – Confusion matrix with CO-STAR prompt and multi-shot examples

Multi-shots without CO-STAR

It is important to confirm that every part of our prompt is valuable to the llm. While we have now a more complex prompt including examples, would the use of CO-STAR be useful this time ? We tried simplifying the prompt not respecting the proper CO-STAR framework anymore.

Context
You will get user messages from a conversation between an hotel's customer and a hotel chatbot.

Objective
Give only a number from 1 to 3 that will represent the customer satisfaction about the hotel services and the chatbot. a 1 means the client is not satisfied / unhappy with the hotel services or with the chatbot conversation and would like to speak to a real person. A 2 means the client is in a neutral state or that the message doesn't contains much information about the satisfaction of the client. A 3 means the client is satisfied or grateful for the service.

Here are some examples :
input : メニュー変更, output : 1
電話での質問は可能でしょうか？ : 1
飲み放題の値段 : 2
スパは当日受付ですか？ : 2
承知しました。ありがとうございます。 : 3
I want to pay now : 1
騒音 : 1
安いお部屋 : 2
雨の日でも屋外プール遊べる？ : 2
調味料 : 2

We see an interesting result (see Figure 14) as the model seems to have gotten a lot more pessimistic for some reason, making it's accuracy for label negative (1) a lot better but in counterpart, the recall of label 1negative (1) and the accuracy of label neutral (2) dropped. This creates a drop of the overall f1 score of 3% and a drop of accuracy of 14%, showing the advantage of following the proper structure of the prompt to properly explain our intents to the LLM.

Basic + 10 shots		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	100	62	3	61%	21%	31%
	2	371	1089	22	73%	94%	83%
	3	0	2	4	67%	14%	23%
					72%		66%

Figure 14 – Confusion matrix with shorted prompt and multishot

Validation and Result

Table 0 summarizes the performance of each prompting technique. The CO-STAR with 10 shots method performs the best by a significant margin.

Method	Accuracy	F1 Score
Basic	0.69	0.64
CO-STAR	0.71	0.64
CO-STAR EN	0.71	0.65
CO-STAR 10 shots	0.86	0.69
10 shots	0.72	0.66

Table 0 – Performance comparison of ChatGPT with different prompts

Having determined that the multi-shot CO-STAR prompt is the best performing, it was used to predict labels for another 100 conversations, serving as a validation set and for comparison between all models. Figure 15 shows similar results with this validation dataset compare to the original 100 conversations, demonstrating that the prompt is not getting overfitted on the original 100 conversations.

CO-STAR + 10 shots (val)		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	72	109	0	40%	46%	42%
	2	86	1219	28	91%	91%	91%
	3	0	7	4	36%	13%	19%
					85%		68%

Figure 15 – Confusion matrix with shortened prompt and multi-shot

While ChatGPT offers a straightforward implementation for sentiment analysis, the results for our specific use case, where sentiment is often subtle and conveyed in short messages, are not as strong as they might be in scenarios with more explicit sentiment expressions.

6.6. Comparison with Other LLM Models

It is valuable to examine whether other popular corporate LLMs might provide significantly better results. This section briefly explores Mistral and Claude as alternatives to ChatGPT, using the same prompt as the best performing one for ChatGPT.

Mistral

Mistral AI is a French company offering LLMs both through an API and as open-source models. However, Mistral presents two major drawbacks that led to its elimination as a viable alternative:

- **Rate Limits**: Mistral AI imposes rate limits that restrict the ability to efficiently process multiple conversations.
- **Language Support**: Mistral AI focuses on European languages (English, French, Italian, German, and Spanish), making it less suitable for our use case, which requires support for Japanese, Chinese, and Korean.

Claude

Claude is an AI agent developped by Anthropic, a recent AI tech company based in the US. Claude, like ChatGPT, regroups multiple models with different cost and efficacity. With the accuracy of a low cost model chat-gpt 3.5 being not so great, we decided to try with a higher quality model for Claude : Claude 3.5 Sonnet[18]. We used the free tier of their api access according 5$ of free tokens with big rate limits in order to test the model. We used the same prompt as the best performing one for chat-gpt, aka the CO-STAR + 10 shots prompt. The result of the model on the validation

set in Figure 16 shows a very good accuracy on label 2 of 99% leading to a high overall accuracy. But the other categories are below 50% accuracy, dropping the F1 score to an overall 74% score. We can evaluate this model to be better than GPT-3.5, which makes sense as the pricing is a lot higher than GPT-3.5 with 3$ per 1'000'000 input token and 15$ per 1'000'000 outputs tokens[19]

CLAUDE		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	70	104	0	40%	82%	54%
	2	15	1454	1	99%	93%	96%
	3	0	6	1	14%	50%	22%
					92%		74%

Figure 16 – Confusion Matrix with Claude 3.5 sonnet model (claude-3-5-sonnet-20240620)

GPT-4o

While we already discussed about why we would prefer to use GPT-3.5 instead of the better but more expensive model GPT-4o in chapter 6.5. It is still interesting to see if 4o is indeed better at this sentiment analysis task compared to his younger brother 3.5. The result in Figure 17 show very similar result to Claude Sonnet, which are better than GPT-3.5.

GPT 4o		Predicted					
		Negative (1)	Neutral (2)	Positive (3)	Accuracy	Recall	F1
Expected	Negative (1)	74	100	0	43%	77%	55%
	Neutral (2)	22	1445	3	98%	93%	96%
	Positive (3)	0	5	2	29%	40%	33%
					92%		76%

Figure 17 – Confusion Matrix with GPT-4o model

6.7. Finetuning

This chapter explore a different approach for the polarity message-by-message sentiment analysis, which is the training of a local finetuned model.

Analysis

One commonly used technology is machine learning through fine-tuning an existing model. But what does this exactly mean, and what are the upsides and downsides of this technique?

Firstly, let's understand how models are trained. A model consists of layers of parameters arranged in a network. Each parameter receives inputs from the previous layer and produces outputs for the next layer. During training, each layer initially has randomly set weights for its outputs. Through a process called backpropagation, these weights are adjusted to better fit existing data. In simpler terms, backpropagation modifies each parameter to make the model's predictions closer to the correct answers, progressively adjusting parameters layer by layer. This method is computationally

intensive, especially for large models where the number of weights can reach into the billions, and it requires a substantial dataset.

Fine-tuning simplifies this process. Instead of starting from scratch, fine-tuning utilizes an already trained model with pre-existing weights. This pretrained model is then further trained on a new dataset, adjusting the weights of connections between layers to better fit the new dataset's data. This adjustment primarily affects the later parts of the model, altering its outputs, while the earlier parts remain largely unchanged (see Figure 18). This allows us to leverage the language comprehension capabilities of the existing model while adapting its outputs to suit our specific needs [20].

Some alternatives exist where the earlier parts of the model are frozen or where additional layers are added to the end of the pretrained model.

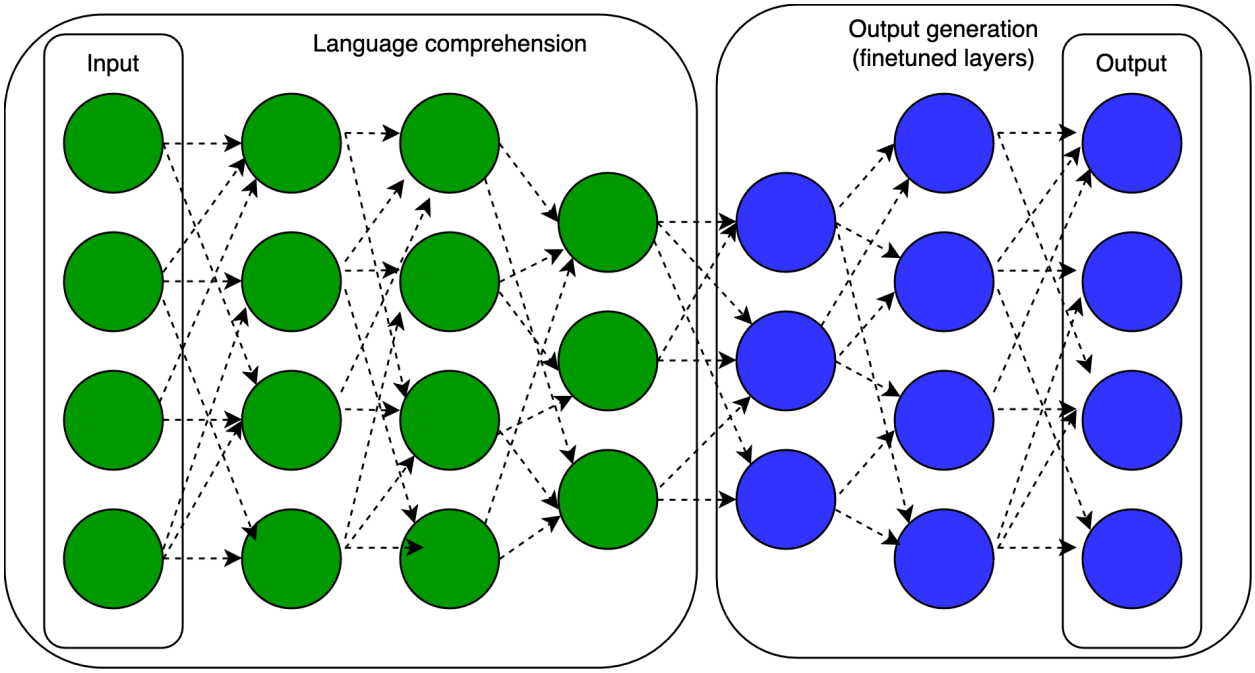

Figure 18 – Illustration of a model. When is is Finetuned, the part towards the end that has a lot of effect on the output will be greatly modify (Blue), while the part towards the start responsible of understand the input will not be modified and only slightly (Green)

Implementation

To Implement the finetuning, the dataset got split into 3 sets : a training set (150 conversations), a testing set (50 conversations) and a validation set (100 conversations). The library transformers of hugging face[21] was used for the training and the base model used was 'Bert-base-uncase'[1]. This model main purpose is to have a base understanding of english that will then be used for a specific tasks, in our case a sentiment analysis. Bert-base-uncase is a popular model for finetuning a sentiment analysis, here is an example of an existing model with bert as a based : finiteautomata/bertweet-base-sentiment-analysis[22]. While using the base dataset, the following results in Figure 19 were obtained on the validation set.

