MASTER'S THESIS 2022

Quality Measurement of Generative Dialogue Models for Language Practice

Johan Bengtsson

ISSN 1650-2884
LU-CS-EX: 2022-34

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2022-34

Quality Measurement of Generative Dialogue Models for Language Practice

Kvalitetsmätning av Generativa Dialogmodeller för Språkträning

Johan Bengtsson

Quality Measurement of Generative Dialogue Models for Language Practice

Johan Bengtsson
jo6064be-s@student.lu.se

June 20, 2022

Master's thesis work carried out at
the Department of Computer Science, Lund University.

Supervisors: Markus Borg, markus.borg@cs.lth.se
Alexander Hagelborn, alexander.hagelborn@nordaxon.com

Examiner: Emelie Engström, emelie.engstrom@cs.lth.se

Abstract

In order to learn a language, it is essential to spend time speaking it. However, it can be hard to find someone to converse with. To solve this, NordAxon is currently developing a product, which aims to enable immigrants in Sweden to practice their conversation skills through a web page or the phone. The product is called Aida/Emely, which is based upon a subdomain of Natural Language Processing called Generative Dialogue Models (GDM), to pose as a highly available Swedish-speaking conversation partner. Basically, GDMs are Machine Learning models capable of producing humanlike answers given some input. Based upon Blenderbot, NordAxon trains several GDMs, and then need to perform a model selection between the models. However, it is unclear on what quality metrics to base such a model selection. In this thesis project, we aimed to address how to assist the Machine Learning engineers in the model selection process. This was done by conveying interviews and distributing a questionnaire to Swedish For Immigrants-professionals, as well as a literature review process to elicit requirements on the GDM. Based upon these sources, the most important quality metrics for such GDMs were prioritised and provided the basis for a test framework. We found that the GDM shall be coherent, non-toxic, and adjust the language level to the conversation partner. Furthermore, we interviewed the Machine Learning engineers at NordAxon to specify their requirements on the test framework. Based on the gathered information, we developed a test framework. The test framework generates thousands of conversations per GDM, which are then analysed. After the analyses are done, the test results are visualised in a Grafana dashboard. The results indicate that meaningful differences between GDMs could be detected, meaning that differences between the GDMs could be detected to help rank the GDMs. This may aid in the process of model selection. Regarding the different versions of Emely, the later versions seemed to be more coherent, less toxic, and have a slightly more frequent vocabulary, as well as having less variant readabilities. These findings suggest that NordAxon has succeeded in improving their GDMs.

Keywords: Generative Dialogue Models, Natural Language Processing, Quality Assurance, Transformers, BERT, Model Selection

Acknowledgements

Many people have been involved in this thesis project throughout this spring term. I would like to thank everyone for their contribution to this thesis project.

First and foremost, I would like to thank the supervisor at the Department of Computer Science: Markus Borg. Constantly throughout this thesis project, you have contributed with your knowledge, helping to bring clarity on what to do, how to do it, and when to do it. Although this thesis project has not been trivial, it has certainly become easier with your help, which I am very grateful for.

Secondly, I would like to thank the supervisor at NordAxon, Alexander Hagelborn. Throughout this spring, you have brought your knowledge and experience into this thesis project. The code of this test framework, as well as my Python skills, have been improved by letting you review all the pull requests and contribute with your thoughts on the code. Besides that, you have also assisted me in the understanding of the field of Natural Language Processing, providing good thoughts on how to tackle different tasks within the test framework.

Thirdly, I would like to thank NordAxon for letting me work at their office and giving me access to their computer. It has been my pleasure to work next to you at your office, spending time with you on the breaks and drinking coffee with you. This term has given me many laughs as well as a good preview of how work life may look like.

Lastly, I would like to thank all friends and my family who have supported me and this thesis project in one way or another. You have helped me achieve the results that I achieved, for which I am very grateful.

Lund, June 2022.

Contents

Chapter 1
Introduction

In this chapter, we present the introduction to this thesis project. Firstly, we present the problem background, which is the problem and preconditions that we encountered when starting this thesis project. Secondly, we present the scientific contributions. In a brief way, we here present the contributions from this thesis project. Thirdly, we present some related works, to guide the reader in how related works have been working with this matter, and how this thesis project differs from those. Lastly, we present the thesis structure, with the purpose of aiding the reader in the reading of this thesis project.

1.1 Problem Background

NordAxon is a company based in Malmö that operates within the field of Applied Data Intelligence, where they combine AI with data, analytics, and automation to help improve businesses. They offer both training within these subjects as well as products and consulting.

NordAxon is currently working on the development of a product — a conversational agent called Aida/Emely (hereafter referred to as Emely) – based upon Generative Dialogue Models (hereafter referred to as GDM). The aim for the product is that it should be used for Swedish language practice[1]. Language learning is a non-trivial task, requiring plenty of hours of practice in order to master a language. Even more so, for practicing the conversation skills, another person is also required as a conversation partner, but which is not always easy to find. NordAxon aims to aid the task of language learning with Emely, which should be a conversation partner with whom the user may practice their conversation skills.

Emely uses an internal GDM, which is a subdomain of Machine Learning (ML) where models are trained to take any text as input and respond with another text, with the aim of making humanlike answers. GDMs are normally trained using deep neural networks. And usually when neural networks are used for training models, model selection, i.e. the task

[1] https://www.svt.se/nyheter/lokalt/helsingborg/ai-kan-forbattra-sfi-undervisningen-i-helsingborg

of selecting the most suitable model from a set of candidate models, is a non-trivial part. Typically, the ML engineer sets up one or several metrics used for evaluating the performance of the candidate models, which are then basis for the selection of the best model. E.g. models used for classification of numbers in images could have a metric of accuracy, namely how good the model is at making correct predictions. For regression tasks Mean Squared Error could be used as a metric measuring how well the model performs [13]. However, there does not seem to be any established metrics today for evaluating GDMs that perform well. Since these conversational agents aim to make humanlike answers, several measures have been proposed to assess to what extent textual output from a GDM is humanlike.

