Aalto University
MS-E2177 - Seminar on Case Studies in Operations Research

Using generative AI in the risk assessment process

Final Delivery

Milana Begantsova, Project Manager
Rami Echriti
Kalle Johansson
Jaakko Vauhkonen

May 19, 2024

inclus

Contents

1 Introduction

1.1 Background

The client of this project, Inclus, is a Finnish scaleup that provides SaaS solutions for collaborative risk analysis. They provide risk assessment and management services at both company and project levels, as well as scenario planning and continuity management solutions. Inclus is interested in using generative AI in their risk assessment process. More specifically, Inclus is interested in how large language models (LLM), such as the GPT models by OpenAI, could be utilized as part of their services.

To understand the potential use cases and the related problems of LLMs, it is important to understand Inclus's product and the data it gathers. Inclus provides a platform for virtual workshops in which a group's views on risks are elicited and stored. In the workshops, the participants identify a set of risks and then evaluate the risks. The participants rate the risks along a predefined set of dimensions like the likelihood and impact of each risk numerically and then provide comments related to the numerical evaluations.

Since a project has multiple types of risk, such as financial, business development, market, IT, and personnel risks, the number of risk evaluations and related comments can be in the hundreds or potentially thousands. This means that manually reviewing and evaluating all the risk data to gather an overall summary is very time-consuming and challenging. This is where Inclus has opted to use a generative AI model.

Currently, Inclus's platform provides AI-generated summaries of the project's or company's risks to reduce workload from reading through the results and comments of the risk evaluations. However, the presence of quantitative data poses difficulties for LLMs as they may struggle to interpret the data and its relation to the qualitative data correctly. It is also known that the accuracy of LLM outputs is often lacking and the structure of the outputs do not always reliably match given specifications.

1.2 Objectives

Inclus has set a high-level goal of using AI to improve the risk management process by eventually being able to predict whether a project will succeed or not based on the associated risk data. This end goal is beyond the scope of our project and we will focus on a subset of this target. To predict the success of a given project, the LLM must be able to properly interpret both the quantitative and qualitative data related to the project's risks. So, this project's objective is to improve the accuracy as well as the prompting to enable extended use of generative AI in the future.

More specifically, this project aims to provide recommendations on how quantitative and qualitative risk data should be combined in prompts for the LLM to ensure that the model interprets both aspects of the data better. This way we can help the LLM interpret the project risks better. A number of approaches have been determined together with Inclus. These try to improve the quality of the outputs by transforming the data from qualitative to quantitative form and vice versa. These approaches have particular use cases as well. These are summary formulation, outlier detection, and feature extraction from given data. These are discussed more thoroughly in Sections 4, 5, and 6.

As a secondary objective, this project aims to provide considerations to avoid AI hallucination to increase the reliability and accuracy of outputs. Hallucination refers to the tendency of LLMs to produce baseless claims and outputs. This is explained and discussed more in Section 2.1. Lastly, we will summarise what kind of problems generative AI is most reliable and useful for in terms of Inclus' risk analysis.

To reiterate, we aim to figure out how Inclus can turn project contents and related risk data such as risk category, comments, and assessment scores of each dimension, into a single prompt for the LLM so that they can receive accurate answers regarding the project. In effect, we form a text representation of the entire project contents in such a way that the AI model can correctly interpret it. In this project, we only use OpenAI's GPT-4 model, as it has been the model of choice for Inclus thus far.

Inclus does not expect a perfect prompt engineering solution to give fully accurate outputs as LLM accuracy is still a large-scale problem, which is yet to be solved. We will test the prompt engineering approaches using both strict and soft benchmarking questions and assess how well the LLM can answer them. An example of a strict benchmarking question could be: "What are the three risks with the highest impact?". Here we would expect an output with the top risks in terms of average impact. This can easily be manually verified and quickly provides an idea of how capable the model is for Inclus' use case.

Conversely, we will also use soft benchmark questions which are expected to be more challenging for the LLM. This could be: "Provide a summary of the risk data.". Here we would expect multiple paragraphs of high-quality text that accurately describes the features of the data, highlighting the most significant risks and high-quality comments. Such an answer is harder to evaluate but this is an important use case for Inclus and therefore must be explored. Failures can be found when the LLM asserts facts that cannot be verified from the given data or if the output disregards the key risks. The aim is to get outputs that accurately answer the benchmark questions and if possible, the outputs would require only minor adjustments for Inclus to be able to use them in their reports and analysis.

2 Literature review

The use of language learning models has significantly increased with the release of ChatGPT by OpenAI in late 2023. Since then, LLMs have been introduced in an extremely wide set of applications and industries. ChatGPT has been explicitly used in the risk assessment and management context and its accuracy was evaluated in [1]. This study focused on the risk management aspect and decision-making and evaluated the performance in risk identification, analysis, response, and monitoring. The study found that the used LLM (GPT-4) had moderate performance in managing risk and that the LLM provided more accurate outputs in terms of risk response and monitoring in comparison to risk identification and analysis when tested on a range of different project contexts. It is therefore most reasonable to further study the possible applications and improvements in the use of LLMs in this field.

2.1 Hallucination

Hallucination in the context of AI and language learning models, refers to instances where the used AI model generates outputs that are unfaithful to given data. These outputs can contain incorrect or entirely fabricated information based on the model's training data and algorithms [2]. Such errors in outputs are dangerous as they are hard to detect and can lead to analysis that is not grounded in fact. This is why studying both hallucination reduction methods and hallucination detection methods is important.

2.1.1 Hallucination detection

Since LLMs are meant to produce coherent, high-quality text, it is often very difficult to spot inconsistencies and factual errors. Different language learning models' accuracy and hallucination tendencies have been benchmarked in [3]. The results of Li et al. [3] show that the LLMs tend to generate unverifiable information related to the given topics. This was done using a benchmark developed by the authors called *HaluEval*. They examined both the LLMs' tendency to generate hallucinations, as well as their ability to detect the hallucinations that they did generate. In specific, for all outputs from *ChatGPT-3.5-turbo*, around 20 percent contained hallucinations. They tested the LLM capabilities on detecting hallucinations in text summaries that they generated. This is of interest as Inclus currently uses an LLM in this way. They found that GPT models only had around 50 percent accuracy in determining if the summaries they generated contained hallucinations. Models *GPT-3* and *GPT-3.5-turbo* had accuracies of 51 and 58 percent respectively. Such accuracies are only barely above chance.

For other tests, such as question answering and knowledge-based dialogue, the *GPT-3* and *GPT-3.5-turbo* were more accurate. It was also found, that providing external data and intermediate reasoning steps improves LLMs' ability to detect hallucinations. This makes the accuracy of question answering and dialogue go even higher.

HaluEvals benchmark setup is quite different from the scope and approach of this project, especially considering the varied topics and contexts used in the evaluations. Fortunately, the ability to test LLMs accuracies using custom-selected datasets is also provided in the GitHub repository corresponding to the paper. This could help assess the amount of hallucinations with the model used by Inclus. The implementation of *HaluEval* was not reasonable with the resources of this project and was therefore not done.

Key takeaways from the above are that the LLMs are prone to hallucinate far more than acceptable for unsupervised use, and providing external data and reasoning steps increases the accuracy and quality of outputs. It is also important to note that by themselves, LLMs are very poor at detecting their self-generated hallucinations, including in summaries. The benchmark was done on multiple LLMs, but unfortunately, this research did not include the GPT-4 model currently used by Inclus.

We have identified other detection methods in addition to *HaluEval*, which can be beneficial for Inclus to increase the accuracy of their LLM outputs. One of these approaches is *SelfCheckGPT*. *SelfCheckGPT* can be used to both detect non-factual statements and rank the statements factually. This is an interesting hallucination detection method, as it requires no external data sources or other resources. It is a black-box approach, meaning it does not require internal knowledge or output distributions of the evaluated model. *SelfCheckGPT* is a sampling-based method, where the same prompt is used multiple times and the consistency of the outputs is assessed using different metrics. The logic behind the method is that if the LLM correctly interprets the prompt, the outputs should have very little variance and contain no factual inconsistencies [4]. SelfCheckGPT provides multiple different evaluation metrics and we examine three of them below.

BERTScore is a metric based on contextual embeddings generated by BERT models and it offers an understanding of the similarity between sentences. In *SelfCheckGPT* this metric is used to assess the quality of generated responses. The method involves calculating the average BERTScore between each sentence and its most similar counterpart from a set of drawn samples. By measuring the similarity between sentences across different samples, *SelfCheckGPT* with BERTScore provides valuable insights into the coherence and consistency of the generated text. Higher BERTScore indicates greater similarity and potentially higher quality responses.

Another approach to evaluating *SelfCheckGPT*'s output involves employing a multiple-choice question-answering generation (MQAG). MQAG generates questions over the main response and assesses the consistency of answers provided by an independent answering system across different samples. The process involves two stages: 1. question generation and 2. question answering. Questions are generated based on the main response and multiple-choice options are provided. Thus, an answering system attempts to answer these questions for each sample. Consistency in the answers across samples indicates reliability and coherence in the generated text. Inconsistencies may suggest areas where the model needs improvement or further training.

The n-gram approach offers a different perspective on evaluating the output of *SelfCheckGPT*. Instead of focusing on semantic similarity or consistency, this method analyzes the probability distribution of the generated text. By training a simple n-gram language model using the generated samples and the main response, researchers can approximate the likelihood of the text being generated by the language model. The average or maximum of the log probabilities of the tokens in the main response is then computed using this n-gram model. Higher probabilities indicate a greater likelihood of the text being coherent and consistent with the language model's training data. This method provides insights into the grammatical correctness and fluency of the generated text.