Finetuned		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	46	135	0	25%	62%	36%
	2	28	1305	0	98%	90%	94%
	3	0	11	0	0%	0%	0%
					89%		65%

Figure 19 – Confusion matrix finetuned model. Label 1 is negative, label 2 is neutral, label 3 is positive. Green represents true positive and red represents wrong prediction, the darker the red the more frequent this false negative is in proportion to true positive

This model basically almost only predicts the label neutral (2) as the very large majority of the labels are neutral (2) and the other labels are lacking examples. Therefore the model will very likely get the right response during the training/testing if it guesses a neutral (2), regardless of the input. We still do see a few negative (1) predicted with a good recall rate.

Data Augmentation

The data is highly unbalanced with very few samples for labels negative (1) and positive (3). While it is typical to have more neutral labels than positive and negative, it's crucial to balance these classes by creating new examples to prevent the model from biasing towards neutral predictions and to avoid overfitting, which can lead to poor performance on the validation set.

Initially, additional positive examples were generated since there were only 16 existing examples in the training set, which increased to 70 after augmentation. This was achieved by feeding all existing label 3 examples to ChatGPT, asking it to generate extra examples and selecting realistic ones from the generated responses of varying lengths.

Following this, traditional natural language data augmentation techniques were applied to labels 1 and 3. These techniques included back translation (into Chinese, Korean, Russian, Portuguese, or Arabic), random word swapping, and random word deletion. For messages with fewer than three words, random word swapping and deletion were avoided to prevent excessive information loss.

A second round of augmentation involved using ChatGPT to replace one to two words per message with synonyms. This technique leverages the capabilities of LLMs to generate additional training examples for a smaller model.

The chart in Figure 20 illustrates the proportion of each label in the training set after each aug-

mentation step. While there is still a predominance of neutral labels, the quantity of labels 3 and especially 1 are now frequent, allowing the model to tune its parameters for those labels more frequently. It's important to note that only the training and testing sets were augmented, as the validation set serves as an evaluation metric and needs to remain accurate to the original data [23].

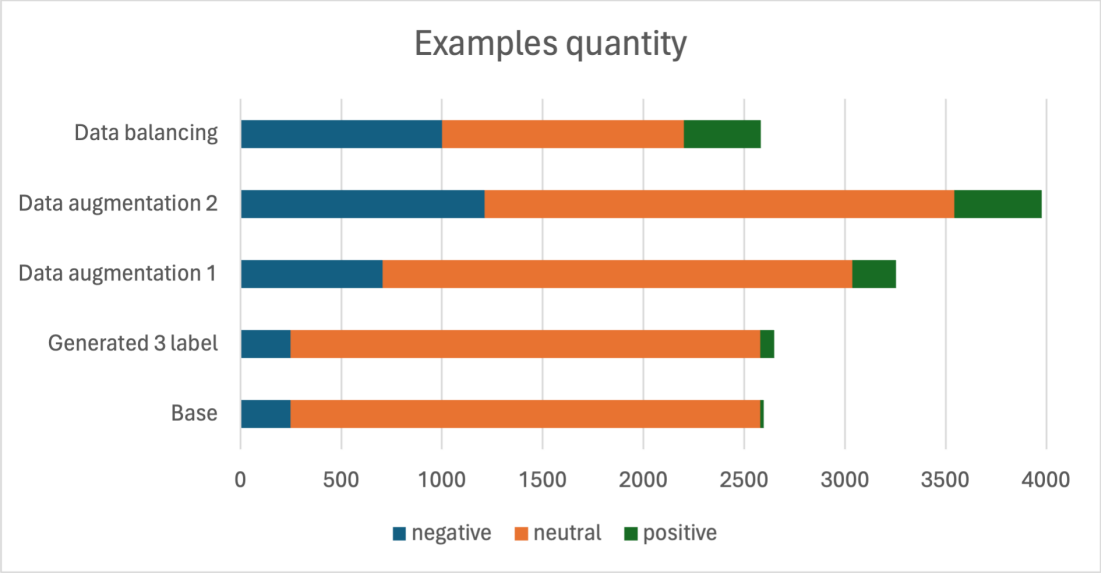

Figure 20 – Quantity of each label in the training set initially, with the generation of label 3, with the first data augmentation and the second data augmentation and finally with some data balancing on the augmented data.

With the first augmented dataset (backtranslation, random swap, random delete), the result in Figure 21 shows a large increase in quality of the F1 score. The model no longer almost only predict the label 2 as the number of entry of different labels increased. There are still a lot of false negative for the negative (1) label.

Finetuned + Augm. data		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	80	98	3	44%	69%	54%
	2	36	1289	8	97%	93%	95%
	3	0	3	8	73%	42%	53%
					90%		78%

Figure 21 – Confusion matrix finetuned model on augmented dataset (step 1)

By augmenting the label negative (1) and positive (3) even more by using synonyms swaps, we get an increase in the result of prediction of the label negative (1), augmenting its accuracy. Surprisingly, the label positive (3) decreased its accuracy, but this can be explained by the amount of sample of that label being too low and therefore having a high variance. (see Figure 22).

Finetuned + Augm. Data 2		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	107	71	3	59%	67%	63%
	2	52	1275	6	96%	94%	95%
	3	1	5	5	45%	36%	40%
					91%		78%

Figure 22 – Confusion matrix finetuned model on augmented dataset (step 2)

Data balancing

After this data augmentation, the label positive (3) is still under represented compare to the other 2 label, in order to counter this, we could do some data balancing by removing some entry randomly from label negative (1) and neutral (2) in order to drop them to less than 1200 for label 2 and 1000 for label 1. While not giving equality between each labels (see Figure 20), this would make a better balance and would give an idea if perfect balance would be worth to try by observing the trend. The result in Figure 23 shows that we increased the accuracy of the label with the least amount of samples (positive (3)) but at a slight cost for label negative (1). The results are close enough to not be significantly different, but we will continue with this model as it does have the best F1 score.

Finetuned + Augm. Data 2 + Balanced		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	97	81	3	54%	64%	58%
	2	54	1274	5	96%	94%	95%
	3	0	3	8	73%	50%	59%
					90%		80%

Figure 23 – Confusion matrix finetuned model on augmented and balanced dataset

Comparison with Out-of-the-Box Model

To validate the results, it's useful to compare the fine-tuned model with an existing sentiment analysis model that has been pre-trained on a different dataset. For this comparison, we selected the 'finiteautomata/bertweet-base-sentiment-analysis' model [24] available on Hugging Face. This model, trained on tweets from Twitter (now X.com), is well-suited for short text sentiment analysis, like our conversation messages, making it a reasonable choice for comparison despite the different context.

However, there are notable limitations to using this pre-trained model. The sentiment expressed in tweets can differ significantly from that in chatbot messages, which may affect the model's performance on our specific dataset.

The results from using the out-of-the-box model are shown in Figure 24. This model primarily predicts neutral sentiment and struggles with detecting negative sentiments, resulting in a relatively low F1 score of 67%, which does demonstrate the importance of the context when performing sentiment analysis.

Out of the box		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	33	147	1	18%	73%	29%
	2	12	1292	29	97%	89%	93%
	3	0	7	4	36%	12%	18%
					87%		67%

Figure 24 – Confusion matrix for the out-of-the-box sentiment analysis model.

6.8. Embedding Messages with Neural Network

In Section 7 of the report, we will explore an alternative solution that involves using a neural network with text embedded as input. This approach is particularly effective for scenarios where the input is complex and limited, such as entire conversations. While this method is more suited for the 300 complex examples we have compared to the 5000 simpler examples used in embedding messages, it is still valuable to assess its performance for polarity analysis.

For this experiment, we developed a neural network model that uses text embeddings generated by OpenAI's "text-embedding-3-large" model from each message of the conversation [25]. The training and testing data were the same as those used for the augmented and balanced dataset (see Chapter 6.7).

The results, shown in Figure 25, reveal that this model performs similarly to the fine-tuned model, with only a 1% lower accuracy and a 2% lower combined F1 score. However, a significant drawback of this approach is the need for an API call to OpenAI for each message, which can be costly. Given these factors, the fine-tuned model is superior due to its lower cost and slightly better performance.