During the summer of 2021, the author spent eight weeks in a research project with this very research topic and laid the ground for a framework that executes thousands of conversations with the GDM under test, during which test cases are injected randomly to test specific humanlike conversation abilities. After all the conversations were done, some further analyses were performed, e.g. analysing the toxicity levels of the responses and summarising the results of the test cases. In total, conversational abilities such as Coherent responses, Consistency, Memory, etc. were set as the quality attributes and they were all measured in different ways. Building on the exploratory work from last summer, presented at a scientific conference on AI engineering [10], this thesis project seeked to develop a systematic and automatised approach for the quality assurance of GDMs, with the goal being to support the ML engineers at NordAxon with the model selection of GDMs taking part of the development process.

1.2 Scientific Contributions

This thesis project aimed to investigate how to perform automated quality assurance to support model selection of GDMs in an industrial context. It came down to addressing the following research question:

RQ How can automated quality assurance support model selection of GDMs in the NordAxon context?

In order to address this research question, the project was initiated with a literature review process. From this, we specified requirements that could be relevant with regards to a GDM. Those requirements are presented in table 4.1.

Then, an information gathering process was initiated. From the interviews with SFI professionals, here are the most relevant insights:

- Learners need a fair and reasonable challenge – They emphasised that it is important for every learner to have a fair challenge in front of them in order for them to learn. Without a challenge, the learner will not learn anything new. However, with a too large challenge, instead the learner risks losing the motivation.
- Big range of skill levels – Learners of Swedish as a second language range from people who are illiterate and who have not studied Swedish at all, to those that have spent many years studying Swedish. This means that there is a huge skill-gap between the learners. With this in mind, along with the previous point, SFI teachers explained that the teaching of Swedish needs to be adjusted to the skill level of the learner, as to provide every learner with a fair challenge.

- Readability indices and word frequency lists – Those are two concepts that the SFI teachers expected to be useful when assessing the language level of the GDM.

The interviews combined with the questionnaires to the SFI professionals helped us prioritise amongst the requirements. We found the three most important requirements to be:

REQ3 The GDM shall have a fair-levelled vocabulary

REQ4 The GDM shall produce coherent responses with regards to the last response

REQ7 The GDM shall use a non-toxic language

Furthermore, interviews with NordAxon were conducted to understand their requirements on the test framework. Those interviews resulted in an additional set of requirements that were specified as user stories in table 4.3.

Based on these requirements, a test framework was developed which is capable of producing thousands of conversations per tested GDM, and then assessing the levels of coherence, and toxicities. It is also capable of analysing the language level of the GDM by calculating readability indices and storing word frequency ranks. The framework then exports these results into a SQLite-file. Further on, a dashboard in Grafana was implemented, which reads the results from the SQLite-file and then visualises those. More specifically, it presents average, median, variance, percentiles, maxima, minima, and histograms per GDM allowing for comparisons.

To evaluate the performance of the test framework, we executed it on several different GDMs. Each of those GDMs produced 2,000 dialogues, where each of the dialogues contained 20 responses from the tested GDM. Then, we looked for meaningful differences in the dashboard between the GDMs. That is, we visualised the results in the dashboard and compared the GDMs' average toxicities, average coherences, standard deviations etc. to find differences suggesting that one GDM outperformed the rest.

To summarise the project, through an information gathering process, we gathered relevant requirements. Based on a prioritised subset of the requirements, a test framework was developed. The test framework produces thousands of conversations, applies test cases regarding toxicity, coherence, and language level assessments. The results are exported and then visualised in a dashboard. The test framework was evaluated by comparing the results between the GDMs. We concluded that the test framework revealed meaningful differences between GDMs, which suggests that it can be used to support model selection at NordAxon.

1.3 Related Work

In this section, we present some of the previous works. We present a short summary of their reports and findings, and how the findings of this thesis project differ from theirs.

"BERTScore: Evaluating Text Generation with BERT" by Zhang et al. [33]. The authors of this study proposed BERTScore, an automatic evaluation metric that can be used for computing similarity scores between two different sentences. They compute token similarity using contextual embeddings, rather than exact matches, which makes BERTScore more robust than other existing measures for sentence similarity. This report presented a publicly available tool for evaluating text generation. However, it seems to rely on having references

to compare with. Instead, this thesis project resulted in a reference-free framework. That is, for every evaluated text, there is no reference being compared with. Rather, all evaluated responses are generated during the run of the test framework.

Chatbottest (`https://chatbottest.com/`) is an existing framework to manually assess the quality of a chatbot. Chatbots are typically not based upon ML, but rather rely on hard-coded responses to certain kinds of input. For assessing the quality of chatbots, the creators have identified 7 categories of abilities that are relevant when measuring the quality. Those categories are:

- Personality – Does the chatbot have a clear voice and tone that fits with the current conversation?
- Onboarding – Is it clear for users what purpose the chatbot fulfills and how to use it?
- Understanding – How much does the chatbot understand when it comes to requests, small-talk, idioms etc.
- Answering – Is the chatbot able to answer in proper ways and answering the correct question?
- Navigation – Is the conversation flowing without friction? Does the user get lost?
- Error management – How good is the chatbot to deal with errors that may occur?
- Intelligence – How intelligent is the chatbot? Does it remember things?