One clear limitation of *SelfCheckGPT* approach is that it is computationally demanding. This is relevant in the context of Inclus' use case where we have to sample prompts containing a large amount of risk data multiple times and then compare the consistency of the outputs. It is also important to note that the study and results of *SelfCheckGPT* were based on OpenAI's GPT-3 model. It is still reasonable to assume that these results are comparable with GPT-4, especially since the logic behind the method is independent of the used model.

In terms of detection, OpenAI provides tools directly in their API to better assess the quality of the LLM outputs. OpenAI provides a parameter called *logprobs* in the API responses [5]. This can be used to find the log probabilities of output tokens to indicate the likelihood of each token occurring in the output sequence given the context. These can be used to compute confidence scoring and thus detect and reduce hallucinations. This confidence can for example be used to evaluate how confident the LLM is in categorizing or classifying a risk comment. This process is explained further in [5]. This was applied in the feature extraction approach, explained in Section 3, since there we ask the LLM to gauge risk comments objectivity and other aspects.

OpenAI's *logprobs* parameter is a viable option as this was easily implemented since it is already in the API, and could therefore provide information about the models' hallucination quickly. This method was tested during our prompt engineering experiments and the implementation and results are discussed more in Section 7.

One reason to consider using *SelfCheckGPT* together with, or instead of, the output probabilities is that according to Manakul et al., their sampling-based approach was able to outperform the output probabilities in most setups [4]. This would need to be tested with Inclus' specific use to see whether the improvements would be worth the additional computational cost.

When hallucinated sentences or passages are detected, these can be manually corrected, meaning implementing detection methods is useful even when no direct hallucination reduction methods are used. It is necessary to highlight that the topics and contexts used to test the LLMs are not risk management specific and therefore these methods may not perform as well for Inclus. Neither *HaluEval* nor *SelfCheckGPT* examined the performance of GPT-4.

2.1.2 Hallucination reduction

Reducing hallucination is a much more complex issue than merely detecting hallucinations. It has even been suggested, that hallucinations are an innate aspect of LLMs and thus fully eliminating hallucinations is not feasible [2].

The LLM hallucination reduction methods explored during our literature review are threefold. These are prompt engineering approaches, data integration, and model fine-tuning. A plethora of specific methods and approaches within and outside these categories have been surveyed in detail in [6]. Prompt engineering involves designing the inputs that are provided to the LLM to get higher-quality outputs [7]. Prompt engineering was chosen as the main method to improve our LLM outputs as it is simple to implement when interacting with the LLM through the API. Prompt engineering is discussed more in Section 2.2.

Data integration and fine-tuning are more complex and harder-to-implement approaches. Data integration refers to ways to extend the models' pre-trained knowledge with external data. This could be connecting Inclus' project risk data with the LLM or making direct database calls. The idea would be that the LLM can access all data that is relevant to the query. If the model is able to access regularly updated data, then the relevance and accuracy of outputs would be even greater.

Popular ways to integrate data with an LLM are retrieval-based methods, the most prominent of which is Retrieval Augmented Generation (RAG). RAG is a method, where the generative LLM is connected to a retrieval system that operates on a large-scale database to augment prompts. This setup has been shown to provide higher quality outputs than typical LLMs [8]. This retrieval system counteracts the LLM's inaccuracies by helping it stay grounded in fact by providing available data. Specifically, RAG determines and retrieves the most relevant information regarding a prompt from the database. In Inclus' case, this could be very beneficial. Creating a database with risk data from every project and client enables Inclus to formulate a question to the LLM about the risks of a specific project and RAG is then able to compare the data with other projects. This could potentially lead to more accurate and relevant outputs or even predictive insights.

A practical problem with the RAG approach is that it is far more complex to implement than prompt engineering as one needs to create the appropriate architecture for the LLM model to work in conjunction with the retrieval system. Similarly, providing this external data to the LLM means that the outputs are now very dependent on the quality of the database. Inclus would then have to make sure the data is properly curated and that low-quality assessments, like ones lacking comments, are not affecting the performance of the LLM [9].

Fine-tuning is another approach to providing more context-specific knowledge to the LLM to help improve the quality of outputs. Here the LLM is trained further using proprietary domain knowledge to incorporate relevant data into the model. The logic is that for specific tasks, it makes sense to train the LLM with a dataset specific to that task. Models like FinGPT and PMC-Llama have been fine-tuned and shown to improve general-purpose LLMs' performance in the financial and medical domains respectively [9]. RAG and fine-tuning have been compared in an agricultural context in [10]. It was found that fine-tuning was useful in teaching the model skills relevant to the domain and providing more succinct and precise answers. In turn, RAG was shown to be effective when the retrievable data is contextually relevant, such as interpreting data points. Balaguer et al [10]. highlight that the initial cost and extensive work required for fine-tuning is a significant consideration. Fine-tuning also requires a very large number of data points, in the order of thousands or even hundreds of thousands to be effective. These considerations mean that it would not make sense for Inclus to fine-tune for single projects, but to fine-tune a single model using all of the risk data and use this to ask questions about specific projects.

Like the prompt engineering techniques, these methods can also be combined for more accurate results. The performance of a pre-trained LLM combined with RAG was compared against a fine-tuned model with RAG and the combination of fine-tuning and RAG was found to produce more accurate results [9]. Fine-tuning can be done before integrating RAG and then combining the retrieval system with the trained generative LLM. RAG can also be a part of the fine-tuning process by having the retrieval system fetch the most relevant data for the LLM during fine-tuning. This fine-tuning method is known as Retrieval Augmented Fine-Tuning (RAFT) [11]. This method is meant to improve the fine-tuning process by using RAG to help avoid distracting, irrelevant data.

The effect of these techniques depends on the tasks that the LLM is used for. Therefore when choosing the reduction methods, the most important tasks as well as future tasks of the LLM should be considered and the specifics of each method studied. It is important to note that the performance results and comparisons in the above studies are highly specific to test setups, context, validation methods and metrics, used models, and fine-tuning parameters. This means that they are never directly applicable to a specific use case. Inclus should therefore evaluate the performance, benefits, and drawbacks of each hallucination detection and reduction method separately.

2.2 Prompt engineering

Prompt engineering refers to providing instructions in the prompt for the LLM to behave in a certain way. This could include specifically instructing the LLM not to state incorrect information if it is not confident in the answer or formatting the output in a specific way. We can also provide a role for the LLM to play, such as LLM being a risk analyst, and additional context on the risk data such as how the data is gathered and for what use in the prompt. Another prompt engineering technique is called few-shot prompting. Here the LLM is provided a couple of examples of what it is meant to do to increase accuracy. This way the model focuses on the specified topic. However, this technique is highly dependent on the quality of the examples. A final example technique for prompt engineering is Chain-of-Thought prompting. Here the LLM is instructed to generate in-between reasoning steps. This method has been shown to reduce logical errors and inaccuracies in the outputs.

These prompting techniques can all be combined easily, which should produce better outputs. For example, the LLM can be instructed to provide reasoning steps along with provided examples of reasoning steps. This same prompt can also include all the necessary formatting and context instructions. The above as well as several other prompt engineering techniques and their positive effects on LLM outputs have been studied in [12].

2.3 Use case assessment

As previously stated, ChatGPT has been explicitly used in risk assessment and management in [1]. The study examined two different construction projects, and the capabilities of the LLM were assessed to somewhat differ between them. In both projects, it was determined that the ability of GPT-4 to consider new potential risks from emerging trends and circumstances was high. The experts consulted in the study also found that the capabilities of GPT-4 in capturing relationships between identified risks were sufficient. It was concluded that GPT-4 was moderately suitable for providing support and suggestions on risk monitoring to aid project managers.

The accuracy of the risk identification by the LLM varied largely and is not recommended based on the results of [1]. Similarly, GPT-4's risk analysis capabilities were deemed poor, with the LLM being unable to consistently evaluate and prioritize risks in line with the project's objectives. This was also the case for determining appropriate risk response strategies.

No external data was provided to the LLM while completing the study. It is also important to note that the model used in [1] was the general GPT-4 model. This means that these results can potentially be significantly improved through providing specific risk data and fine-tuning.

This topic has also been studied more recently in [13]. Here the capabilities of GPT-4 were rated more highly in general, and at times even higher than human risk analysts. It is still concluded that the model lacks context-specific knowledge and that the provided risk management suggestions and strategies were too often vague and nonspecific and therefore unactionable. Indeed, LLMs are better served as a tool for human experts in the risk management context.

3 Prompt engineering setup

3.1 Language learning model

We are using Open AI's GPT-4 model [14] for all of the prompt engineering experimentation. It is the preferred model for this project since Inclus has previously used it. This model is one of the best LLMs currently available [15, 16] and all the research suggests that this model has the highest probability of giving optimal solutions for our questions. Other models were not tested because of the resource limitations of this project but the research suggests that other models are unlikely to yield significantly better results.