Neural Network		Predicted					
		1	2	3	Accuracy	Recall	F1
Expected	1	99	77	5	55%	56%	55%
	2	78	1248	5	94%	94%	94%
	3	0	4	7	64%	41%	50%
					89%		78%

Figure 25 – Confusion matrix for the neural network model using embedded input for polarity analysis.

6.9. Recap

In the table 1, we see the accuracy and the overall F1 score of every different technique explored. We can observe the better result of the more complex LLMs models compared to GPT-3.5 We can also see the difference between the finetuned Bert model on our dataset performing a lot better than the Out-Of-The-Box model trained on a different sentiment analysis dataset.
The best model for the polarity sentiment analysis in our use case in the finetuned Bert model. It has the best F1 score and doesn't require to use an LLM through an API. The result is great and definitely usable for a future use case within Tripla's software.

Model	**Accuracy**	**F1 Score**
GPT-3.5	0.85	0.68
Claude Sonnet	0.92	0.74
GPT-4o	0.92	0.76
Bert	0.90	0.80
Out-Of-The-Box	0.87	0.67
Embedding NN	0.89	0.78

Table 1 – Performance comparison of different models

7. Conversation Sentiment Keyword Analysis

This approach shifts the focus from analyzing individual messages to evaluating entire conversations. The aim is to identify keywords that capture the overall sentiment of the conversation, as determined by a language model (LLM). See Section 5.2 for an overview of the different sections in this chapter.

7.1. Keyword Determination

To identify relevant keywords related to sentiment in conversations, we used the ChatGPT API (model GPT-4o) to recommend keywords for each conversation. These recommendations were then clustered to select the most pertinent keywords.

Prompt Determination

The first step involved developing an effective prompt to extract keywords from conversations. To achieve this, seven conversations with clear sentiment were selected. The keywords suggested by ChatGPT were compared to the expected sentiment of each conversation. The evaluation criteria were as follows:

- **0**: The list of adjectives does not reflect the sentiment of the conversation at all.
- **1**: The list of adjectives partially describes the sentiment of the conversation.
- **2**: The general sentiment is captured, but some elements are missing.
- **3**: The list of adjectives accurately describes the sentiment of the user during the conversation.

The expected sentiments for the conversations were summarized as follows:

- Confused/unsatisfied with registration but grateful for the service.
- Neutral.
- Frustrated/angry about noise.
- Unsatisfied, concerned about meal options and their pet.
- Lost, annoyed, chatbot not providing the correct responses.
- Confused, irritated by the inability to contact them by phone.
- Grateful but slightly worried about payment procedures.

The initial prompt iteration was based on the CO-STAR framework discussed in Section 6.5. The prompt instructed the LLM to provide a list of adjectives describing each conversation. All conversations were provided within the same prompt, and the output consisted of approximately 3-6 adjectives per conversation.

Context
You are a tool to extracts sentiment from messages of an hotel customer to a chatbot, the conversation will be in japanese, english, chinese or korean, you will get messages send by user from several conversation # Objective #
Give a list of keyword describing how the customer is feeling during each discussion, it needs to be sentiment.

Style / Tone
Reply only with the list of word selected. Use only english words describing sentiments

Audience
Those keywords will be used to automatically alter the chatbot comportement when using the chatbot.

Response
Start your reply with the list of keywords, after that you can give a small explanation of keywords chosen for each conversation (up to 3 sentences of explanation per conversation).

---

1. Curious, Eager, Confused, Grateful, Concerned, Anxious (2/3)
2. Inquisitive, Interested, Concerned (0/3)
3. Frustrated, Annoyed, Complaining (3/3)
4. Concerned, Caring, Inquisitive (1/3)
5. Inquisitive, Concerned, Seeking Guidance (1/3)
6. Frustrated, Disappointed, Eager (2/3)
7. Inquisitive, Grateful, Anxious (3/3)

The total score from these seven conversations was 12/21, averaging 1.7/3.

Following this, several iterations of the prompt were made to improve keyword extraction:

- **Prompt Refinement**: Avoided using terms like "curious" or synonyms, as they were frequently misinterpreted due to the nature of user inquiries.
- **Enhanced Context**: Provided more context about the flow of conversations to help the model better understand and interpret the exchanges.
- **Detailed Instructions**: Clarified the concept of sentiment and outlined the steps the model should follow to extract keywords, improving accuracy in sentiment classification.

- **Explanation Request**: Added a request for a brief explanation alongside the keywords. This approach was hypothesized to reduce hallucinations and ensure more accurate keyword suggestions.

Figure 26 shows the evolution of the scores across different iterations of the prompt. The improvements in accuracy can be attributed to the refinements made in the prompt design.

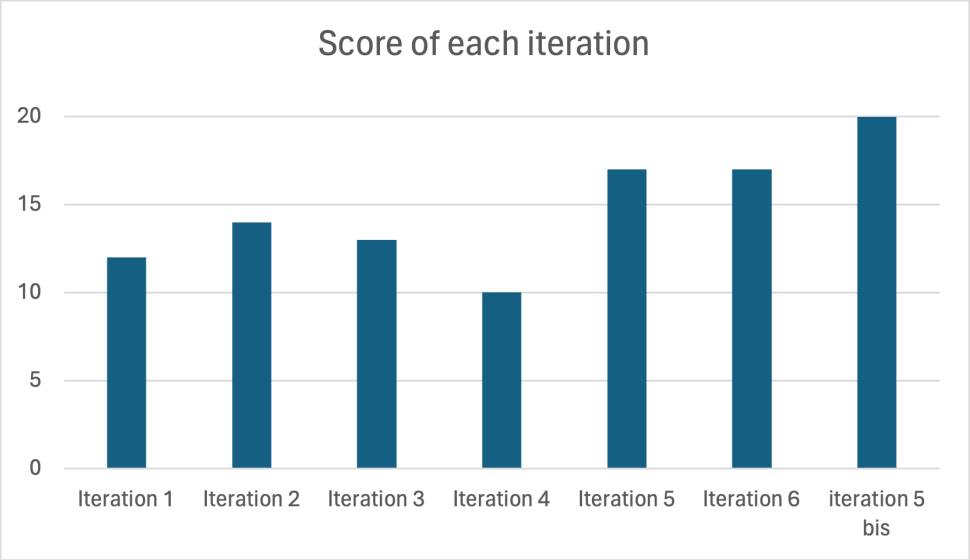

Figure 26 – Score of each iteration of prompting out of the total 21

Additionally, in Iteration 5 bis, the prompt was applied to individual conversations one at a time rather than processing all seven in a single request. We observed that the LLM had an easier time processing one conversation at a time.

These iterative refinements have led to better performance in identifying relevant sentiment keywords, preparing the following prompt to extract sentiment from every conversation :

Context
You are a tool to extract sentiment from messages of an hotel customer to a chatbot, the conversation will be in japanese, english, chinese or korean, you will get messages send by user from several conversations. The chatbot reply by selecting one of a list of possible answer, a repetition of same or similar question is usually explained by the user not getting the answer wanted by the chatbot.

Objective
Give a list of keyword describing how the customer is feeling during each discussion, it needs to be sentiment. The keywords can each describe positive, negative or neutral emotions. Start by understanding what the user intent is with each messages, are they satisfied with the way the conversation is going, with the hotel service? Then put everything together to understand their emotion during the entire conversation. Do not include keywords that explain that the customer is curious or that he ask questions. This is obviously the case when someone use the chatbot.

Style / Tone

Reply with the list of word selected. Use only english words describing sentiments, use a formal but easy to understand style.

Audience
Those keywords will be used to automatically alter the chatbot comportement when using the chatbot.

Response
Start your reply with the list of keywords, after that you can give a small explanation of keywords chosen (up to 3 sentences of explanation). DO NOT PUT "CURIOUS" OR A SYNONYM IN THE LIST OF KEYWORDS.

The prompt gave the following result for a total score of 20/21 for an very good average of 2.85/3:

1. Grateful, Dissatisfied, Confused (3/3)
2. Neutral, Inquisitive (3/3)
3. Frustrated, Annoyed (3/3)
4. Annoyed, Frustrated (2/3)
5. Confused, Needing Assistance, Unsure(3/3)
6. Frustrated, Annoyed, Confused(3/3)
7. Appreciative, Anxious, Reassured (3/3)

List of keywords

After using the prompt, 200 conversation are input into the prompt decided on the previous chapter. Afterward, a second prompt extracts the list of keywords from the output in order to have an usable json format. Here is the entire list of keyword extracted by chatgpt and the amount of time they appeared :

Confused: 84,
Frustrated: 83,
Inquisitive: 67,
Neutral: 60,
Concerned: 33,
Unsatisfied: 31,
Anxious: 26,
Uncertain: 19,
Impatient: 14,
Curious: 12,
Annoyed: 12,
Interested: 11,
Polite: 6,
Unsure: 6,
Persistent: 6,
Satisfied: 5,
Hopeful: 5,
Detailed-oriented: 4,
Grateful: 3,
Calm: 3,
Determined: 2,
Needy: 2,
Worried: 2,
Patient: 2,
Excited: 2,
Appreciative: 2,
Eager: 2,
Engaged: 2,
Expectant: 2,
Inconvenienced: 1,
Resistant: 1,
Seeking clarification: 1,
Inquiring: 1,
Happy: 1,
Anticipative: 1,
Detail-oriented: 1,
Cautious: 1,
Unanswered: 1,
Unclear: 1,

Positive: 1,
Anticipatory: 1,
Frustration: 1,
Impatience: 1,
Confusion: 1,
Uncertainty: 1,
Eagerness: 1,
Inquiry-driven: 1,
Detached: 1,
Unconcerned: 1,
Scared: 1,
Disappointed: 1,
Detailed: 1,
Upset: 1,
Hesitant: 1,
Seeking reassurance: 1,
Doubtful: 1.