Within these categories they have proposed several questions that any chatbot tester may use when examining a chatbot. Using these questions, the chatbot tester may receive indications on what parts that are the most problematic for the chatbot. Their work require a human to assess a GDM, whereas the test framework of this thesis project evaluates the GDM in an automated way.

See et al. [26] examined different quality metrics, and their correlation with human assessment. They proposed several controllable attributes of GDMs, which they then changed to assess how they correlate with human judgement. Those attributes were attributes such as the repetition, the specificity of answers, response-relatedness, question-asking balance, interestingness, making sense, fluency, listening, inquisitiveness, humanness, and lastly engagingness. The focus of their work was to change controllable attributes of the GDM and see how the change affects a human judgement. This thesis project differs from their project in a way such that this thesis project automatically assesses the quality of a GDM without any human intervention, basically.

Deng et al. published "Compression, Transduction, and Creation: A Unified Framework for Evaluating Natural Language Generation" [11]. The authors of the study mentioned that text processing tasks can be divided into three broad categories, namely:

1. Compression – Text of information being compressed whilst preserving the meaning.
2. Transduction – Text of information being paraphrased, either within the same language or translated to another language.
3. Creation – Text of information being created from scratch.

Regarding 1) Compression, they propose consistency and relevance to be relevant subparts of compression that needs to be assessed in order to assess the grand task of compression. Secondly, to assess 2) Transduction, they propose that one needs to assess preservation, i.e. how much of the information is preserved. Lastly, for 3) Creation, they propose the need of assessing engagingness, to create answers that engage the respondent, and groundedness, how well a sentence refers to the knowledge context. They base their metrics on the concept of information alignment between the text before and after the task has been done. And to operationalise their findings, they have trained self-supervised models to approximate information alignment as a prediction task. Lastly, their results indicated that their metrics achieve stronger or comparable correlations with human judgements compared to other state-of-the-art metrics. Their findings regarding 3. Creation is what primarily resembled this thesis project, where they assessed the quality of generated text. However, they focused on engagingness and groundedness, whereas we focused on coherence, toxicity, and assessing the level of the language.

Thoppilan et al. published "LaMDA: Language Models for Dialog Applications" [28], in which they present their work on how to improve the performance of LaMDA. LaMDA is a GDM, which consists of a family of Transformer-based language models with the primary specialisation being dialogues[2]. LaMDA has up to 137B parameters and is pre-trained on a number of 1.56T words. In the paper, they assessed the performance of LaMDA by observing metrics such as quality, safety, and factual grounding. They assert that model scaling alone does improve quality in general, but does not benefit safety and factual grounding as much. They define safety as a metric indicating to what extent the model's responses are consistent with a set of human values. These values consist of the likeliness to make harmful suggestions, and if any unfair bias is present. Factual grounding as another metric is quantified by using a groundedness metric, which really is that whenever the GDMs responds with facts, its correctness is checked whether it is true or not. They primarily studied how to improve the quality of one GDM, whose quality was assessed by humans, whereas the test framework of this thesis project assesses the quality automatically.

Guo et al. [14] proposed an approach where topic-based metrics should be used to evaluate dialogue quality. Using a topic classifier, they proposed metrics such as topic depth and topic breadth, that could be of use to assess the quality of a GDM. They evaluated their results using data collected from the Alexa Prize competition. Unlike them, we did not use a topic classifier as we did not find it to be amongst the most important attributes for Emely. Neither did we use publicly available data to assess the quality, but rather we let the test framework use the GDMs to generate data from scratch. The generated data is then assessed using open-source tools, formulas and NLP concepts to analyse the chosen requirements.

Mehri and Eskenazi [20] introduced a Fine-grained Evaluation of Dialog (FED) metric, which is an automatic evaluation metric based upon DialoGPT, a pre-trained transformer GDM that is based upon 147 million conversation-like messages from Reddit [34]. They set up metrics that are both on the turn-level (an analysis between two responses), and on the dialogue-level (an analysis of the whole conversation). They proposed 18 parameters/metrics that together pose their fine-grained metrics. Their turn-level metrics were:

- Interestingness
- Engagingness

[2]In this thesis project, the words "conversation" and "dialogue" are synonymous.

- Specificity
- Relevancy
- Correctness
- Semantical appropriateness
- Understandability of the produced response
- Fluency

For the dialogue-level metrics, they proposed:

- Coherence
- Ability to recover from errors
- Consistency in information
- Diversity in responses
- Topic depth
- Likeable personality
- Understandability of the input
- Flexilibity/adaptability to the user
- Informativeness
- Inquisitive, showing interest in the user
- Overall impression

They proposed several quality metrics proposed to the reader, and they are keeping their tool open-source. Their tool analyses one or several strings, and returns the scores, whereas our test framework generates conversations, analyses the conversations, and then presents the results. Hence, their tool cannot solely support model selection, as it needs data and some way of presenting the results, all in an automated fashion. Nevertheless, it could be integrated as a test case into our test framework.

Mehri and Eskenazi [21] asserted that standard language generation metrics have been ineffective when it comes to assessing GDMs. Therefore, they did propose USR: an UnSupervised and Reference-free evaluation metric. The USR is a reference-free metric used for training unsupervised models to assessing different quality attributes of dialogues. They also showed in the report that the metric has strong correlation with human judgement on some specific annotated datasets, namely PersonaChat and Topical-Chat. The quality attributes that USR is based upon are:

- Understandability

- Natural
- Maintains context
- Uses knowledge
- Overall quality

As for their previous work, they present a tool to assess quality, but unlike our test framework it needs data as well as some way of presenting the test results in order for it to aid in the model selection. As before, this tool could also be integrated into this test framework as a test case.