3.2 Data

The dataset used in this study is provided by Inclus. The data was collected as a questionnaire from Inclus employees and customers of Inclus. The data contains eight dimensions shown in Table 1:

Dimension	**Explanation**
Participant ID	Identification number or code assigned to each participant.
Participant name	Name of the participant involved in the session.
Session	Refers to the specific session or meeting in which the data was collected.
Your "department"	Department or division to which the participant belongs, if applicable. (e.g. customer, dev)
Item	The specific risk comment and valuation are about.
Dimension	The particular aspect, feature, or characteristic being evaluated or measured in relation to the risk. (Likelihood or Impact)
Answer	Participant's response or rating related to the risk on a scale from 0 to 5 (0 corresponds to smallest effect and 5 to largest).
Comment	Any additional remarks or explanations provided by the participant regarding their answer or evaluation. This is the only dimension that can be left empty.

Table 1: Dimension Explanations

Out of these dimensions, the relevant variables for our study are Item, Dimension, Answer, and Comment. These dimensions are chosen because using all the dimensions is impractical and therefore only the most meaningful dimensions are used for this study. Example entries can be seen from Table 2. All the examples shown in Table 2 are simulated and are not from the actual dataset used in the prompt engineering part because the real dataset is protected under an NDA.

Item	Dimension	Answer	Comment
Cybersecurity Threats	Impact	3.5	Identified potential vulnerabilities in the network infrastructure and implemented enhanced security measures to mitigate risks of data breaches.
Market Competition Increase	Impact	4	Increased competition might cause a decrease in sales.
Decreased Product Quality	Likelihood	2	Conducted product testing and received positive feedback from beta users, indicating high customer satisfaction with the new features.
Supply Chain Disruptions	Likelihood	1.5	Implemented contingency plans to address potential disruptions in the supply chain due to external factors such as natural disasters or geopolitical issues.

Table 2: Simulated example data

The total dataset contains 1097 entries and in 303 of those, there is also a comment. Some of the comments are very short and can not be analyzed very efficiently. Since we got the data directly from Inclus, the data set can be assumed to be reliable.

3.3 Benchmark questions

In this study, we use two strict benchmark questions and one soft benchmark question. The strict questions are 1. "What are the most likely risks / most impactful risks?" and 2. "What are the risks with the most comments?". The soft benchmark question is: "Write an executive summary of the results and risk analysis". Strict benchmark questions are easier to validate since the answers to these questions can be directly calculated from the data.

For the soft benchmark question, we validated the results by asking the LLM to provide all comments that were used to provide the summary and then manually going through the summary and checking that each of the provided comments indeed appeared in the summary. We also checked that the summary did not include any additional information obtained for example from data hallucination.

4 Summary Formulation

4.1 Methodology of summary formulation

Inclus' risk assessment datasets are long and time-consuming to analyze manually. The motivation behind this method is to summarise the comments of the data set into an executive summary, providing a text with the main risks facing a company. The benchmark question with this approach is for the LLM to generate an executive summary of the results of a risk analysis, a similar benchmark question is also used in Section 5. The idea of this part is to also compare whether converting numbers into words affects the quality of the summary. The complete benchmark question is given in the Appendix A.2.

To build this summary, we provide the LLM with an input structured as follows:

Roles and constraints + Background about data + Data + Expected output

The roles and constraints are the same for those specified in outlier detection. The background information regarding the data given to the LLM is provided below:

"Inclus is a Finnish scale-up company specializing in collaborative risk analysis. The dataset comprises assessments from various individuals within Inclus, covering multiple risk events. Each risk event is prefaced by a header providing context for the individual assessments."

To guide the LLM in generating the desired output, clear instructions for the expected output are provided:

"Provide only a few paragraphs of high-quality text accurately describing significant features of the data, such as notable risks and insightful comments."

The format of the actual data input is as follows:

Risk Type: "[...]", Dimension: "[...]", Answer: [...]", Comment: "[...]".

Each risk in the data set has a dimension, which consists of either impact or likelihood, a numerical answer, which corresponds to the dimension, and a comment. When building the summary, the LLM takes into account the dimension and answer corresponding to each risk and constructs the summary with the most informative comments.

With these roles and constraints, background information about the data, the dataset itself, and instructions regarding the expected output aligned with the benchmark question, we can generate an executive summary of a few paragraphs.

The LLM was also asked to count how many comments correspond to each risk, but in general, it performed poorly, since it was miscounting the number of comments by one or two in each iteration. This approach was not continued further. The full benchmark question is provided in Appendix A.3.

4.2 Validation

Validating the results of the soft benchmark question presented some challenges. The quality of the summary can not be easily verified, since different experts may have different opinions of what is considered a good summary. However, to validate the results, we carefully reviewed both the summary and the comments used, verifying their consistency.

To make the validation more accurate, we introduced specific restrictions on the prompt itself. Commands such as "Do not add any conclusion and comments or provide any additional data" and "The output should be solely based on the provided data" aimed to ensure that the summary relied solely on the available comments. Additionally, we made sure that for any conclusions drawn, all comments used in the assessments must be cited as sources beneath the summary. This was done by adding the command "For any conclusion made, provide all comments used in assessments as the source and write the sources underneath the summary" to the benchmark question. These measures were essential in confirming that the summary accurately utilized the comments, without any hallucination.

Despite these modifications, we observed that the LLM often used only around 10 out of 303 comments to provide the summary. To address this, we implemented a minimum requirement for the number of comments used. However, even with these adjustments, we noticed that the quality of the selected comments was not optimal. By manually analysing each comment, we noticed that some of the selected comments contained no essential information or were really short to actually provide meaningful analysis.

The comments used were not always written by experts or very informative. We addressed this issue by adding another constraint to the prompt stating that "Use mainly the comments that are written by experts and that have a strong argument". With these constraints, the summary was informative and well-written.

The validation process consisted of going through the generated text and cross-referencing each citation to ensure that the referenced comment indeed existed within our data file and was accurately incorporated into the summary.

4.3 Results

As a result of the soft benchmark question, an executive summary of the identified risks was provided. The summary appeared to accurately capture the potential risks given in the dataset. However, the correctness of a summary is difficult to measure, since different experts will have different perspectives on what is actually considered a valid risk. We could only validate the summary by assessing how the given data set was utilized.

By validating the results, we noticed that the summary consisted of over 30 comments that indeed were part of the data set. Most of the comments used were informative and seemed to be written by experts. In this approach, the LLM used the numbers to determine the quality of the comments, meaning the numbers were not converted to words before providing the LLM with the data set. The validation of the results was conducted manually by going through each comment and determining its quality.

5 Quantitative-to-qualitative transformation

5.1 Transformation method

The main motivation for this approach is that most LLMs are not able to analyze combinations of numbers and text very efficiently. This method aims to improve that by changing the numerical answers of the data to text. The data points are transformed in the following format:

Table 3: An example of the data before transformation.

Risk type	Dimension 2	Comment	Answer
Outsourcing risk	Impact	This would have very large impact	5

Table 4: An example of transformed data.

Risk type	Dimension 2	Comment	Answer
Outsourcing risk	Impact	This would have very large impact	Major

These transformed data points are then turned into text prompts. These prompts are input into the LLM and then we can ask the model questions regarding the risks the company is facing. This approach aims to enhance the quality of the LLM output and thus help the risk analyst to make better risk analysis. The conversions of the numerical values can be seen from table 5. Conversion words in Table 5 were chosen by testing which words produce the best result.

Table 5: Conversion of Numeric Values to Text

Dimension	**Converted Value**
Likelihood	
≤ 1	Insignificant
$1 < x \leq 2$	Small
$2 < x \leq 3$	Medium
$3 \leq x \leq 4$	Likely
> 4	Very Likely
Impact	
≤ 1	Insignificant
$1 < x \leq 2$	Small
$2 < x \leq 3$	Medium
$3 \leq x \leq 4$	Large
> 4	Major

Now the final prompt is in the form:

```
Restrictions+Benchmark question+Risk type+Comment+textual valuation of the answer
```

For this approach, we use the same restrictions as in outlier detection and two different benchmark questions:

“What are the X Most likely risks / most impactful risks?”

”Write an executive summary of the results of the risk analysis”

The final prompt could look like this:

Restrictions+What are the 5 Most likely risks+ Risk type: Outsourcing risk+ Comment: This would have very large impact + Impact: Major

The final result is printed out in a single-string text.

5.2 Validation

The first part of the validation was done using the first strict benchmark question: “What are the most likely risks / most impactful risks?” The first strict benchmark question is relatively easy to validate analytically since we have numerical data to find the most impactful and most likely risks and we can compare the results of the analytical numeric data to the answer the LLM gives us.

The second part of the validation was done using the soft benchmark question: “Write an executive summary of the results and risk analysis”. This part is not possible to validate analytically and thus the validation method requires the validator to examine the answer that the LLM outputs and evaluate if the answer is good enough.

5.3 Results

The LLM was asked to provide a number of the most serious risks based on the given data without performing the Qualitative-to-Quantitative transformation first. The data presented in Table 6 highlights the effectiveness of the Qualitative-to-Quantitative transformation approach in identifying and ranking risks. Accuracy, in this context, refers to the percentage of risks correctly identified within a specified range and was evaluated through manual verification.

Table 6: Validation of the Quantitative-to-Qualitative approach.

Risks asked	Correctness
1	100%
2	100%
5	80%
10	75%

The results indicate that the model's accuracy decreases as the number of requested risks increases. This can be explained by the fact that it is harder for the model to analyze a larger amount of risks and rank them. Overall the accuracy seems to be fairly good and this method could be used as a tool for the risk analyst.

The results of the Quantitative-to-Qualitative approach indicate that the approach increases the accuracy of the model slightly. From Table 7 we can see a version where the data is input in an untransformed form. If we compare Tables 6 and 7, we see a small increase in accuracy, around 15 percent when the number of top risks required increases.