We can observe some similar adjectives such as unsure and uncertain or anxious and concerned. There are also some keywords that comes up with a very high frequency. The next step is to find a list of category of sentiment with similar meaning in order to give a list of keyword that can be detected in a prompt.

7.2. Keyword grouping

Moving beyond ternary classification (negative, neutral, positive), we aim to derive specific sentiment categories from the keywords obtained by asking chatgpt to give some corresponding to each of the 200 conversations.

The first method attempted for effective category selection is clustering. This process requires transforming keywords into vectors that represent their meaning. For example, in a simplified view, we could vectorize an object with dimensions representing size, color, shape, etc.

Multiple solutions are available for this transformation, such as GloVe (Global Vectors for Word Representation)[26]. GloVe analyzes how frequently words appear together in large text corpora, identifying words with similar contexts and meanings. It then uses these occurrences to create vectors that reflect the meanings and relationships between words, a process called vector embeddings. For instance, words like "king" and "queen" or "apple" and "fruit" end up with similar vectors. The assumption is that adjectives with similar meanings will have similar vectors, and by creating clusters, categories would emerge.

Another solution is to use an embedding Large Language Model (LLM). This approach differs from GloVe as it doesn't rely on a dictionary of pre-determined words. Instead, the LLM tokenizes the input string, understands its meaning using principles similar to a regular generative LLM, and then produces a vector representing the string and its significance. The main difference between the two methods is that GloVe examines word placement in text databases and their frequent co-occurrences to position common words close together. In contrast, LLM embedding employs the transformer architecture to comprehend the input and transform it into a vector[27][28].

For this project, we will use the model text-embedding-3-large from OpenAI[25], as one of our objectives is to explore LLM functionality and utility. This model represents words in 3072-dimensional vectors. Such high dimensionality introduces considerable noise, necessitating dimension reduction to an optimal level. This reduction ensures that only the most relevant information for differentiating between keywords is retained, facilitating more effective clustering.
Here are the steps required to perform the clustering:

- Create a vector embedding of each extracted keyword by ChatGPT from the 200 conversations.
- Generate the similarity matrix using the cosine similarity, which represents how close every

vector is to each other on a scale of 0 to 1.

- Find the optimal dimension reduction that will be a good compromise between reducing noise and retaining maximum useful information. To achieve this, one can either select a portion of the vector dimensions or use an algorithm that reduces the dimensionality of the vectors. For this clustering, we chose, after some quick experimentation, to use MDS (Multidimensional Scaling)[29]. This method uses the similarity matrix and minimizes the information loss created by dimensional reduction, measured via the stress value. We can then use the number of dimensions that gave the smallest loss. In this case, it was 130 dimensions, as we see in Figure 27.

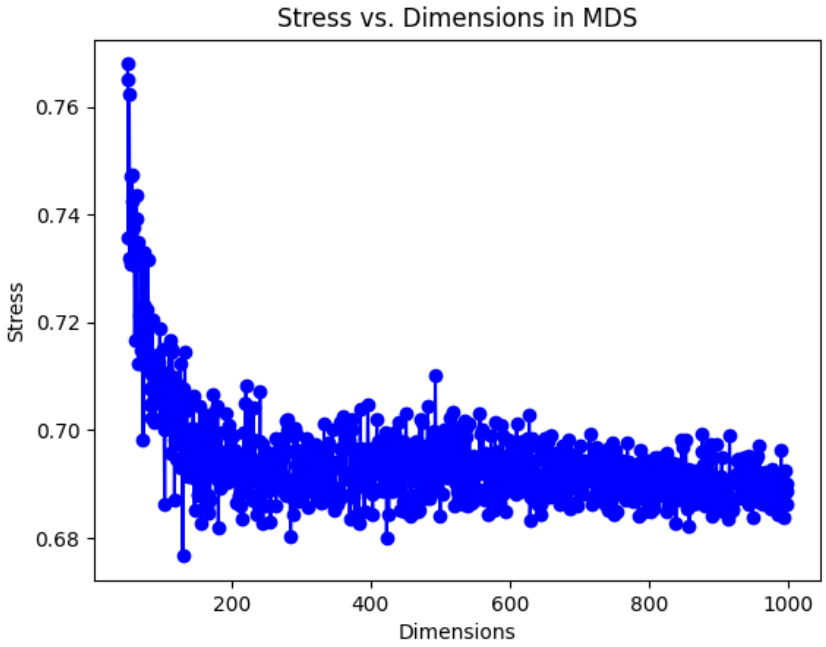

Figure 27 – Stress with each dimension

- Use MDS to reduce the number of dimensions of the keywords embedding.to the amount chosen in the previous step.
- We can then cluster the vectors in the MDS representation to form categories of words. The algorithm used for clustering is K-Means, a popular unsupervised learning algorithm. K-Means aims to partition the dataset into K clusters by minimizing the variance within each cluster. It starts by creating K points as centroids. It then iteratively assigns each data point to the nearest cluster centroid and updates the centroids based on the mean of the data points assigned to each cluster. This process continues until the centroids no longer change significantly, indicating that the clusters have stabilized, or after reaching a set number of iterations.[30]
- Choose an adequate number of dimensions, taking into account that too many clusters will create excessive subcategories, while too few will lead to overly broad categories.

The results of the clustering of the list in chapter 7.1 with 6 clusters are as follows:

- **Cluster 1:** Concerned, Needy, Worried, Anxious, Cautious, Unconcerned, Scared, Hesitant, Seeking reassurance
- **Cluster 2:** Determined, Grateful, Patient, Excited, Appreciative, Eager, Anticipative, Engaged, Hopeful, Expectant, Positive, Anticipatory, Eagerness
- **Cluster 3:** Impatient, Frustrated, Unsatisfied, Inconvenienced, Resistant, Persistent, Dissatisfied, Satisfied, Happy, Annoyed, Frustration, Impatience, Detached, Disappointed, Upset
- **Cluster 4:** Neutral, Polite, Calm, Detailed
- **Cluster 5:** Inquisitive, Curious, Interested, Inquiring, Detail-oriented, Detailed-oriented, Inquiry-driven
- **Cluster 6:** Confused, Unsure, Uncertain, Seeking clarification, Unanswered, Unclear, Confusion, Uncertainty, Doubtful

We observe well-formed categories with adjectives of similar meaning. The first cluster represents a form of concern/anxiety. The second is about positivity and anticipation. The third contains negative feelings of dissatisfaction or annoyance. The fourth cluster represents neutral states. The fifth is about curiosity and interest, and the last one is the category of confusion and lack of understanding. There seems to be one downside with this technique, as a few antonyms were mixed within the categories, with "Unconcerned" in cluster 1 and "satisfied" and "happy" in cluster 3.

The different clusters are represented in Figure 28, where the vectors are visualized in a 2D space using t-SNE [31]. We can see the amount of information lost when reducing the dimensions, with some points being isolated from the rest of their clusters.

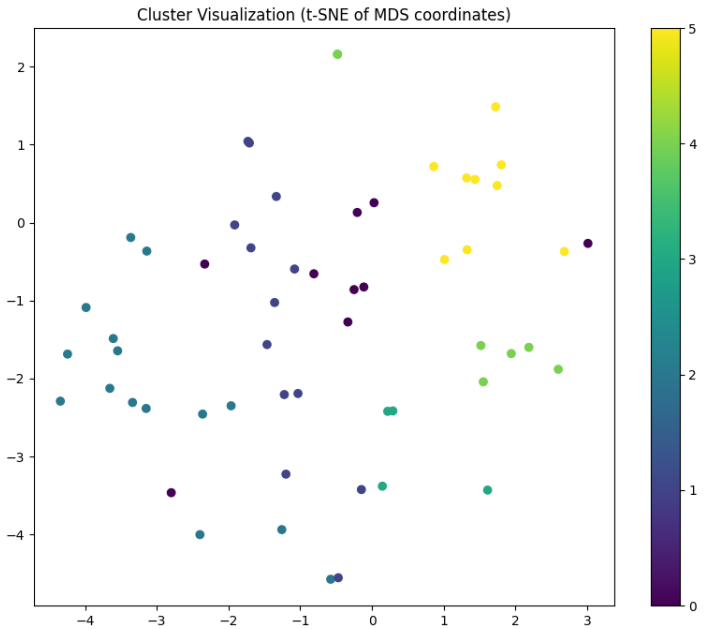

Figure 28 – Visualization of each cluster in 2D using t-SNE representation

An alternative method the clustering is to simply ask a LLM to group the list of adjectives into categories based on similarity in meanings. By asking such a task to GPT-4o, we get the following results :

---

1. Confusion and Uncertainty

Confused Unsure Unclear Uncertain Doubtful Confusion Uncertainty

2. Inquisition and Interest

Inquisitive Curious Interested Seeking clarification Inquiring Inquiry-driven Engaged Seeking reassurance

3. Impatience and Frustration

Impatient Frustrated Annoyed Inconvenienced Unsatisfied Dissatisfied Frustration Impatience Upset

4. Concern and Anxiety

Concerned Worried Anxious Scared Seeking reassurance Unanswered Disappointed Hesitant

5. Positive and Appreciative

Grateful Polite Satisfied Happy Excited Appreciative Hopeful Positive Eager Anticipative Anticipatory

6. Neutral and Determination

Neutral Determined Needy Resistant Persistent Patient Detailed Detail-oriented Detailed-oriented Calm Detached Unconcerned Eagerness Expectant

---

The similarity between this LLM-based grouping and the previous clustering method is 52.05%. While this result might seem low, the main categories are still quite similar, as we can see in table 2. We consistently observe similar core ideas, with perhaps only one notable difference: "unsatisfaction/annoyance" in the clustering method versus "Impatience/Frustration" in the LLM grouping. However, these two titles are still quite similar in concept. This comparison demonstrates that LLMs are capable of performing grouping tasks in a simpler way than the entire clustering process, while still producing meaningful and coherent categories.