Our previous work proposed a test framework that set up a dialogue between two GDMs [10]. By executing thousands of dialogues, we generated enough text data for assessing the quality of the GDM under test. After a requirements elicitation process, we gathered 37 requirements for the Emely GDM. Those requirements were then subject for a prioritisation process, where we prioritised based on a cost-value procedure. The result was a set of 15 particularly interesting requirements, on which test cases were designed and implemented. Two different kinds of tests were used:

- Injected tests, with a Question-Answering structure where the tested GDM is asked for some information, and the reply is assessed according to some ground truth.
- Static tests, which were different static analyses where the dialogues were statically assessed, meaning that existing frameworks and techniques were applied to all the dialogues to assess the metrics per dialogue.

We showed that 6 out of the 15 implemented tests did reveal meaningful differences between candidate GDMs. In this thesis project, several parts of the open-source infrastructure were used, e.g. how conversations are generated, how coherence is measured etc. However, this thesis project differs from our previous work in a way such that we have now focused on the ML engineers and how they want to integrate the test framework into their pipeline. Also, this thesis project exports the results into a database file, so that the results may be visualised in any visualisation tool capable of interpreting SQLite.

1.4 Thesis Structure

In this section, we present the structure of this thesis. It has the purpose of giving the reader a better overview of the thesis, and to guide the reader in the process of reading.

2. Background The Background consists of the foundational information needed to understand this thesis project. It introduces important concepts used in the remainder of the thesis. On top of the important concepts, a description of the product under test is presented as well.

3. Research Approach In the Research Approach chapter, we present the full work process from the start until the end. Firstly, the methodology used throughout the whole thesis project is described. Then, three different phases for the project are presented, in an order corresponding to the order for when the different phases started. That is, the different phases were not sequential, but rather overlapped.

The three phases were Information gathering, Development phase, and Evaluate results. The work process for the information gathering phase is here described, introducing how the information was gathered and managed. Then, based upon the gathered information, the development phase is described to the reader. Lastly presented is how the results of the test framework were evaluated.

4. Results In this chapter, we present the contributions per contribution type according to the Design Science Methodology. That is, the Problem Conceptualisation, the Solution Design Proposal, the Solution Instance, and the Evaluation are all presented.

5. Discussion In the Discussion chapter, we discuss the different contributions per contribution type. Per contribution type, some interesting points to notice and to discuss are discussed. After those contributions, we present and discuss some threats to validity as well as some possible future work.

6. Conclusions In the Conclusion chapter, we summarise and present the findings of this thesis project in a clear and concise manner. Moreover, we end the Conclusions chapter by answering the research question.

Appendix In the appendix, we present the results from the questionnaires that were sent out to the SFI professionals, which were part of the requirement elicitation process. That is, figures are shown that present how the SFI professionals responded to the questions of the questionnaire.

Chapter 2
Background

In this chapter, we present the necessary background for understanding the problem domain. Moreover, we also present the overall architecture of Emely.

2.1 Natural Language Processing

Natural Language Processing (NLP) is the domain in which computer programs aim to deal with natural language. That is, it consists of tasks such as analysing, understanding, and generating natural language [22]. More specifically, such tasks could be the following to name a few:

- Generating whole new texts
- Finding the answers to questions about a text
- Identifying the topics of texts
- Identifying names and places within text
- Summarising texts

NLP consists of the algorithms used for training models capable of performing these varying tasks [22]. This thesis primarily handles the first one "Generating whole new texts", since the topic of this thesis is about quality measurement of GDMs, models capable of generating new texts.

Word frequency list A concept within Natural Language Processing is word frequency lists. A word frequency list is a word list where specific words are mapped to their specific rank, a rank which is based upon how frequently used the word is within the language. That is, within a given corpus or set of data, every word's total occurrence is counted.

Then, a sorted list is set up, which has sorted the words' frequencies in a descending order. The result is a word frequency list, where the first word is the most common word, and the last word is the least common word. Since a higher ranked word is more common to encounter within the language, the higher ranked word is both more useful and easier to learn compared to a lower ranked word [31] [15].

2.2 Readability

Readability is defined as how easy it is for a reader to read and then understand a written text. This measure is something that may vary a lot depending on several different factors, e.g. the complexity of the vocabulary, its familiarity, typography etc. As an example, readability benefits the user experience when visiting websites. E.g. if content is hard to grasp, the user experience will be worse compared to if the content is easier to grasp. Furthermore, readability scores have been created in an attempt to provide a measure to tell what level of education a person reading a text needs to have in order to read and understand the text easily. Typically, these scores calculate a score based on different factors, such as sentence length, syllable density, word familiarity [1].

One form of a readability score is the Läsbarhetsindex (LIX), which is Swedish for Readability Index. It is calculated by first counting the number of sentences (S), the number of words (w), and the number of words larger than 6 letters (W) within a text. Then a readability index is calculated according to the following formula [6]:

$$\text{ReadabInd} = \frac{w}{S} + \frac{W}{w} * 100$$

2.3 Generative Dialogue Models

In this section, relevant concepts within the field of GDMs are presented.

Transformer The transformer architecture was first introduced in 2017 by Vaswani et al. [29]. Prior to the transformer, complex recurrent and convolutional neural networks were the architectures dominating amongst the sequence transduction models, i.e. models taking input sequences and transforming it into output sequences. However, with the introduction of the transformer architecture, this changed. The transformer architecture offered a simple network architecture, yet with unparalleled capabilities. Those capabilities compared to recurrent neural networks were that the transformer did not need to process the sequences in order, giving the user more parallelisation during training. More specifically, this enabled the user to use larger datasets for training the model, with the results being models with superior results compared to the predecessors.