Table 7: Benchmarking results with untransformed data.

Risks asked	Correctness
1	100%
2	100%
5	60%
10	60%

Overall, even though the accuracy is increased we can not explicitly say that the method increases accuracy since the data set used for this study is relatively small and the increase in accuracy is also fairly small. To get more affirming results, the methods should be applied to multiple different types of datasets.

For the second benchmark question: "Write an executive summary of the results and risk analysis" for this dataset, the answer was that the model output seemed to be very good and accurate. The main risks were highlighted and they seemed to correlate with the data. The model also gave some suggestions on how to mitigate the biggest risks but the suggestions were not very good. The accuracy of the validation could be better if the people analyzing the answers were more experienced than the authors of this project. The summary all in all was quite similar to the one presented in Section 4. It is hard to determine how much of an improvement there was and the quality of a summary is hard to evaluate in general.

Results could be improved by doing more testing on defining the keywords defined for each numerical value. The used LLM also has a major impact on the results and as the LLM develops further the results of this approach are expected to improve.

6 Outlier detection approach

6.1 Methodology of outlier detection

The outlier method aims to find the comments that do not match the valuation given to them by the participant. The main motivation of this approach is that sometimes the participants may give numerical valuations to certain risks that do not match the comment. The LLM is used to change the comments into numerical values between 0-5 based on the wanted dimension. For example, the LLM could change the following data input:

Original data:

Risk type	Dimension	Comment	Answer
Outsourcing risk	Impact	This would have very large impact	1

Transformed data:

Risk type	Dimension	Comment valuation	Answer
Outsourcing risk	Impact	5	1

After this, the algorithm will look out all the transformed data inputs where the absolute difference between "Answer" and "Comment valuation" is more than 1 and mark those comments as outliers. The risk analyst can then choose to remove those data points to get better results or change the "Answer" to correspond better to the given comment. This could enhance the accuracy of the risk analysis and lead to better results.

The transformed data points can also be used to find the biggest risk that the company is facing. This can be done by sorting all the risks by "Comment valuations" and finding the risk that has the highest impact and likelihood. Finding the biggest risk can be beneficial for the risk analyst when creating a risk analysis since finding the biggest risk is one of the most important aspects of risk analysis.

Raw data is input for the LLM model in this format:

1. Role and constraints + 2. Reference data + 3. Data

1.The role and constraints part contains basic information about the role of the LLM model:

"The assistant is a risk analyst. The data that is provided below is gathered from multiple people within the Company and concerns multiple risk events. Each risk event is described in a header above the individual assessments. Treat each comment equally. Give impact or likelihood for each comment (based on the Dimension) in a scale 0-5 (number can have decimals), print the comment and add the assessment after it, and print the risk the comment is related to"

This section of the prompt has been added to ensure that the LLM can properly handle the data that is fed into it. Without this part the LLM returns a lot of outliers and the model is not functional [12].

2. The reference data part contains some reference data for the model. The reference data is simulated by the GPT model in a separate LLM-query and the simulation can be completely automated:

> “Simulate example comments with valuations between 0-5 based on the comments given to you”

References are given in this form:

> “Here is example of comments and valuations: Impact 0: ”This feature has no significant impact on user experience.”

Providing example comments with corresponding valuations was found to be useful since without it, the model tends to only give valuations close to 3. This shows that adding example comments improved the quality of the outputs.

3. Finally, the data is provided to the model in the following format:

> “IT security risk, Comment: Our company is well prepared for IT risks”

The prompt is then input into the LLM in a one-string element. Example prompts without data is provided in appendix A.1. The results are also printed as a one-string element in the following format:

> ***Risk type***Comment***GPT’ valuation for the comment.

The outliers or the biggest risk can then be found by comparing the numerical values given by the LLM.

6.2 Validation

Validation of the Outlier detection approach was done by comparing the numbers given by the LLM for each comment and then comparing those numbers to the actual valuations given by the participants. If the difference between these numbers is greater than one then the comment is marked as an outlier. The marked comments are then looked through manually and checked whether they were outliers.

Outlier detection can also be used to find the biggest risks by using the numerical results that the comments were transformed into. All the listed risks were put into order by the numerical values given by the outlier detection. Then the list was checked and compared to the values given by the participants.

6.3 Results

Out of the 303 comments, the model detected 21 outliers. Many of the outliers were caused by short comments. For example:

Risk	**Dimension**	**Valuation**	**Comment**
Partner risk	Impact	3	This is not good

This kind of comment cannot be evaluated by the LLM very reliably since it does not contain a lot of useful information and even an experienced risk analyst could not make an assessment based on this. These kinds of outliers should be ignored.

There were some cases where the method found outliers:

Risk	**Dimension**	**Valuation**	**Comment**
Cyber security risk	Likelihood	3	This risk just occurred

Here we can notice that the evaluation of likelihood might be off since the valuation of the risk likelihood is only 3 but the comment says that this risk has just been realized. The LLM rated the risk likelihood to 5 based on the comment. The risk analyst might consider changing the valuation of the answer to higher to give more accurate results.

Currently, there are a lot more false flags than there are correctly detected outliers. Current accuracy is less than 10%. This means that it is most likely not worth it to the risk analyst to go through all the marked comments since the cost of time exceeds the possible benefit.

Table 8 shows that this approach is not very accurate for finding the biggest risks that the company is facing. The "Correctness" is checked by manually checking if the risks that the LLM lists are within the biggest risks. The accuracy is bad and only increases as the more the model lists more risks because as the number of risks increases the probability that the model finds the biggest risks increases by sheer luck.

Table 8: Outlier detection approach validation table

Risks asked	Correctness
1	0%
2	50%
5	40%
10	40%

One way to improve this model would be by using the logprobs introduced in [5]. The parameter Logprobs provides a method to validate the confidence of the LLM as described in Section 2. In outlier detection, this could be used to check how confident the model is when it outputs the numerical evaluation for the comment. The model could then ignore all the results where the models confidence is under a certain parameter. This could remove all the false flags that the method produces because some of the comments are very short or uninformative.

Overall this approach is not recommended for datasets that have very short or uninformative comments. For datasets with very long comments, this could be a possible approach. The main motivation of this approach was to increase the accuracy of risk predictions but the results indicate that this approach would decrease the quality of risk analysis.

7 Feature extraction approach

7.1 Feature extraction method

In this approach, we are not directly interested in how to incorporate numbers into the LLM. We are rather interested in how the LLM can be used as a feature extraction model to be able to give a fast and efficient overview of the comments in terms of a set of features. Given a useful set of features, this approach can be particularly interesting for Inclus, because it would allow filtering the comment and leaving only those that are "trustworthy". There is also a possibility to give weight to each comment depending on the set of features. For example, if the comment is identified as "the author is not an expert" or the author is not "objective", then the weight of such comment in the overall assessment can be set to be less. One more way to apply this approach is to train a machine learning model given enough amount of comments that would be able to predict the likelihood or impact of some risk.

The main idea is as follows: given a set of comments, the LLM should be able to assess at least the next features:

- keywords
- semantics of the comment, meaning is the comment overall positive, negative, or neutral

- objectivity of the author, meaning does the author's valuation correspond to comment
- expertise, meaning does the author seem proficient in the subject
- argumentativeness, assessing the quality of the argument of the comment argument

The main problem with this approach arises in the validation of the LLM output since we are working with a black box model. There could also be some potential problems with the hallucination of results, which would be fatal since we are constructing a model for risk analysis. That is why this approach requires attentive prompt engineering.

There are two possible approaches to get the embedding for each comment. The first approach to the problem can be formulated as follows. Formulate a prompt that would consist of some system content and a benchmark question. System content contains background about the company to be analyzed, context information about the data itself, and constraints for the output. Constraints are added to ensure that the LLM for each feature outputs a value only from the allowed list of values ("positive"/"negative"/"neutral") or a set of words.

The prompt also does not have to answer the benchmark question if it is not possible. This is so that we can limit the hallucination of data. The embedding for each comment is formatted as an asterisk-separated list to ease the parsing of output. Then "positive"/"negative"/"neutral" values are mapped to 1/-1/0 numerical values in the same order. The benchmark question is formulated as follows "For each comment provide an embedding". The whole prompt can be found in Appendix A.4.1.

This approach is computationally more or less efficient, since it takes only 4 minutes to provide embedding for 40 comments. However, the trustworthiness of these embeddings is under question since the user is not given a justification for each embedding.

The second approach is computationally heavier since for the same set of 40 comments it requires prompting twice. At first, the comment is asked to be analyzed by the LLM model to provide reasoning. The analysis should contain answers to these questions:

1. Is author objective, meaning that they are realistic and not overpessimistic/overoptimistic
2. Is comment argumentative, meaning that the risk is discussed thoroughly
3. What is the semantic of comment, meaning if the comment is negative, positive or neutral
4. Is author an expert in risk analysis and are they able to assess this risk

The LLM would be restricted to providing only a short summary of 2-3 sentences. The analysis can include only information that we request and nothing else. This prompt can be found in Appendix A.4.2.

Example of LLM analysis:

"The comment has a neutral sentiment, recognizing the need to plan for partner problems. It is realistic—anticipating issues without overreacting. Expertise in risk management and the impact of partner reliance is suggested."

When we provide LLM with this analysis and ask it to provide the embedding, it concludes that the author is an expert, the sentiment is neutral, and the comment is objective and argumentative.