Clustering	**Generative LLM**
Confusion and non understanding	Confusion and Uncertainty
Curiosity and Interest	Inquisition and Interest
Unsatisfaction or annoyance	Impatience and Frustration
Concern/anxiety	Concern and Anxiety
Positivity and satisfaction	Positive and Appreciative
Neutral	Neutral and Determination

Table 2 – Categories with each method

7.3. Category annotation guidelines

It is primordial to have a good descriptions of each category in order to have a consistent and precise labelesing. In order to control it, we will compare the labelisation of several people on a sample and see how big the delta is. The following instructions were initially generated with an LLM and subsequently revised to enhance their clarity.

Rating Scale

To better categorize the user messages, each message can be rated on a scale from 1 to 5 for each sentiment category, where 1 indicates the sentiment is not present at all and 5 indicates the sentiment is strongly present.

- **1:** The sentiment is not present at all.
- **2:** The sentiment is slightly present.
- **3:** The sentiment is moderately present.
- **4:** The sentiment is strongly present.
- **5:** The sentiment is very strongly present.

The scoring increase by both the amount of time a sentiment appears and also with how strong the sentiment appears and how certain we are of our interpretation.

Concern

Description: Messages in this category often express worry, concern, or a need for reassurance. The user may appear cautious or hesitant and might seek confirmation or detailed information to alleviate their fears.

Characteristics:

- Expresses worry or concern about booking, safety, or other details.
- Seeks reassurance about specific aspects (e.g., “Is the area safe?”, “Are there any hidden fees?”)
- Shows hesitation or asks for detailed information to mitigate anxiety (e.g., “Can you guarantee a quiet room?”).

Example Messages:

- “I’m worried about the noise levels in the rooms. Are they quiet?”
- “Can I take a bath with my baby?”
- “Is there any dress code requirement ?”

Positive

Description: Messages in this category are characterized by positive feelings. The user might express eagerness and look forward to their visit or express appreciation or gratitude to the service provided.

Characteristics:

- Shows excitement or eagerness about the stay (e.g., “I can’t wait to visit!”)
- Expresses gratitude or appreciation (e.g., “Thank you for the great service!”)

Example Messages:

- “I’m so excited to stay at your hotel next month!”
- “Thank you”

Frustration and Dissatisfaction

Description: Messages in this category express negative emotions such as frustration, impatience, or dissatisfaction. The user may have encountered problems or be unhappy with certain aspects of their experience, including their interactions with the chatbot.

Characteristics:

- Express a complain (e.g., “My room is dirty”)
- Shows impatience or annoyance (e.g., “I’ve been waiting for a response for too long.”)
- Details problems or complaints about the service or facilities (e.g., “The Wi-Fi is too slow.”)
- Indicates dissatisfaction with chatbot replies, often repeating questions or explicitly asking for a human operator (e.g., “I need to speak to a human!”, “You’re not answering my question!”).

Example Messages:

- “The air conditioning in my room isn’t working properly.”
- “hair iron“, “Hair iron Type“, “Hair Iron Manufacturer“
- “I want to cancel“
- “I need to talk to a real person.“

Inquisitiveness

Description: Messages in this category reflect curiosity and a desire to learn more about the hotel and its surroundings. The user is actively seeking detailed information.

Characteristics:

- Asks detailed or specific questions (e.g., “What types of activities are available for kids?”)
- Shows curiosity about various aspects (e.g., “What are some good restaurants nearby?”)
- Displays an interest in exploring and understanding more (e.g., “Can you tell me about the history of the hotel?”).

Example Messages:

- “What are some nearby attractions we can visit?”
- “Can you provide more details about your spa services?”
- “I’m curious about the local culture. Any recommendations on what to see?”

Confusion and Uncertainty

Description: Messages in this category indicate confusion or uncertainty, with the user seeking clarification or answers to unclear aspects of their experience. The user may also show signs of not understanding how to use the chatbot or the replies given by the chatbot.

Characteristics:

- Expresses confusion or uncertainty (e.g., “I’m not sure how to use the booking system.”)
- Seeks clarification or more information (e.g., “Can you explain how the loyalty points work?”)
- Asks questions to resolve doubts (e.g., “Is the breakfast included in my booking?”)
- Shows signs of not understanding the chatbot or its replies (e.g., “I don’t understand your response.”, “How do I ask a question?”).

Example Messages:

- “I’m confused about the different room types. Can you help?”
- “I don’t understand the cancellation policy. Can you clarify?”
- “Is there a difference between the deluxe and standard rooms?”
- “I don’t understand how to use this chatbot.”
- “Your answer isn’t clear. Can you explain it again?”

Neutral

Description: Messages in this category are neutral and polite, often seeking or providing information without strong emotions. The user is calm and composed in their communication and may use very short, neutral messages. Typically messages with low score in other category will have a high neutral score.

Characteristics:

- Straightforward (e.g., “Could you provide me with the check-out time?”)
- Uses very short, neutral messages (e.g., “Check-in”, “Location”).

Example Messages:

- “What time is breakfast served?”
- “How many beds are there in our room?”
- “Check-in”
- “Location”

Example 1

Here is an example of conversation with rating of each sentiment with an explanation


```
"messages": {
    "0": {
        "text": "プリチェックイン",
        "translation": "precheck-in"
    },
    "1": {
        "text": "シャトルバスの予約",
        "translation": "Shuttle Bus Reservations"
    },
    "2": {
        "text": "キャンセル料は何日前からかかりますか？",
        "translation": "How many days in advance is the cancellation fee?"
    },
    "3": {
        "text": "何日前の予約がお得ですか？",
        "translation": "How many days in advance is the best deal?"
    },
    "4": {
        "text": "連泊で別の部屋にはできますか",
        "translation": "Can we stay in different rooms for consecutive nights?"
    },
    "5": {
        "text": "別タイプの部屋に連泊したい",
        "translation": "I want to stay in a different type of room for
            consecutive nights."
```

```
    },
    "6": {
        "text": "連泊中に部屋を変えられますか",
        "translation": "Can I change rooms during. consecutive nights?"
    },
    "7": {
        "text": "2種類の部屋に泊まりたい",
        "translation": "I want to stay in two different rooms."
    },
    "8": {
        "text": "電話で予約は出来ますか",
        "translation": "Can I make a reservation by phone?"
    },
    "9": {
        "text": "キャンセル料",
        "translation": "cancellation charge"
    },
    "10": {
        "text": "1か月前のキャンセル料",
        "translation": "Cancellation fee one month in advance"
    }
}
```


- **concern : 4.** The customer seems to be quite worried about being able to cancel, whether they will be able to change rooms or even how to get to the hotel. While the customer didn't directly talk about something worrying, we can deduce from all the questions that they are not in an ease of mind concerning this trip.
- **positive : 1.** Nothing particular shows any sign of satisfaction, gratitude or other positive sentiment.
- **frustration and dissatisfaction : 5.** The Customer repeats the same query about wanting to change room, which probably means they didn't get their response from the chatbot. The user also talks often about cancellation and about wanting to get in touch with a real human on phone.
- **inquisitiveness : 1.** The questions from the customers are not about details of services from the hotel and surrounding but they are about practical details
- **confusion and uncertainty : 4.** The customer to be confused about the ability to change room and how a cancellation would work. Those uncertainty are on specific aspect and the customer didn't directly specified that they are lost or don't understand some process.
- **neutral : 2.** A few messages of the customer are short and regular queries, but we can clearly some other sentiment in the conversation.

Example 2


```
"user": {
    "0": {
```

```
        "text": "貸し出し",
        "translation": "lending",
        "label": "2"
    },
    "1": {
        "text": "ナイトウェア",
        "translation": "nightwear",
        "label": "2"
    },
    "2": {
        "text": "ナイトウェアサイズ",
        "translation": "Nightwear size",
        "label": "2"
    },
    "3": {
        "text": "朝食",
        "translation": "breakfast",
        "label": "2"
    }
}
```


This second conversation is mostly neutral, with an ounce of dissatisfaction detected the two similar queries about nightwear, a bit of inquisitiveness with nightwear a detail of the hotel service and not a main topic.