BERT In 2018, Devlin et al. [12] at Google introduced the Bidirectional Encoder Representations from Transformers (BERT). It is an extension of the transformer, with the capability to represent words based on both the preceding words as well as the upcoming ones, hence the name "bidirectional". To reach these results, BERT is pre-trained using two unsupervised tasks: Masked LM (MLM), and Next Sentence Prediction (NSP).

MLM means that some percentage of the input to the model is masked, upon which the masked tokens are then predicted. Devlin et al. [12] let 15% of the tokens be masked, which then were to be predicted. This strategy allowed them to obtain the bidirectional pre-trained model.

Secondly, for each pre-training example inputted to the model, 50% of the time it is the correct example, labeled as `IsNext`, and during the other 50% of the time it is an incorrect example, a random sample from the corpus labeled as `NotNext`. By doing so, the model was also trained to better understand the relationships between words. This task is called Next Sentence Prediction. Furthermore, this results in capabilities of the model beneficial to some general NLP tasks, such as Question-Answering (QA) tasks, and Natural Language Inference (NLI). [12]

NSP-BERT A technique that was originally used by Devlin et al. [12] was later on released as NSP-BERT by Sun et al. [27] as an open-source tool. The tool is capable of predicting the probability that one text succeeds another text, that is, the probability that one texts directly follows another. More specifically, given the two texts `text1` and `text2` as input, NSP-BERT would return two predictions: the positive probability, meaning the prediction that `text2` does succeed `text1`, and the negative probability, meaning the prediction that `text2` does not succeed `text1`. This is further demonstrated in figure 2.1. Note that the sum of the positive and negative probabilities equals 1.

Blenderbot Roller et al. [24] presented their work, posing as a recipe for building open-domain GDMs. In this work, they presented several GDMs of different sizes (the number of parameters used for the model), which they gave the name Blenderbot. Blenderbot is a GDM extending the transformer architecture, and its different variants of different sizes can be found [2] as open-source on Huggingface, an AI community where you may find different GDMs [5].

2.4 Overall Description of Emely

NordAxon is currently developing a product based upon a GDM that aims to provide Swedish learners and teachers with a tool primarily for enabling Swedish learners to practice conversing Swedish. That is, the product should enable the user to practice performing a conversation, which involves both producing sentences orally whilst listening and interpreting what the conversation partner says. As of now, two different profiles of the product are being produced, being an Interviewer and a Fikakompis (Swedish for coffee-buddy, meaning friends/colleagues chatting in an informal setting). The Interviewer has the purpose of enabling the Swedish learner to practice both Swedish as well as on a job interview. Regarding the Fikakompis, it has the purpose of enabling the Swedish learner to practice Swedish in a more informal setting.

In order to fulfill these purposes, NordAxon has based their product upon the high-leveled architecture that can be seen in figure 2.2. In the upper half of figure 2.2, the process of input from the user being inputted to the GDM is visualised. Then, in the lower half of the same figure, the path from the GDM producing a response to reaching the user is visualised.

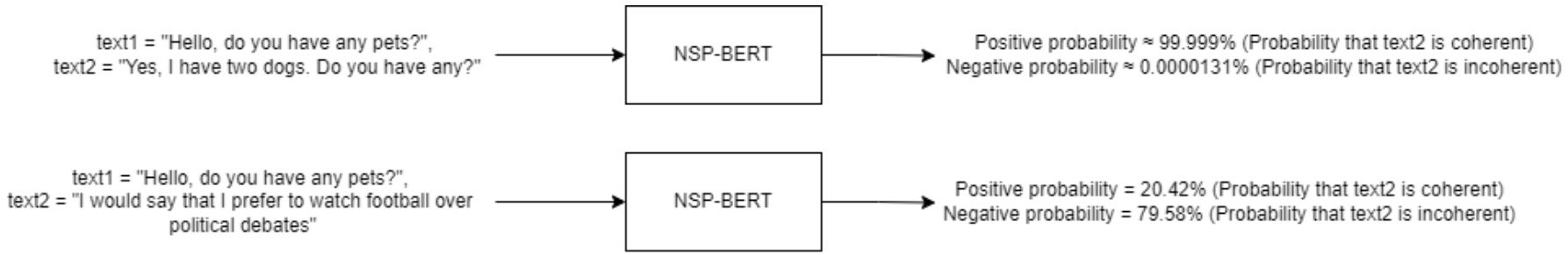

Figure 2.1: The functionality of NSP-BERT.

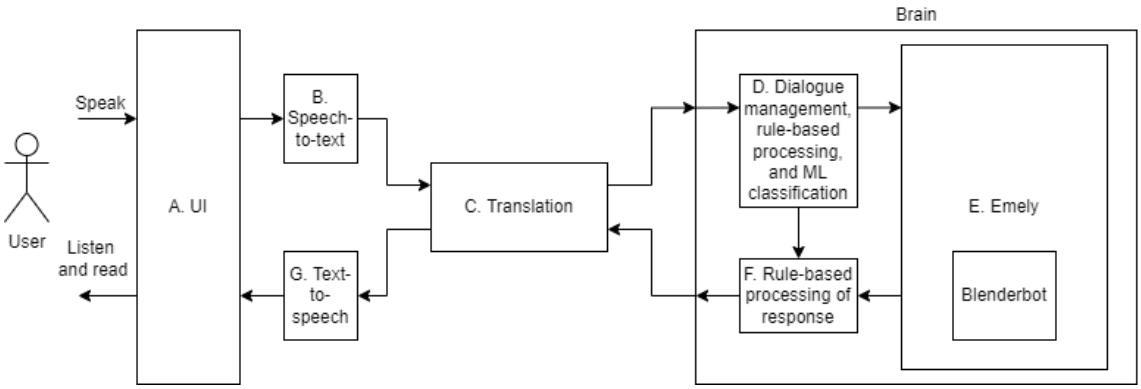

Figure 2.2: The architecture of the product on a high-level.