This approach can be more applicable to risk analysis since there is an explanation behind the embedding and people are more willing to believe specific reasoning. The second approach also can be used as a validation for the first one, since in the ideal case output should be the same.

There is still a potential problem with such an approach. Namely, we are requesting LLM to provide an assessment of features, for example, what is the objectivity of comment? Objectivity can be interpreted in different ways and even people can provide different definitions. So, while constructing the prompt we should make sure that LLM is given context about the requested feature. We also had an idea that we could fine-tune the LLM model by providing comments and our assessment as examples. However, when we tried out this approach and conducted the literature review, we concluded that the provided dataset was not large enough to formulate enough examples to significantly affect the quality of the results.

The overall structure is as follows: background and context information, comments provided as an indexed list along with corresponding risk type, dimension, and benchmark question asking about embedding.

7.2 Validation

It is really important to validate these two approaches so that the user is able to understand all of the risks of relying on the output of LLM. If the quality is measured accurately and reported to the decision maker, it would be the responsibility of the decision maker. That is why we are putting a lot of emphasis on the validation of these embeddings.

The first approach is to manually assess the embedding for each comment. This approach is problematic since we are not experts in text analysis, so we can not be completely sure about our assessment. This process also takes a substantial amount of time, so only a limited number of comments can be accessed. Later on, if needed, Inclus can refer to expert supervision to access these features and properly validate results with this approach. We still assume that our assessment is good enough to check if the LLM model output makes any sense. We compare this assessment with both approaches in prompting. Two different outputs of LLM will be also compared to each other. As a metric for this approach, we would use accuracy. The reasoning behind choosing accuracy is that it can be easily interpreted.

The second approach is to use *logprobs* and treat the embedding as a classification problem. This prompt can be found in Appendix A.4.3. For each comment, the LLM is asked to provide answers separately and answer only with one word. The GPT-4 model has a built-in possibility to check what was the log probability of the output token and if the number of options for tokens is restricted to a set of words, it can be considered as a classification problem. If the LLM model is often not sure of the output token, we would conclude that the assessment of a particular feature is not accurate enough or risk analyst should take into account. Example of the prompt:

"You will be given a comment about given risk type and either impact or dimension of the risk. Return only the answer to the question and nothing else. MAKE SURE your output is one of the options stated by question.
Comment: {comment} Question:{question}"

We also use *logprobs* to assess the effect of hallucination. As it was already mentioned before hallucination occurs when LLM is requested to provide an answer question to which it doesn't know the answer. The LLM is generally good in the context assessment, so it can be directly asked if there is enough information to answer the question fully. It is be restricted to answering either "true" or "false". If the model tends to output "false", Inclus would need to take measures to ensure that the comments are detailed and qualitative enough, otherwise, LLM would hallucinate its analysis. Example of the prompt (the whole prompt can be found in Appendix A.4.4):

"You will be given a comment about given risk type and either impact or dimension of the risk. You will be given also a question about the comment. Before even answering the question, consider whether you have sufficient information to answer the question. Respond with just one word, the boolean true or false. You must output the word 'True', or the word 'False', nothing else. Respond 'True' if there is enough information and 'False' if it is not enough.
Comment: {comment}, Question: {question}"

We validate the quality of the analysis of comments by using the approach explained in the paper about *SelfCheckGPT* in Section 2. For each comment, we prompt the LLM to produce an analysis sample 5 times. The hallucination effect is then analyzed based on MQAG, BERT, and Ngram scores. Here N refers to the length of the phrases. This can be seen in detail in the prompt in the Appendix A.4.5. The functions that compute these metrics are implemented by the authors of this paper and it requires installation of their library. This method helps to assess hallucination since the outputs with strict constraints and a low-temperature model tend to be approximately the same. Higher temperature makes models be more creative in its responses. If the variability in the sentence is too high, it signals that there is not enough information to answer the question and therefore the model hallucinates. This approach can be useful in identifying which features can be evaluated by the LLM. While validating with this approach we should make sure that the output format and order of the answers in the analysis are the same, so that it is easier for Inclus and us to interpret the results.

7.3 Results

At first, we report the accuracy scores of the LLM output which was based on the comparison with our own assessment. It should be noted that these numbers are based only on 40 comments and our manual assessment is not 100% accurate. In Table 9, you can find how similar our assessment of numerical features is to LLM assessment based on comments directly. The keywords were validated manually and they are meaningful and not hallucinated.

Table 9: Accuracy validated by comparing the embedding provided by the LLM based on the comments and our manual assessment.

	Expert	Semantic	Objectivity	Argumentative
Accuracy	71.05%	34.21%	81.58%	68.42%

The accuracy results imply that the LLM is able to predict the correct 70-80% embedding for the comments for features “expert”, “objectivity”, and “argumentative”, which can be considered a satisfactory result. The result for the semantic features is much poorer, which could be due to the model guess. This can be explained by the fact that not all comments are of good quality and detailed enough to identify semantics properly. Also, our assessment can be confusing since we had a tendency to mark comments as “neutral semantic” and the LLM marked some of these comments as “positive” semantic. The LLM sentiment assessment should be good enough since it is listed along the primary functions of LLMs. So, the problem may lie in our manual assessment.

The second approach is more difficult to interpret since it relies on prompting the LLM twice and the embedding quality directly relies on the output of another prompt. The quality of the analysis sample is also under question. Even a slight change in the prompt that formulates how the analysis of a comment should be done leads to either a significant drop or an increase in accuracy. In Table 10, in the first row, you can see how similar the embedding of two LLM outputs is. In the ideal case, we would expect them to be close to 100%, which is not the case. Based on this assessment, LLM embedding cannot be considered reliable. This can be possibly explained by the effect of hallucination. In the second row, you can find the comparison of the second approach with our manual assessment.

Table 10: Comparison of embedding produced based on the analysis samples of each comment with our assessment and first approach.

	Expert	Semantic	Objectivity	Argumentative
First approach	81.58%	63.16%	65.78%	81.58%
Our assessment	73.68%	52.63%	73.68%	71.05%

As you can see from Table 10, when the LLM makes conclusions about the features based on its own analysis, it tends to give embeddings that are closer to our assessment than a direct assessment of the features except for objectivity. We highlight that the semantic assessment with 'analysis first, conclusions later' is much better and was not identified by luck.

The LLM model was then requested to judge if there was enough information to answer the posed questions about comments confidently. The results can be found in Table 11. These results suggest that the comments are not detailed at all and the context that they provide about each risk type is not enough to draw any confident conclusions. We also noticed that the features where LLM sometimes marks the comment as “good enough” to measure 'argumentative' or 'semantic' appear to have lower confidence than with other features.

These validation results suggest that the comments are not good or long enough for analysis and some time should be spent on collecting data of higher quality.

We then applied a *logprobs* approach to see how confident the model is in its assessment of the features. For each comment, the question about each feature was asked separately. The average confidence of the tokens was calculated based on the output probabilities and can be found in Table

Table 11: The fraction of comments where the LLM suggests that the comment has enough information to answer questions. The output probabilities are provided as well.

	Expert	Semantic	Objectivity	Argumentative
Fraction of true values	0.00%	2.00%	0.00%	23%
Confidence	100%	94.42%	99.10%	88.67%

12. Analysis was made based on 100 comments. This time for expert, objective, and argumentative fields the LLM could also answer "maybe".

Table 12: Average confidence of prediction of the token for each feature.

Objectivity	Argumentative	Semantic	Expert
77.89%	83.66%	89.99 %	83.84 %

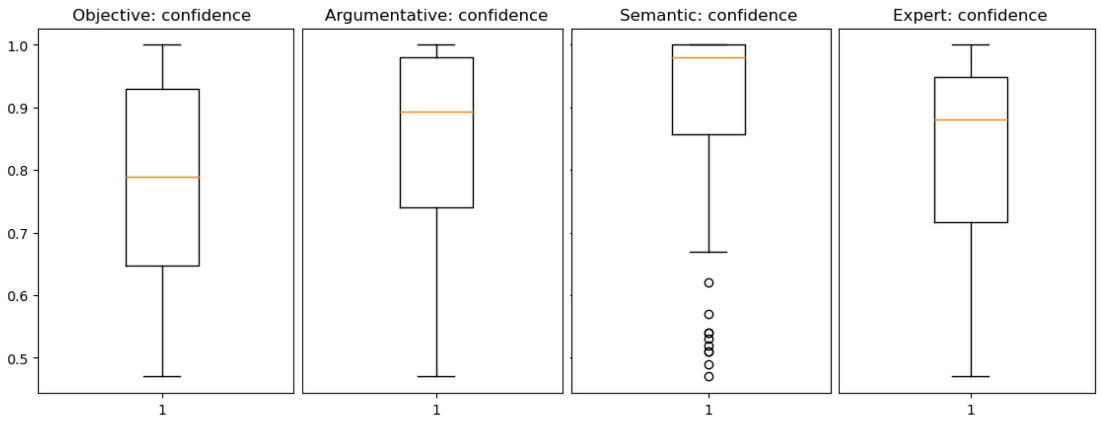

Figure 1: Box plots of confidence values associated with each feature.

As you can see from Table 12, the LLM model is ∼ 80% confident in the tokens that it outputs. It can be useful to look at the box plots (see Figure 1) of how confidence values are distributed. We notice that the semantic feature has many outlying samples with low confidence (around 60-70%). Since the model is 100% confident with other samples, the average for semantic feature assessment is high. This finding confirms that semantics is the hardest feature for assessment by LLM.