- **concern : 1.**
- **positive : 1.**
- **frustration and dissatisfaction : 2.**
- **inquisitiveness : 2.**
- **confusion and uncertainty : 1.**
- **neutral : 5.**

7.4. Categories validation

It is crucial to validate each of these categories to ensure they are distinct enough to make meaningful predictions. To test the distinctiveness of each category, we employed the weighted Cohen's Kappa test. This is an alternative version of Cohen's Kappa. Cohen's Kappa is a measure of agreement between two raters when categorizing items. The higher the agreement between the two raters, the higher the Cohen's Kappa score will be. The scale ranges from -1 (complete disagreement) to 1 (complete agreement), with 0 representing the average agreement of two random classifications. The weighted version adds a layer that takes into account the degree of disagreement, which is particularly useful when the classification is done with an ordinal value (values that can be ordered). In our case, we rate each sentiment from 1 to 5. For example, if the first classification rated concern with a 2 for conversation 5, and the second classification rated it with

a 3, this would be considered a small disagreement. However, if the second classification was a 5, it would be considered a large disagreement. The larger the disagreement, the more it negatively impacts the weighted Cohen's Kappa score. In Figure 29, we see the comparison of labeling for each category using weighted Kappa. We can observe that the two people who did the labeling agreed very well on positivity and frustration/dissatisfaction. Neutral and inquisitiveness show decent and acceptable scores. However, confusion/uncertainty and concern have scores close to 0, indicating that it is too difficult for these sentiments to be reliably detected, as people do not agree on whether or not the sentiment is present in the conversation. This would make any prediction of these categories very challenging. For this reason, the next model will only use the four categories: positivity, frustration/dissatisfaction, neutral, and inquisitiveness.

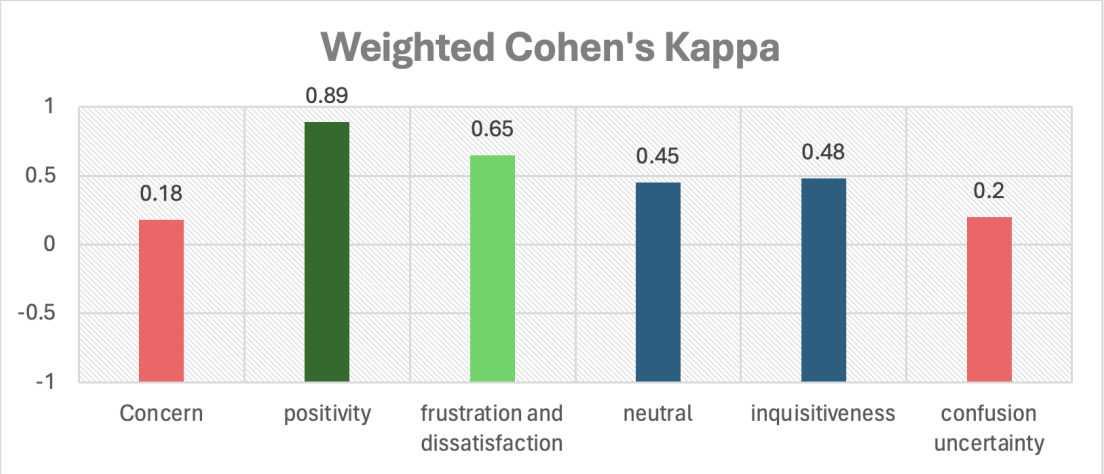

Figure 29 – Weighted Kappa of each label between the labelization of 2 people

7.5. Conversation Labeling with Predefined Categories

Once the categories were determined, it became possible to label each conversation from the 300-conversation dataset. This sample size should be sufficient to train a small model and to get an idea of the prediction performance by validating it on a small sample. The results of labeling the 304 conversations are as follows:

- Neutral: 130
- Frustration/dissatisfaction: 65
- Inquisitiveness: 101
- Positivity: 8

Unfortunately, there are too few "positivity" labels to effectively train our model as it stands. As mentioned in the polarity section, it is rare for users to express happiness when interacting with a chatbot. For this reason, we will exclude this category when analyzing the results of the whole conversation part. A potential solution would be to acquire more conversations from the Tripla dataset and then proceed with balancing, either by adjusting weights or by using data reduction techniques on frequent labels. Another solution would be to use data augmentation, as we did in

the first approach with the polarity (see chapter 6.7). The main issue is that the generation of entire conversation from other existing one is a lot tougher to do than only one message. Only changing each messages slightly would result in a very similar conversation, we would need to be able to generate different flow of conversation from the existing one.

7.6. Model Creation

In this section, two models were explored in order to predict sentiment. The 300 conversations were separated into 70% used for the training and 30% for the testing. The dataset size is limited, because of that, we need to use model that do not require large amount of data and take any evaluation as just an estimation and not a truth. As the testing set will only contain 2 conversation with the "positive" label, we decided to not take that label into account for the f1 score as the data is too small in order to determine it correctly. To avoid this issue, a larger amount of positive conversation would be required.

7.6.1. ChatGPT

The first model possible is by using chatgpt in order to do the prediction.

Prompting

By using the same logic as the first model, we went with the following few shots CO-STAR prompt :

Context
You are a tool to extract sentiment analysis from chatBot conversation between an hotel's customer and a hotel chatbot, able to answer question about the hotel.

Objective
Choose the most adequate sentiment out of the following list : positivity : characterized by positive feelings. The user might express eagerness and look forward to their visit or express appreciation or gratitude to the service provided. frustration and dissatisfaction: express negative emotions such as frustration, impatience, or dissatisfaction. The user may have encountered problems or be unhappy with certain aspects of their experience, including their interactions with the chatbot. inquisitiveness: reflect curiosity and a desire to learn more about the hotel and its surroundings. The user is actively seeking detailed information. neutral: neutral and polite, often seeking or providing infor- mation without strong emotions. The user is calm and composed in their communication and may use very short, neutral messages. Default to this sentiment.

Style / Tone
Only reply with one of the four category Neutral, Inquisitiveness, Frustration and Dissatisfaction, Positive. No extra explanation

Audience
Those category will be used to detect automatically the mood of the user and change the flow of the chatbot when appropriate.

Response

Only the word. Here is an example for each label
Input : "電話で問い合わせ", "お世話になっております。", "子供設定の名称変更方法を知りたいです", "部屋タイプにて和室の場合、布団なのですがその場合はベッドタイプはどのようにすればよろしいでしょうか？","ベッドタイプは必ず入れなくてはなりませんか？","2023年1月1日にご予約をいただいているお客様の予約変更をお願い致します。","2
/11宿泊の下記お客様の宿泊日と料金の変更をお願いします。","お疲れ様です。","4
/20ご予約されている" Output : Positivity

Input : "調味料を教えてください""ドームテントサイトとは","BBQグリル", "ガスコンロ","レンタル","フライパン", "ホットプレート","チェックアウト"
Output : neutral

Input : "バイキングのメニュー教えて", "おもちゃ王国の入場料は？", "ベビーグッズ", "入浴時間は？", "チェックアウト日に温泉利用できる？", "チェックイン前に温泉利用できる？", "宿泊者の温泉料金は?", "大浴場の金額は？", "宿泊のアメニティは？", "子供用食器ある？"
Output : inquisitiveness

Input : "あごだで予約した場合のチェックイン場所", "台帳とは？", "ネットで予約した場合のチェックイン場所", "フロントへの声のかけ方", "鍵は", "鍵はそのまま持っててもいいのか", "ルームカードキーは返すのか", "チェックアウトの時鍵は返す？", "鍵の返す場所", "鍵を返す場所", "チェックアウトの仕方"
Output : frustration and dissatisfaction

Result

On the 90 conversation of the testing set we got a result of 70% accuracy and a combined f1 score for the first 3 labels of 65% (see Figure 30). In about 80% of the cases, chatgpt doesn't detect "frustration and dissatisfaction" but is able to distinguish between Inquisitiveness and neutral with each more than 80% accuracy. It might be explained by our use case containing has more subtle frustration/dissatisfaction examples than what the model is used to assign this sentiment with in its training data. If this hypothesis is correct, it would one of the weakness of LLM. They are trained on a very large amount of data that will not necessarily be relevant to our use case.

Chatgpt	frustration and dissatisfaction	inquisitiveness	neutral	positivity	Accuracy	Recall	F1
frustration and dissatisfaction	4	6	7	2	21%	80%	33%
inquisitiveness	1	27	2	0	90%	66%	76%
neutral	0	7	32	0	82%	76%	79%
positivity	0	1	1	0	0%	0%	0%
					70%		65%

Figure 30 – Confusion Matrix, prediction classification chatgpt

7.6.2. Neural Network Classification

The second model explored is a neural network with one hidden layer of 100 nodes. It is trained using MLPClassifier from scikit-learn [32]. This model optimizes the log-loss function using the backpropagation technique to tune each node. The input to the model is the user messages in an

embedded form, meaning that the conversation is not in text but instead represented as a vector containing its features. The use of an embedding instead of raw text has three major advantages:

- The messages of a conversation form a large text when combined, making it difficult for a model to isolate the right parts of the text to understand the nuances necessary for classification. An embedded input automatically has all the features in a numerical form, which is much easier for a model to process.
- The number of examples we have is low, with only 214 in the training set. This is not enough to train a model on raw text, as there is a high risk of overfitting to certain words. The vectorized input will be less prone to overfitting as it only shows the features of the text, not its content.
- We do not need to translate the messages, after experimentation we saw very similar result if we used the conversation translated or in their default language to create the embeddings.