The product has a UI with which the user interacts, which can be seen at (A). The user speaks Swedish to the UI, which then at (B) uses a Speech-to-text third party component for translating the Swedish speech into text. After the speech has been interpreted into Swedish text, at (C) it is translated from Swedish to English using another third party component. This is done as the most tools within the community of GDMs and NLP are only available for the English language. When the text has been translated into English, it is then inputted into the Brain. It is the Brain that then interprets the input and produces a response, similar to how a real person would respond to the sentence of another person. Within the Brain, the input is first inputted into (D), where the message is inputted into a layer relying on ML classifiers to handle certain kinds of input that is known to be problematic. Specific scenarios that are known to be problematic for the GDM can here be addressed, implicating that high-quality responses are produced even for those scenarios. Such scenarios could be if the user asks for the salary offered for the actual job position, for the Interview-Emely. Nonetheless, if the Dialogue manager does not detect any such scenarios, the text is inputted into the GDM, at (E). Given the input, Emely - based upon Blenderbot - produces a response. It is the component called (E) that is the component being subject to the tests of this thesis project.

Given the produced response from Emely, some rules are applied at (F) to the response. For instance, the response is checked for whether it contains any toxic content. The purpose of (F) is to ensure that no unacceptable content is produced and presented to the user. Moreover, at (C) the response is then translated from English to Swedish, prior to being transformed into speech again at (G). Finally, the produced response is presented to the user both by being spoken as well as being presented in a text format, in order for the user to be able to listen as well as read the response.

Chapter 3
Research Approach

In this chapter, we describe the methodology of this thesis. During the setup of the project, we identified and planned for three phases. They were not planned as being chronologically distinct phases, but rather concurrent. Those were:

1. Information gathering
2. Development process
3. Evaluation

Moreover, in this chapter we present the development methodology in a general way, then we give a description about the information gathering process. Thirdly, we describe the development process further into detail. Lastly, we present the process for evaluating the results of the test framework and the presentation dashboard.

3.1 Design Science Research

In order to create a test framework and dashboard that should be of value for NordAxon, the work process of this thesis was inspired by the Design Science Methodology [25]. The work process of the Design Science Methodology is presented in figure 3.1.

Regarding the types of contributions from Design Science Research, they could briefly be visualised in 3.2. In this report, we present contributions to each of the four contributions boxes: Problem constructs, Design constructs, Solution instances, and Problem instances. We describe here the methodology leading to the contributions of this thesis project per contribution type.

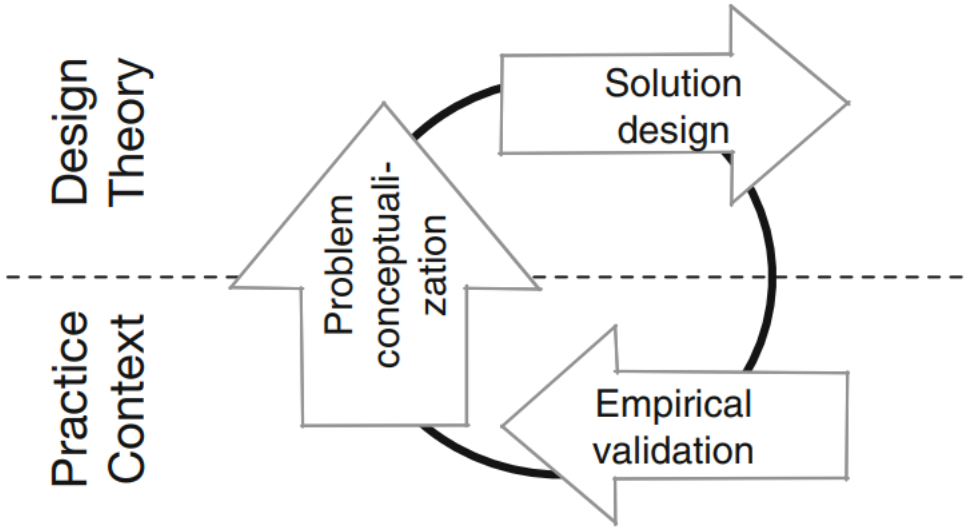

Figure 3.1: The research activities that take place in the Design Science Methodology