The distribution of labels can be found in Figure 2. The distributions of answers given by LLM are reasonable. It has much confusion in defining if the author is an expert or not because the comments are not detailed enough. Also, the LLM suggests that comments poorly communicate the argument about the risk type they describe. The majority of comments have neutral or negative semantic which makes sense. The model in the majority of cases is not sure if the author is objective or not because this feature is a bit difficult to identify. A large proportion of 'maybe' answers in the assessment of expert and objectivity features also highly correlated with the fraction of comments that the LLM thinks do not contain enough information.

The second method of generating analysis samples first and the embedding later should also be validated. The main cornerstone of this approach is the quality of analysis samples. This is why when using sampling, we see how the prompt outputs vary. The method is computationally heavy, which is why we tested using only 10 comments. These metrics assess each sentence separately and since each analysis answers the same set of questions in the same order, we can actually interpret the values as the LLM's ability to evaluate the feature. MQAG and BERT scores are scaled to 0-1, where 0 means that the text sample is factually correct, 0.5 means that there are some inconsistencies and 1 represents non-factual text. Ngram score is a bit different since it is not upper-bounded, so we would not be able to make sense of the values themselves. The only thing we would be able to do is to compare scores for each sentence with each other.

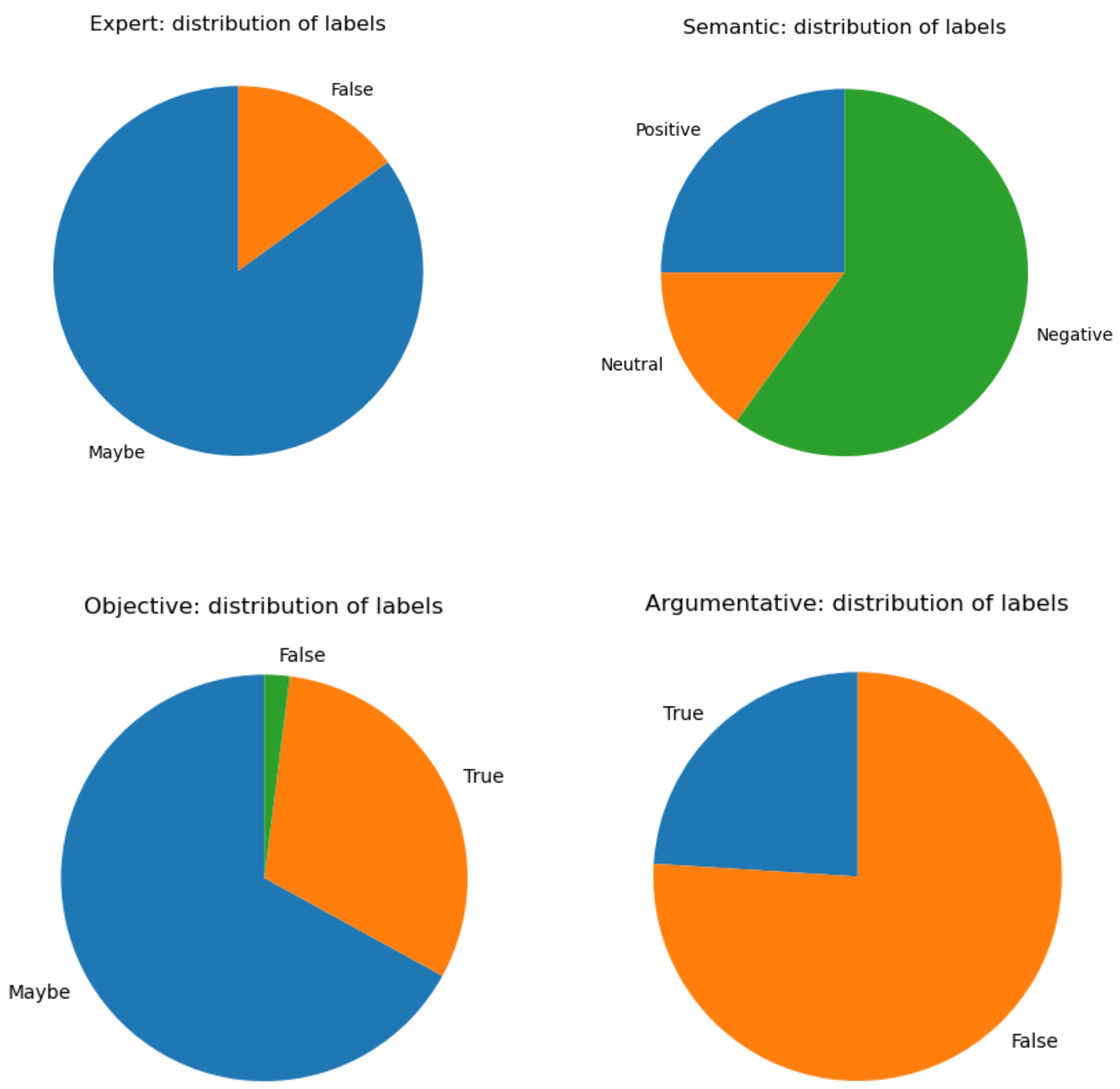

Figure 2: Distribution of labels given by the LLM model.

Table 13: MQAG scores for 10 comments based on 5 samples.

#	Objective	Argumentative	Semantics	Expert	**Mean**
1	0.3112	0.1478	0.5470	0.4718	**0.3695**
2	0.3541	0.3005	0.0385	0.3249	**0.2545**
3	0.2925	0.3448	0.0057	0.1636	**0.2017**
4	0.5708	0.4682	0.1403	0.3002	**0.3699**
5	0.0727	0.2114	0.5747	0.0751	**0.2335**
6	0.2996	0.2460	0.3100	0.2488	**0.2761**
7	0.1255	0.0075	0.5350	0.2795	**0.2369**
8	0.0017	0.0110	0.2431	0.5140	**0.1924**
9	0.0002	0.0000	0.3475	0.3399	**0.1719**
10	0.0015	0.0328	0.3965	0.0864	**0.1293**
Mean	**0.2030**	**0.1770**	**0.3138**	**0.2804**	

MQAG scores in Table 13 suggest a small impact from hallucination in the analysis of samples (0.5 score corresponds to a slightly in-factual sample). The highest score corresponds to semantics, which corresponds to our previous finding that the LLM tends to make many mistakes with semantics features compared to the manual assessment. The results for the objective and argumentative features are good enough. The expert feature is also hallucinated a lot in comparison to objective and argumentative. A manual check of these samples showed that the LLM tends to interpret why it is not sure if the author is an expert in different ways.

Table 14: BERT scores for 10 comments based on 5 samples.

#	Objective	Argumentative	Semantics	Expert	Mean
1	0.2215	0.1193	0.2348	0.2952	**0.2177**
2	0.2325	0.3640	0.1848	0.1527	**0.2335**
3	0.0180	0.1564	0.2335	0.2745	**0.1706**
4	0.3711	0.4348	0.2404	0.4198	**0.3665**
5	0.2004	0.1903	0.2337	0.3062	**0.2326**
6	0.5301	0.3895	0.4322	0.1596	**0.3779**
7	0.3129	0.2228	0.3435	0.2669	**0.2865**
8	0.0487	0.2254	0.2338	0.3088	**0.2042**
9	0.4190	0.0954	0.3435	0.5646	**0.3556**
10	0.0194	0.1492	0.3186	0.5217	**0.2522**
Mean	**0.2374**	**0.2347**	**0.2799**	**0.3270**	

Based on the BERT scores in Table 14, we can come to the same conclusion except that the BERT scores for the Expert feature tend to have more hallucination than Semantics. BERT scores and MQAG scores can be actually used for the detection of low-quality comments by looking at the mean values corresponding to each comment. For example, comment # 4 "[...] concerns are likely to become even more important" was reported to be highly hallucinated by both metrics.

Table 15: Ngram scores based on phrases of length 1 (n=1)

Objective	Argumentative	Semantics	Expert
4.41124356	4.318532	4.35048047	4.49826405

A smaller Ngram score implies a better result (see Table 15), yet we are unsure how to interpret the Ngram score in the case of our dataset. The differences between the features are not significant and these values do not correlate much with the previous two metrics. The Ngram score indicates that LLM tends to hallucinate the most when it assesses the expertise and objectiveness of the comment.

8 Discussion

During our research and testing, we found that LLMs have clear limitations due to their current capabilities. For instance, when tasked with finding the risk with the most number of comments based on a given dataset, the LLM consistently failed despite the analytical simplicity of the task. The primary goal of LLMs is to be a language model that is able to generate text, do completions tasks and summarization, so it can not be used as an analytical tool.

The reliability of our results varied a lot, depending on the design of the prompts and the nature of the questions we asked. This variability highlights that prompt engineering crucially impacts the LLM's output, suggesting that LLMs should not be relied upon for risk analysis unsupervised due to the variability in performance.

LLM did show potential in feature extraction. It can at least provide a meaningful explanation about its assessment and the hallucinations can be controlled. The only limitation of this approach was the dataset itself. If the comments were of higher quality, the LLM model would be more confident in its answers and provide useful insights about the comments for decision-makers in a short time.

LLMs' are currently developing at a fast pace and the risk analysis properties will most likely only increase in the future. All of the methods described in this paper are viable even though the LLMs are developing and will most likely yield even better results as the models improve. None of the methods described in this paper are specific to the used model and the same prompt engineering techniques are viable for different types of LLMs as well. There are many more ways to engineer prompts and validate results. While manual validation is common, this report also shows how to automate the process. The techniques used here are inspired by recent research papers. It is important to mention that most papers on this topic are quite new, mostly from 2023 or 2024, which means they have not been reviewed by many peers yet.