The embedding model used is again text-embedding-3-large, which can create a 3072-dimensional vector from a string [25]. This format is preferable to the previous methodology used in the polarity prediction, as here we only have one example per conversation and the conversations are much longer strings than single messages. Combining these factors would make it difficult to extract enough information directly from the text. The model was trained on 213 conversations from the training set and evaluated on a validation set of 90 conversations. The model achieved an overall accuracy of 78% and a combined F1 score of 77% for the first 3 categories (see Figure 31). We observe lower accuracy and recall for frustration and dissatisfaction, which might be due to their lower representation in the dataset.

Neural Network	frustration and dissatisfaction	inquisitiveness	neutral	positivity	Accuracy	Recall	F1
frustration and dissatisfaction	10	4	5	0	53%	67%	59%
inquisitiveness	1	27	2	0	90%	77%	83%
neutral	2	4	33	0	85%	83%	84%
positivity	2	0	0	0	0%	0%	0%
					78%		77%

Figure 31 – Confusion Matrix, prediction classification neural network

To improve the model, we decided to add an additional feature to the embedding that is not present in the user messages: the rate of repetition in the bot replies. In theory, the more the bot repeats the same response, the less likely the client will be satisfied, as they keep trying to get the answer they desire without success. In the training set, the repetition rates are between 0.22 and 0.23 for neutral and inquisitiveness labels, and 0.35 for the frustration and dissatisfaction label.

To account for this, the repetition rate attribute is added to the model with a higher weight than the other 3,072 features of our embedded user messages to give it more importance. This improvement did enhance the results for frustration and dissatisfaction, as expected (see Figure 32).

Neural Network + Repetition	frustration and dissatisfaction	inquisitiveness	neutral	positivity	Accuracy	Recall	F1
frustration and dissatisfaction	15	1	3	0	79%	75%	77%
inquisitiveness	0	28	2	0	93%	88%	90%
neutral	3	3	33	0	85%	87%	86%
positivity	2	0	0	0	0%	0%	0%
					84%		85%

Figure 32 – Confusion Matrix, prediction classification neural network with extra repetition features

The final result give us 84% accuracy and 85% F1 score of the 3 first labels. This is a lot better than the result of Chat, the usage of a embedded conversation in a neural network seems to be adequate.

7.7. Validation

Towards the end of the project, access to new conversations was granted. We can use these additional conversation data to validate our models. It is important to note that there is one significant difference between the original data and the new validation dataset: some of these conversations have fewer than 11 messages, with an average of 8.03 messages per conversation (compared to 16.3 in the original dataset, as mentioned in chapter 6.1). This difference is because originally, the data provided to us was selected on having a length higher than 10, probably in order to have conversation with enough content to see sentiment from. The newer data provided wasn't made with this check so in order to ensure a fair comparison, we will only use messages from conversations with at least 11 messages, making the check on our side. Out of the 730 conversations in the new dataset, 157 met this criterion (with 3 more removed as the messages send by the user in those conversations were not coherent).
The validation results show an accuracy of 84% and a combined F1 score of 82%, as illustrated in Figure 33. We observe slightly worse results, which might be due to the variance resulting from the low number of conversations and also possibly due to some differences in the selected conversations for validation. As we see differences in message sizes, there might be other slight variations explaining the lower results. These results validate our findings from the testing set. The performance is good for our context, where many conversations aren't heavily typed in emotion, making the difference while labelling between one label or the other slim.

Validation	frustration and dissatisfaction	inquisitiveness	neutral	Accuracy	Recall	F1
frustration and dissatisfaction	19	4	3	73%	68%	70%
inquisitiveness	3	36	7	78%	86%	82%
neutral	6	2	74	90%	88%	89%
				84%		82%

Figure 33 – Confusion matrix on validation set

8. Presentation and demonstration

A very crucial point of the project is being able to transfer the knowledge from the project to the other member of the team at Tripla. The project is an exploration of possibilities and doesn't apply real use cases yet, therefore, we need to pass the knowledge to the rest of the team so it can be used for futur projects. Other than this report, two step were taken in order to pass the knowledge.
The first one is the creation of two notebooks, explaining every steps of the project. Each notebook also contains a simple UI making it possible to use the models created. The UI is done via Gradio [33]. Gradio is a simple python library allowing for quick demonstration of machine learning models. In the Figure 34, we can see the UI for the message polarity model. This UI can be accessible directly via the notebook or on a browser.

Figure 34 – Gradio UI of the polarity model

The second is a presentation within Tripla, presenting the project, the context and different step taken during it, in a similar fashion as the oral defence at the end of the bachelor project. This allowed to share the knowledge among other member of the data team, in order for them to understand the context of the project, the goals, the steps and the futur use cases.

9. Future Work and Discussion

This project has allowed us to create a robust process for developing sentiment analysis models for both individual messages and entire conversations. From this baseline, numerous possibilities are open to continue this project. This chapter will cover some ideas that emerged during the project but were out of scope or beyond the available time budget.

9.1. Model Improvement and Methodology Validation

The focus of the project was not to push the performance of the models to their limits, but rather to create a step-by-step process for training and using such models. While we did implement multiple steps to improve model quality, there are still other techniques possible to enhance model performance. Here is a list of ideas:

- The first and simplest approach is to gather more data for training. The quality of both the fine-tuned model and the neural network model depends on the amount and quality of the dataset. A larger number of samples, especially for the rarer labels, will definitely improve the quality of the models.
- Use quantized LLMs to reduce overfitting by reducing the precision of nodes.
- Explore more LLMs available on the market. For example, the new model from OpenAI, GPT-4o mini, released in July 2024 [34], or some of the Gemini models from Google. As models continue to improve and become more cost-effective, they will become better alternatives to locally fine-tuned models. We could also try out the fine-tuning option on the OpenAI API, which could yield great results but at a relatively high cost [35].
- Implement data augmentation techniques on entire sequences of conversations. We decided not to do this, as creating different conversations from existing ones that are realistic but sufficiently different is much more challenging than augmenting single messages. Taking the time to develop a data augmentation technique for entire conversations would improve the quality of the model, but it might also be easier to just extract and label more conversations from Tripla's database.

Another valuable idea would be to validate the methodology used with our dataset against other, larger existing datasets. For example, we could assess how reliable the results of the evaluation are on only 100 conversations. We could train a model on the other dataset and run the evaluation on multiple sets of 100 conversations, then compare how similar the results of each evaluation chunk are. The greater the difference, the less reliable our results would be. We could also check the effectiveness of our data augmentation techniques by comparing a model trained with, for example, 10,000 entries to one trained with 2,000 entries augmented five times. If the results are close, then the data augmentation technique provides valuable new examples for the model to train on. The main reason this would be useful is due to our low-entry dataset, which might lead to overfitting, creating a model that doesn't work well with new data.

9.2. Use case and integration in Tripla software

Currently, the models are not used in a software, but are proof-of-concept. The natural flow following this project would be to integrate them in some way within Tripla Software. There are a few use cases possible like the following :

- Change the conversation flow depending on the current sentiment of the user. For example : If the user is inquisitive, we can show them some extra plans, activities, travelling options as the inquisitiveness show they are still fully fixed on how their trip will go and are more likely to be open to suggestions, possibly increasing sales. If the user is frustrated / dissatisfied or show a lot of "negative" messages, we could connect them to a real operator in order to fix their issue in a efficacy and quick matter. In the current state of the software, the client can connect to a real operator by pressing a "thumb down" button. A automatic detection of the negative experience will lower the time spend by the user having a bad experience with the chatbot.
- Creating weekly report for hotel manager : We could use positive and negative sentiment of each user, combine them with the intent of each message to get a global view of strong and weak points of the hotel. For example, if several customer had negative messages on the subject of the furniture of a specific chamber, we could report it to the hotel manager as a point to improve. In contrary, if many customer has positive sentiment messages on the spa, this will show that the spa is an element to put in the spotlight for advertising the hotel.
- Using the sentiment detected after each types of possible replies from an hotel in order to avoid those inducing a negative sentiment or detecting the lack of a possible reply in the list of choices for the hotel. Those two upgrades will improve the customer experience by modifying the list of possible replies and their choice in order to meet customer expectations.

9.3. Exploring cheaper conversation embedding (sequence)

Currently, the embedding of the conversation for the model of the sequence is done via openai API, which will represent some relatively big cost if we want to predict the sentiment of every conversation from Tripla users. It would be nice to explore different way of vectorising the conversation in order to reduce the cost / computation time.

9.4. Using Japanese Fine-tuned Model (Polarity)

The local model for polarity still requires translation to English for most messages, increasing the cost and computation time of prediction, especially when using an API such as DeepL. The ideal scenario would be to have a Japanese model, but as mentioned in chapter 6.3, we decided to focus on an English model as it was the safest solution. Using a Japanese model would allow us to skip the translation time for 85% of conversations instead of only 7.5% with English.

9.5. Streamlined Model Creation

An interesting continuation of the project would be to create a pipeline capable of automatically adapting both the classification categories and the model with new data. This would allow the model to modify the different classification categories depending on the input data. The main challenge in this approach is to replace all manual parts of the process with LLM prompts, mainly the labeling of the training dataset and the choice of the number of clusters. While this sounds feasible, it would require significant effort, and it would be important to consider whether the effort is worthwhile.