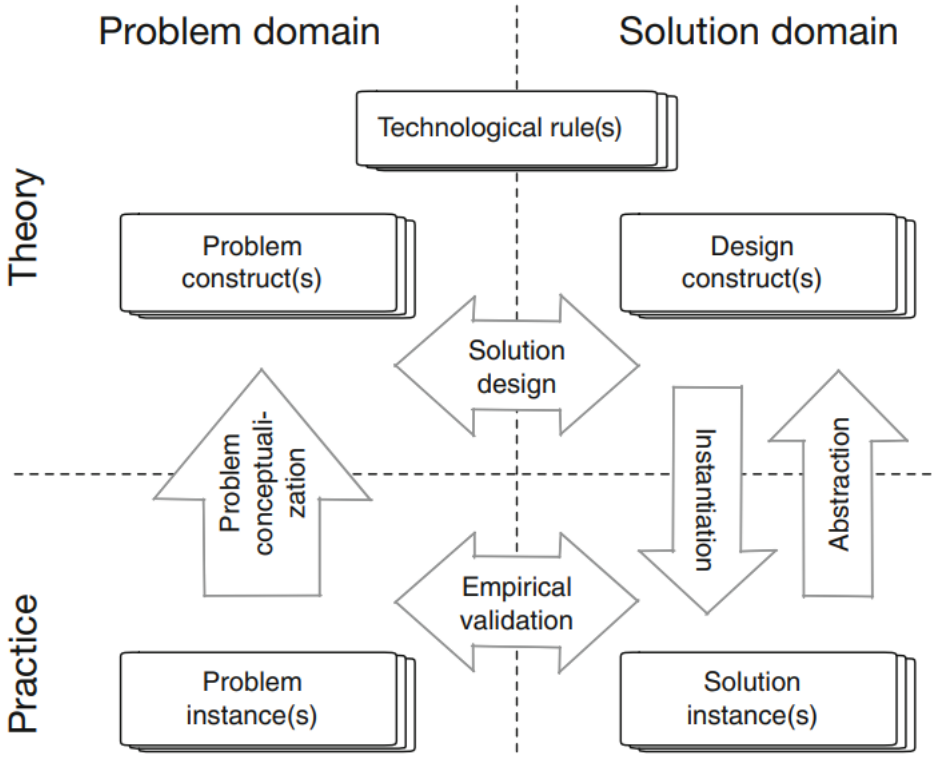

Figure 3.2: An overview of the Design Science Methodology contributions, where the boxes show theoretical and practical contributions, and the arrows show knowledge creating activities.

Problem constructs To understand and conceptualise the problem, it is of interest to observe the problem. More specifically, that is to observe the practitioners in action while they are encountering the problem, or to discuss the problem with them in order to gain understanding.

Design constructs Based on a conceptualised problem, a solution designed can be forged. That is, based on the understanding of the problem, along with eventual complementary information, a solution can be created that could solve the problem.

Solution instances Based on a solution design, a tool is implemented, whose purpose is to solve or handle a problem. The tool can then be used to evaluate whether the solution suffices to solve or handle the problem.

Problem instances The occurrence of the problem subject to being solved or studied. Based on this occurrence, the problem can be conceptualised. Also, the solution instance may be applied onto the problem instance, in order to evaluate the performance of it.

3.2 Information Gathering

We divided the information gathering phase of this project mainly into two parts: 1) a literature review and 2) requirements elicitation with internal and external stakeholders through interviews and a questionnaire-based survey. We describe these two parts of the phase in this section.

3.2.1 Literature Review

We initiated this phase by conducting a literature review. That is, using LUBSearch and Google Scholar, the current research within the domain was read. The purpose of this process was to understand what had already been done, how other researchers handled different kinds of quality measurements of GDMs, what existing theories/frameworks could be applied in this work, and to better understand the problem domain and the knowledge field. Through searches on keywords such as NLP, Generative Dialogue Models, Quality Measurement NLP etc., we found relevant publications. Then additional reports were found by looking into the related work of those reports, i.e. a backward snowballing search strategy [32]. We assessed plenty of reports for relevance to the project at hand, and the pieces of work closely related to this thesis project are presented in section 1.3.

3.2.2 Requirements Elicitation

After the literature review process, we deemed it relevant to initiate communication with the stakeholders that were relevant for this project. The obvious stakeholder was NordAxon, but other relevant external stakeholders include the future users of the test framework. Namely, those were found to be Swedish immigrants, and professionals involved in teaching Swedish as a second language – Swedish For Immigrants-teachers (SFI teachers). We also found SFI

researchers to be of interest. Due to time constraints, we prioritised interviews with SFI teachers over immigrants. Nevertheless, we highlight that interviews with Swedish immigrants would be an important direction for future work.

Internal stakeholder interviews with NordAxon As the test framework had the purpose of assisting NordAxon within their model selection process, it was vital to have an ongoing communication throughout the project. This was achieved by working at their office and having at least one meeting per week. However, in the early phases of this project, it was especially interesting to grasp their requirements on the test framework. Through a meeting with their lead ML engineer, we elicited requirements on the test framework. Those requirements were specified with regards to the ML engineers, with the purpose of understanding their requirements and how the test framework better could fit into their pipeline for developing GDMs. Later, we presented the requirements to NordAxon, with the purpose of validating the requirements to ensure that they were meeting NordAxon's needs.

Requirements elicitation with SFI professionals Prior to initiating the communication with experts, we clarified and defined the purpose of the GDM, as to better understand what information was needed from the communication. The purpose of the GDM was defined as ...:

- ... it shall be able to help the learner learn the specific language
- ... it should be fun/interesting enough to talk to, as to motivate the learner to speak the language on a regular basis during a longer period of time

Furthermore, we divided the communication with the SFI professionals into two parts: 1. Virtual meetings discussing open-ended questions regarding Swedish as a second-language, and 2. Questionnaires, where their feedback on the requirements found in table 4.1 was gathered. During the virtual meetings, the focus was on giving the interviewees the opportunity to present their thoughts without too much intervention, but rather just commenting their thoughts when necessary. The interviews were guided by the following questions:

- What do you think is important for the characteristics of the Swedish language being used when speaking to Swedish immigrants?
- What is specifically important for the characteristics of the Swedish language to support effective second language acquisition?
- How can you as a conversationalist be more engaging and motivating towards the immigrant?
- What are the typical errors learners of Swedish make?
- Would it be important for the learner to stick to one conversation partner, or to change partner on a regular basis?
- How important is the accent that the conversation partner has?
- How can you assess the level of any given sentence by analysing it in retrospect?