Even though LLMs should not be used alone to do risk analysis, in the future this might be possible. LLMs contain enormous analytical capabilities and can internalize more literature than any human being ever could. This could potentially create better risk analytics than would be possible using human brains. This development is still quite far from finished and this kind of performance might not be seen in decades.

It is good to consider the ethical consequences as well. When the use of AI increases in the future new kinds of questions arise. If an AI fails to notice a crucial risk and something terrible happens, who has the responsibility? Right now this might not be topical but in the future it most likely will be. It would be beneficial to solve these kinds of issues before AI tools develop any further. Even though AI tools contain a lot of problems to be solved, the overall potential outweighs the negative aspects, and further development should not be restricted because of these issues. AI tools can not create any safety issues as long as there is still a human eye that overlooks all the decisions the AI makes. As AI tools develop, we should not start to trust them blindly.

9 Conclusions

The overall results indicate that LLMs can be valuable tools for Inclus' risk management process. Prompt engineering approaches presented in this study can be used to enhance the overall accuracy of the outputs. The key to having high-quality results with an LLM is having a very large amount of high-quality data. The best approaches to detect and reduce hallucinations are RAG and fine-tuning. These are both dependent on the quality and amount of data provided. Prompt engineering is a valid approach but by itself is not nearly as powerful as the above methods and is similarly highly dependent on the quality of the comments in the risk assessments.

The summary formulation method provided an executive summary of the results of the risk analysis. The validation of the summary could be further improved and possibly automatized. However, with a quite small summary, this is not necessary. Longer texts or summaries may need automated validation to avoid the tedious manual work that would arise. The LLM effectively assessed the nature of each comment, recognizing expert input and essential information. However, whether these comments are actually relevant, could also differ between experts.

The quantitative-to-qualitative approach showed that it might have some potential since according to tests it performed better than the version where quantitative and qualitative data were mixed. Quantitative-to-qualitative could be developed further by using better-defined algorithms for changing numerical data into textual data. These results are not conclusive since the testing was only done for one dataset.

The outlier detection method could be applied for datasets that have long and well-explained comments but in most cases using this method will most likely only decrease the analysis quality. If the risk analysis data has hundreds of data points, few outliers will not make significant errors, and thus using this method is not recommended.

Our research of the feature extraction approach suggests that it is better to generate embedding based on analysis of comments than comments themselves. The downside is that it is computationally more costly since it requires prompting twice. Also, the features can be reliably assessed if the dataset is of good quality, which means that the comments are meaningful and detailed enough. Once the embeddings are generated, they can be reliably validated by suggested methods. Our experiments also suggest that if the risk analyst wants to reliably asses features with LLMs, then they should also define the feature. The set of features also can be expanded by prompting the

LLM model if it is enough information to asses this feature. If the fraction of comments where the feature can be accessed is big enough, the feature can be placed on a list.

The feature extraction method can be useful for Inclus for several reasons. It would improve analysis since we can concentrate on comments with particular embeddings. For example, a risk analyst wants to predict the likelihood and impact of a risk based only on comments written by "experts". Based on these embeddings, during the calculation of the likelihood and impact, comments can be given different weights. For example, a comment with a negative value for semantic, argumentative, and objective, that is written by an expert can have higher weight than a non-objective comment with poor argument. Another way to use those embeddings by Inclus can be to train a machine learning model that would be able to predict likelihood and impact values. In this case, we should ensure that the list of features is large enough and that there are enough comments to make any conclusions. Training a machine learning model can be useful if the person assessing risks does not provide a number associated with a comment or for outlier detection.

We found the LLM to perform quite poorly at tasks such as counting the number of provided comments. We suggest using LLMs for tasks such as providing fast insights regarding the data by assessing a set of features about the data or providing a summary of the risks based on the survey data. Summaries would still require implementing hallucination detection or manual checking. LLM could also be used as a tool for identifying poor-quality comments. These can be expanded on further as the generative language learning models' capabilities improve with time as well as after implementing and refining methods and techniques that improve the performance of the LLMs.

We suggest continuing by testing the *SelfCheckGPT* sampling method for detecting hallucinations, as this requires no internal knowledge of the LLMs workings or external data and is therefore quickly implementable with mainly a computational cost. Later on, improvements can be made via fine-tuning the model using data from multiple projects to ensure the training set is large enough and integrating the project data with the generative LLM through a method such as RAG.

The source code of the project can be found in this Github repository: Link

References

[1] H. Aladag. Assessing the accuracy of chatgpt use for risk management in construction projects. *Sustainability*, 15(22), 2023.

[2] Z. Xu, S. Jain, and M. Kankanhalli. Hallucination is inevitable: An innate limitation of large language models. 2024.

[3] J. Li, X. Cheng, W. Zhao, J. Nie, and J. Wen. Halueval: A large-scale hallucination evaluation benchmark for large language models. 2023.

[4] P. Manakul, A. Liusie, and M. Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv:2303.08896*, 2023.

[5] OpenAI. Using log probabilities, 2022. URL `https://cookbook.openai.com/examples/using_logprobs`. Accessed: May 13, 2024.

[6] S. M. Towhidul Islam Tonmoy, S. M. Mehedi Zaman, V. Jain, A. Rani, V. Rawte, A Chadha, and A. Das. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv:2401.01313*, 2024.

[7] F. Thomas Heston and C. Khun. Prompt engineering in medical education. *International Medical Education*, 2(3):198–205, 2023. ISSN 2813-141X. doi: 10.3390/ime2030019.

[8] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Advances in neural information processing systems. pages 9459–9474, 2020.

[9] J. Mathav Raj, V.M. Kushala, H. Warrier, and Y. Gupta. Fine tuning llm for enterprise: Practical guidelines and recommendations. *arXiv:2404.10779*, 2024.

[10] A. Balaguer, V. Benara, R. Luiz de Freitas Cunha, R. Estevão Filho, T. Hendry, D. Holstein, J. Marsman, N. Mecklenburg, S. Malvar, L. Nunes, R. Padilha, M. Sharp, B. Silva, S. Sharma, V. Aski, and R. Chandra. Rag vs fine-tuning: Pipelines, tradeoffs, and a case study on agriculture. 2024.

[11] T. Zhang, S. Patil, N. Jain, S. Shen, M. Zaharia, I. Stoica, and J. Gonzalez. Raft: Adapting language model to domain specific rag. *arXiv:2403.10131*, 2024.

[12] S.M. Bsharat, A. Myrzakhan, and Z. Shen. Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4. *arXiv:2312.16171*, 2024.

[13] R. Nyqvist, A. Peltokorpi, and O. Seppänen. Can chatgpt exceed humans in construction project risk management? *Engineering, Construction and Architectural Management*, 31(13): 223–243, 2024.

[14] OpenAI. OpenAI GPT-4 Turbo and GPT-4 Documentation, 2024. URL `https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4`.

[15] A. Borji and M. Mohammadian. Battle of the wordsmiths: Comparing chatgpt, gpt-4, claude, and bard. *GPT-4, Claude, and Bard*, 2023. June 12, 2023.

[16] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F.L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv:2303.08774*, 2023.

A Prompts

A.1 Outlier detection

The assistant is a risk analyst. Inclus is a Finnish scaleup company that provides a platform for doing collaborative risk analysis. The data that is provided below is gathered from multiple people within Inclus and concerns multiple risk events. Each risk event is described in a header above the individual assessments. Treat each comment equally.

HERE IS EXAMPLE OF COMMENTS AND VALUATIONS: Impact 0: "This feature has no significant impact on user experience." "The change will have negligible impact on system performance."

Impact 1:"Minor improvements may have a slight impact on customer satisfaction." "The update is expected to have a minimal impact on overall sales."

Impact 2:"This decision could have a moderate impact on project timelines." "Changes in marketing strategy may have a moderate impact on brand visibility."

Impact 3:"The proposed changes will have a noticeable impact on production efficiency." "Increased competition may have a significant impact on market share."

Impact 4:"The new feature is expected to have a substantial impact on user engagement." "Market trends suggest a high impact on pricing strategies."

Impact 5:"The security breach had a severe impact on customer trust." "Natural disasters can have a catastrophic impact on supply chain operations."

1. "ITEM: EXAMPLE RISK, EXAMPLE COMMENT, EXAMPLE DIMENSION ....."
2. "ITEM: EXAMPLE RISK, EXAMPLE COMMENT, EXAMPLE DIMENSION ....."
3. "ITEM: EXAMPLE RISK, EXAMPLE COMMENT, EXAMPLE DIMENSION ....."

Benchmark question:

Give impact or likelihood for each comment (based on the Dimension) on a scale of 0-5 (number can have decimals), print the comment and add the assessment after it, print the risk the comment is related.

A.2 Soft benchmark question

Write an executive summary of the results of the risk analysis. Do not add any conclusion and comments or provide any additional data. The output should be solely based on the provided data. Use mainly the comments that are written by experts and that have a strong argument. For any conclusion made, provide all comments used in assessments as the source and write the sources underneath the summary. The amount of sources used should be at least 45.