10. Conclusion

To conclude this project, let's review the project and the experience it gave me.

10.1. Feedback on objectives

This chapter will be recapitulating the realisation of each objectives of the project (objectives details are in chapter 3)

10.1.1. Primary Objectives

The first objective : "State-of-the-art sentiment analysis on chatbot traces" is a success. A finetuned bert model was created in order to detect the polarity of sentiment from a message in a chatbot trace. The model was compared to other solution such as using ICL with chatGPT. Those solution were based on research about the current best practices for sentiment analysis. The model has an accuracy of 90% and F1 score of 80% which is a good result for small messages with low sentiment context. The model could probably be improved by increasing the amount of data used in the training.
The second objective : "Prototype for analyzing the entire sequence of exchanges in the chatbot" is also a success. A process was designed in order to define the types of sentiment tag we can assign to each conversation. Those tags were then validated by controlling how detectable they are in the data. Finally a neural network model was created in order to perform the sentiment classification for a conversation by using the power of embedding LLMs. The result is good with 84% accuracy and 82% F1 score. The model does have the downside of not detecting the positive conversation as we were lacking examples in our small dataset. We ended up not using the previous sentiment tags as it wasn't necessary, but we did add repetition pattern as a feature.

10.1.2. Secondary Objectives

We chose to focus on model elaboration. For this reason, The two secondary objectives which were related to model usage were not implemented. Discussions of the usage were still done (see section 9.2) in order in what matter could the models be used and what will be the following of this project.

10.2. Experience in a company in Japan

The particular settings of this project allowed me to have a wonderful experience both in the professional and the human side. For first, having the opportunity to work in a company, and a foreign one allowed me to greatly advance in my professional knowledge and prepare me for a future job in any settings, possibly even abroad. I learned about communication between the a development team, how to keep it efficient while not taking too much time per day. I also got the chance to participate in a hackaton, where all the employees got together to come up with different new ideas and prototypes in two days, which make everyone collaborate with their creativity.

On the more human side, I got to meet a lot of people from every part of the world both at the company and within the sharehouse I was living at. Living in a big city as Tokyo is a very opposite experience to the life in Switzerland. This opportunity really helped me to grow, I enjoyed it greatly and I'm very grateful for it to Tripla and the HEIA-FR for giving it to me.

10.3. Thoughts on project

This project was more data oriented, in contrast with my option during my formation which were more focused on software development. This did not cause any issues thanks to my previous semestrial projects about LLMs and the great help I got from the responsable of the project within Tripla M. Jean Cadic, from my teach Ms. Sandy Ingram and my two experts M. Robert Van Kommer and M. Juan Carlos Farah. I am very grateful to all of these people that accompanied and helped me during this project. It was challenging to plan this project in which I had a lot of freedom in the direction to lead it to. It was important to take some steps back in order to see the project in it's entirety. I fortunately didn't encounter any major setback. I am satisfied with the results and the knowledge learned during the project.

11. Declaration on honour

I, the undersigned, Florian Chassot, declare on my honour that the work submitted is my own work. I certify that I have not resorted to plagiarism or any other form of fraud. All sources of information used and author citations have been clearly stated.

Bibliography

[1] *google bert/bert-base-uncase - Hugging Face*. URL: https://huggingface.co/google-bert/bert-base-uncased.

[2] Javad Hassannataj Joloudari et al. "BERT-deep CNN: state of the art for sentiment analysis of COVID-19 tweets". In: *Social Network Analysis and Mining* 13.1 (July 2023). ISSN: 1869-5469. DOI: 10.1007/s13278-023-01102-y. URL: http://dx.doi.org/10.1007/s13278-023-01102-y.

[3] Zengzhi Wang et al. "Is ChatGPT a Good Sentiment Analyzer? A Preliminary Study". In: *arXiv preprint* (2023).

[4] D. Y. Khasawneh et al. "Cross-Lingual Sentiment Analysis: Comparative Study of Opinion Expression Across Different Languages". In: *Migration Letters* 21.S1 (2023), pp. 548–560. DOI: 10.59670/ml.v21iS1.6179.

[5] Md Saef Ullah Miah et al. "A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and LLM". In: *Scientific Reports* 14.1 (Apr. 2024), p. 9603. ISSN: 2045-2322. DOI: 10.1038/s41598-024-60210-7. URL: https://doi.org/10.1038/s41598-024-60210-7.

[6] Y. Takakusagi et al. "Validation of the Reliability of Machine Translation for a Medical Article From Japanese to English Using DeepL Translator". In: *Cureus* 13.9 (Sept. 2021), e17778. DOI: 10.7759/cureus.17778.

[7] K. Dashtipour, S. Poria, A. Hussain, et al. "Multilingual Sentiment Analysis: State of the Art and Independent Comparison of Techniques". In: *Cognitive Computation* 8.4 (2016), pp. 757–771. DOI: 10.1007/s12559-016-9415-7. URL: https://doi.org/10.1007/s12559-016-9415-7.

[8] *Sentiment Analysis | Paper with code*. URL: https://paperswithcode.com/task/sentiment-analysis.

[9] Łukasz Augustyniak et al. *Massively Multilingual Corpus of Sentiment Datasets and Multi-faceted Sentiment Classification Benchmark*. 2023. arXiv: 2306.07902 [cs.CL].

[10] *Bleu - a Hugging Face Space by evaluation metrics*. URL: https://huggingface.co/spaces/evaluate-metric/bleu.

[11] *Tatoeba: Collection of sentences and translations*. URL: https://tatoeba.org/en/.

[12] *Helsinki-NLP/opus-mt-ja-en - Hugging Face*. URL: https://huggingface.co/Helsinki-NLP/opus-mt-ja-en.

[13] *Pricing | OpenAI*. URL: https://openai.com/api/pricing/.

[14] Qingxiu Dong et al. *A Survey on In-context Learning*. 2024. arXiv: 2301.00234 [cs.CL]. URL: https://arxiv.org/abs/2301.00234.

[15] Rishabh Agarwal et al. *Many-Shot In-Context Learning*. 2024. arXiv: 2404.11018 [cs.LG]. URL: https://arxiv.org/abs/2404.11018.

[16] Kenneth Leung. *Visualizing Data using t-SNE*. 2023. URL: https://www.kdnuggets.com/2023/01/micro-macro-weighted-averages-f1-score-clearly-explained.html.

[17] Amir Ahmed Khan. "Bid Adieu To Plain Generic Instructions — CO-STAR Empowers You To Craft Prompts With Surgical Precision, Ensuring LLMs Not Only Understand, But Exceed Your Expectations." In: *Medium* (2024). URL: https://medium.com/@amirahmedkhan15/bid-adieu-to-plain-generic-instructions-co-star-empowers-you-to-craft-prompts-with-surgical-cd3a465865e0.

[18] *Models - Anthropic*. URL: https://docs.anthropic.com/en/docs/about-claude/models.

[19] *Introducing the next generation of Claude Anthropic*. URL: https://www.anthropic.com/news/claude-3-family.

[20] Dave Bergmann. "What is fine-tuning?" In: *IBM* (2024). URL: https://www.ibm.com/topics/fine-tuning.

[21] *(Huggingface) Transformers*. URL: https://huggingface.co/docs/transformers/en/index.

[22] *finiteautomata/bertweet-base-sentiment-analysis - Hugging Face*. URL: https://huggingface.co/finiteautomata/bertweet-base-sentiment-analysis.

[23] Shahul ES. "Data Augmentation in NLP: Best Practices From a Kaggle Master". In: *Blog - neptune.ai* (2023). URL: https://neptune.ai/blog/data-augmentation-nlp.

[24] Juan Manuel Pérez, Juan Carlos Giudici, and Franco Luque. *pysentimiento: A Python Toolkit for Sentiment Analysis and SocialNLP tasks*. 2021. arXiv: 2106.09462 [cs.CL].

[25] *Embeddings | OpenAI API*. URL: https://platform.openai.com/docs/guides/embeddings.

[26] *GloVe : Global Vectors for Word Representation*. URL: https://nlp.stanford.edu/projects/glove/.

[27] Matthew Freestone and Shubhra Kanti Karmaker Santu. *Word Embeddings Revisited: Do LLMs Offer Something New?* 2024. arXiv: 2402.11094 [cs.CL]. URL: https://arxiv.org/abs/2402.11094.

[28] Fanghua (Joshua) Yu. *Text Embedding — What, Why and How? | by Fanghua (Joshua) Yu | Medium*. URL: https://medium.com/@yu-joshua/text-embedding-what-why-and-how-13227e983ba7.

[29] Leland Wilkinson. *Multidimensional Scaling*. URL: http://cda.psych.uiuc.edu/mds_509_2013/readings/systat_scaling_manual.pdf.

[30] *K means Clustering – Introduction - GeeksforGeeks*. 2024. URL: https://www.geeksforgeeks.org/k-means-clustering-introduction/.

[31] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605.

[32] *MLPClassifier - scikit-learn 1.5.1 documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

[33] *Gradio*. URL: https://www.gradio.app/.

[34] *GPT-4o mini: advancing cost efficient intelligence | OpenAI*. URL: https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/.

[35] *Fine-tuning | OpenAI API*. URL: https://platform.openai.com/docs/guides/fine-tuning.

ChatGPT was used as a tool to upgrade the formulation of sentences after an initial manual redaction and as spelling checking tool.