A.3 Counting the amount of comments corresponding to each risk

List the risk with the most total comments, and list the risks in an order with the most amount of comments to the least, the output format is

*Risk*Comment*

A.4 Feature extraction

A.4.1 Approach 1 – Direct assessment

Assistant is an expert in risk analysis (it can judge about impact and likelihood of risk based on the set of comments) and natural language processing expert, so it can evaluate parameters associated with text features. It can make conclusions only based on the provided comments. If assistant is not sure in the answer, it can return text "None" The assistant would be provided a list with indexes, risk type and comment it should answer on posed benchmark question. Do not add any additional text except for required output format. Inclus is a Finnish scaleup company that provides a platform for doing collaborative risk analysis. The data that is provided below is gathered from multiple people within Inclus and concerns multiple risk events. Treat each comment equally.

1. Risk type: [...]. Dimension: [...]. Comment: [...]
2. Risk type: [...]. Dimension: [...]. Comment: [...]
3. Risk type: [...]. Dimension: [...]. Comment: [...]

Benchmark question:

Find embedding for all comment in the provided list (for all 38). Only semantic can take neutral value. Each comment should be accessed next features with regards to assoicated risk type: objectivity of author(positive / negative) is person over (optimistic or overpesimistic which is "negative" or realistic which is "positive"), argumentative(positive / negative) how well it corresponds to risk type, list of top 5 or less keywords that are not common words (use words only from comment), semantic of the comment(positive / neutral / negative), is author is expert in risk analysis (positive / negative).

The output should be provided in next format:

i. [comment]

positive / negative * list of keywords=[keyword1, keyword2...] * semantic = positive / neutral / negative * objectivity = positive / negative * argumentative = positive / negative

A.4.2 Approach 2 – Generate analysis first and then asses features

The prompt is exactly the as in A.3.1 except for a different benchmark questions.

At first we prompt this **bechmark question**:

For each comment provide analysis that should comment how well argument of the author is constructed (does it correspond to risk type) and why assistant thinks so, comment the semantic of the comment is it positive, neutral or negative and why. Also assistant should comment if the comment is over pesimistic / optimistic or realistic about the risk type corresponding to comment (objectivity of comment). Also it should comment if the author is expert in risk analysis. Output should be provided in the next format: i. analysis: put analysis with 2–3 sentences (at most). Assistant shouldn't add any additional information that it is not asked. Provide answer according to output format. The language style of the output should be exact and not vague.

Then we use output of the LLM model to test against second **benchmark question**

Based on the provided analysis of comments. Find embedding for each analysis in the provided list (for all comments). Only semantic can take neutral value. Each analysis comment should be accessed next features with regards to associated risk type: objectivity of author(positive / negative) is person over (overoptimistic or overpesimistic which is "negative" or realistic which is "positive"), argumentative(positive / negative) how well comment corresponds to risk type, semantic of the comment(positive / neutral / negative), is author expert in risk analysis (positive / negative).

The output should be provided in next format:

i. expert = positive / negative * semantic=positive/neutral / negative * objectivity = positive / negative * argumentative = positive / negative

Note that assistant shouldn't access semantic of analysis, but semantic of the original comment based on analysis.

A.4.3 Testing quality of feature assessment with *logprobs*

You will be given a comment about given risk type and either impact or dimension of the risk. Return only answer to the question and nothing else. MAKE SURE your output is one of the options stated by question.

Comment: [...] Question: [...]

We refer to *text description* as mapping from numbers to text as suggested by Table 5. The comment is provided in the next format

The comment is about [Risk type...] and author discusses [Dimension (likelihood/impact)] of this risk. It graded [Dimension...] of the risk as [text description] and argumented it by saying: [Comment from dataset]

Each comment was accompanied with a question list:

questions = [

Is author is objective (which means that he is realistic or overpesimistic / overoptimistic). You can answer only with one word "true" or "false" or "maybe"

Is author comment is argumentative (which means that author thoroughly discusses the risk)? You can answer only with one word "true" or "false" or "maybe"

What is the semantic of comment. You can answer only one word "negative", "positive" or "neutral"

Is author is an expert in risk analysis and have enough expertise to judge about this risk. You can answer only with one word "true" or "false" or "maybe"

]

A.4.4 Check against hallucination by asking "is there is enough information"

You will be given a comment about given risk type and either impact or dimension of the risk. You will be given also a question about the comment. Before even answering the question, consider whether you have sufficient information to answer the question. Respond with just one word, the boolean true or false. You must output the word 'True', or the word 'False', nothing else. Respond 'True' if there is enough information and 'False' if it is not enough.

Comment: comment, Question: question

Comment and question are in the same format as in the previous part.

A.4.5 Check effect of hallucination on generation of analysis paragraphs

```
Assistant is an expert in risk analysis and linguistics, so based on the
comment it can evaluate parameters associated with text features. It can
make conclusions only based on the provided comment. The assistant would
be provided some information about the author and its opinion as comment.
Comment itself is about some risk type and discusses other impact or like-
lihood of risk type. Author comments how likely or impactful the risk type
and explains why he/she thinks so by giving additional comment. Return only
answer to the question and nothing else. MAKE SURE that the language style
of the assistant answer should be clear and consistent, it should contain
all the information asked in the question without unnecessary and vague words.

Comment: [...] Question: [...]
```

Comment is provided in the same format as in the previous prompt, but this time question is added as follows:

```
Provide analysis of the comment. The analysis should contain answers to
these questions:

- Is author objective (which means that he is realistic and not overpes-
imistic/overoptimistic)?

- Is author comment argumentative (which means that author throughtly dis-
cusses the risk)?

- What is the semantic of comment? Is it negative/positive or neutral

- Is author is an expert in risk analysis and have enough expertise to judge
about this risk?

Analysis should be through out and assistant should support the analysis
with proper reasoning: why assistant thinks so. Provide exactly 4 sen-
tences and each sentence should be related to only one question. The anal-
ysis should be based only on the comment. It should be a consistent para-
graph.
```

B Self assessment

B.1 How closely did the implementation of the project follow the initial plan?

There was a major change to the original project plan, namely the departure of the original project manager of the project around the end of March. This led to a lot of adjustments halfway through the project, especially to the schedule. Scheduling issues and a workload increase were expected consequences of a team member leaving. The scope and aims of the project remained mostly unchanged throughout the whole project though.

We compensated for the change in team composition by increasing the amount of weekly meetings. In the original plan, we had one meeting every week but after the departure of one team member, we realized that one meeting per week would not be enough and we increased the frequency to two meetings per week. Toward the end of the project, our meeting frequency increased to more than three meetings per week because we were behind on our schedule.

We originally determined two approaches with Inclus: qualitative-to-quantitative and quantitative-to-qualitative. These were refined after the presentation of the interim report. We ended up with two test cases from the qualitative-to-quantitative direction, these being the Outlier detection approach and the Feature extraction approach. The quantitative-to-qualitative direction was used in turn for finding highlights and key findings in the risk data and for providing summaries of the dataset.

Overall, the tasks that were determined at the beginning of the project covered what needed to be done during this project quite well. This meant that distributing the work was easier.

B.2 In what regard was the project successful?

We determined possible ways for Inclus to detect and reduce hallucinations through a literature review. Currently, there are no implementations for these processes at Inclus. We also made discoveries related to fitting use cases for Inclus through literature and importantly through our own testing. This way we were able to provide a summary of fitting uses for an LLM in the risk management context, which was one of our objectives.

We constructed and tested prompt engineering approaches with realistic use cases for Inclus. This included examining and testing the prompting for generating summaries, feature extraction, and outlier detection. We were able to assess the performance of the LLM in generating summaries,

This project was also a great learning experience for us. None of us had ever been part of this kind of real-world project where the problem itself is not just some theoretical question on a piece of paper but an actual problem that is asked to be solved by a real business organization. The real-world aspect meant that the question and the solution were not as confined as they usually are in regular school projects.

We also learned a lot about project planning since any of us had not been part of this kind of long-term group project. After completing this project each one of us would have the readiness to work as a team manager in this kind of project.

The AI industry is growing and understanding how LLMs work and can be optimized will likely be a desired skill when we graduate. All the things we have learned about AI modeling are on a generic level and further developments in LLMs will not decrease the value of our learnings. We all now understand a lot better how LLM's and AI's overall work and I think the skills that we acquired will be useful to us in the future.

B.3 In what regard was it less so?

A lot of our findings feel quite generic and hard to act on. Due to the tight schedule, we were not able to dive deep enough into the subject or perform a rigorous study, especially with our generous scoping of the topic.

Our ability to stay on schedule was very weak. The majority of the work on this project was completed last minute leaving no options for delays. Finishing the testing and implementations felt quite rushed in the end, and reporting our findings and conclusions with little time left meant a heavy workload and stress at the end of the course. We also did not take into account that the 4-period exam week and Wappu were all stacked up at the end of April / start of May which meant that we were not able to use that time period very efficiently.

We should have started to work with the prompting earlier since it was a lot more time-consuming than we first assumed. We also had some technical difficulties with the LLM server which caused some further delays for prompt engineering development.

B.4 What could have been done better, in hindsight?

In retrospect, we should have had more consistent meetings and updates with the client to keep a clear view of the scope of what we are doing and avoid getting distracted. We also think a more well-defined, focused scoping would have been

We should have made a schedule that was not based on getting a lot of tasks done during the last 3 weeks since that time period should have been reserved as a buffer in case some processes were not happening as fast as we wanted. In general, we should have worked harder to meet deadlines.

The course itself was arranged with a good format and schedule. The time between meetings was appropriate and it was nice to see the progress of other groups as well. It was great to see a diverse set of topics!