- Is there anything else you would like to mention?

The goal of communicating with SFI professionals was to gain insights into which requirements to prioritise for the test framework, and which ones that could be initially disregarded. We recruited the SFI professionals as participants in the interviews by manually looking up the email addresses of different SFI-schools. We then sent emails to those schools, asking for access to their teachers. Furthermore, we used convenience sampling [9] to invite researchers at Malmö University studying acquisition of Swedish as a second language. The virtual meetings were informal and ample notes were collected.

After the meetings, we distributed a Google Forms questionnaire to the interviewees. To further gather responses to the questionnaire, we requested selected regional SFI schools to distribute the questionnaire among their teachers in February 2022. The questions of the questionnaire contained the requirements that were gathered from the literature review process, which is specified in table 4.1. We formulated the questions in such a way that the respondent of the questionnaire should specify using a Likert scale [17] [19] the significance of every requirement with regards to the role as a conversation partner to a second-language learner. I.e. the respondent specified using a 1-5 scale the significance of every requirement, where 1 indicates that it is particularly unimportant and 5 indicates that it is particularly important.

3.3 Development Process

In this section, we present how the development process of the project was performed. That is, first we present the work process behind the creation of the solution design. Then, we present the work process behind the creation of the test framework.

3.3.1 Initial Test Framework Architecture

This sub-phase, we initiated by planning the architecture and flow of the test framework. We wanted to produce a figure corresponding to the process view of the 4+1 view model of architecture [18]. For this, we were inspired by our previous work [10], where the principle was to let two GDMs converse with each other, producing conversational data onto which test cases could be applied. Then, the plan was for those test results to be visualised in some visualisation tool.

Moreover, inspired by the principles of object-oriented programming, it was also deemed relevant to create a class diagram to grasp the initial needs of classes for the framework. For this, we wanted to produce a figure corresponding to the logical view of the solution according to the 4+1 view model of architecture [18]. This view, along with the earlier mentioned process view, were supposed to provide assistance in the start-phase of the development to visualise and better understand what was needed to be developed.

We set the architectural goal to develop a flexible, modular, and scalable test framework. That is, we strived towards a modular architecture so that parts within the framework could easily be improved, replaced with another part, or that parts easily could be inserted into the flow. E.g. that GDMs could easily be added to the framework, or that more test cases easily could be integrated. Or even that the results could be visualised through any kind

of external tool. As for the scalability, the aim was to implement a framework that easily could scale up, as to be able to reach actionable insights, without extending the time of the runs to unreasonable times. As a comparison, when the ML engineers at NordAxon train models, such trainings typically last for 30-60 minutes, but that durations of 150 minutes have also occurred. The metrics used for other kinds of ML tasks, e.g. accuracy, are calculated and presented directly after training. Therefore, we realised that an optimal test framework would produce actionable insights instantly. However, we understood that it was not realistic to reach given the time-consuming nature of GDMs combined with the idea of producing loads of conversations. Thus, the goal was for the execution time of the test framework to not exceed the training times by more than a factor 100 – corresponding roughly to running the test suite over a weekend, which we believed could be integrated into the pipeline and to be of value to NordAxon.

3.3.2 Implementation

We initiated the development guided by the plan we had set up, according to section 3.3.1. The test framework was implemented in Python, and by working from NordAxon's office, support within ML and Python was close at hand. That combined with weekly meetings with both the supervisors shortened the feedback loops, assuring close communication. Even more so, we uploaded the framework to a GitHub repository, where the industry supervisor reviewed each pull request, i.e. constantly following the development, and highlighting any potential issues early. This also assured that best practice could be applied during the development. Ultimately, it would also enable them to better understand how the framework would be structured, and continuously take part in the design process.

3.4 Evaluation

To evaluate the test framework, we chose to run the test framework on the Emely models at hand, but to also include the Blenderbots into the test as to have some kind of baseline on how open-source GDMs perform. For the evaluation of this thesis project, the computer used is specified in table 3.1.

Table 3.1: The specifications used for evaluating the results.

CPU	Intel(R) Core(TM) i9-7900X CPU @ 3.30GHz
GPU	Nvidia RTX 3090 Ti 24GB
RAM	32GB

We evaluated the results by comparing the different metrics that we have set up. E.g. which GDMs perform best on average on some metric, or which GDMs are performing the same on average but are less variant. Here we wanted to see meaningful differences between the GDMs. That is, suppose that Emely v05 has a better average in a test compared to Emely v02. Then that is a meaningful difference such that the difference in the average means that Emely v05 is a better version and should be chosen over Emely v02 when taking that test case into consideration.

Chapter 4

Results

In this chapter, we present the results per type of contribution according to the Design Science Methodology. That is, we present the contributions per each of the four types Problem Conceptualisation, Solution Design Proposal, Solution Instance, and Evaluation.

4.1 Problem Conceptualisation

When ML models are trained, the ML engineer typically relies on some metric to assess the quality of different models, and then to perform the model selection. E.g. suppose that a ML engineer wants to create a model capable of predicting whether there is a dog, a cat or a human in a picture. Then, the ML engineer could train several models to perform this task. After training the models, the ML engineer could choose to select the model achieving the highest accuracy for instance. That is, the model capable of making the most correct classifications on a given set of pictures. But when it comes to the field of GDMs, there are no such metrics clearly capturing the quality level of a GDM.

As earlier mentioned, a person needs to speak a language during a period of time to learn it. This, however, requires someone to speak with – a conversation partner. It is this person that the GDM needs to be able to replace. This can be visualised in figure 4.1.

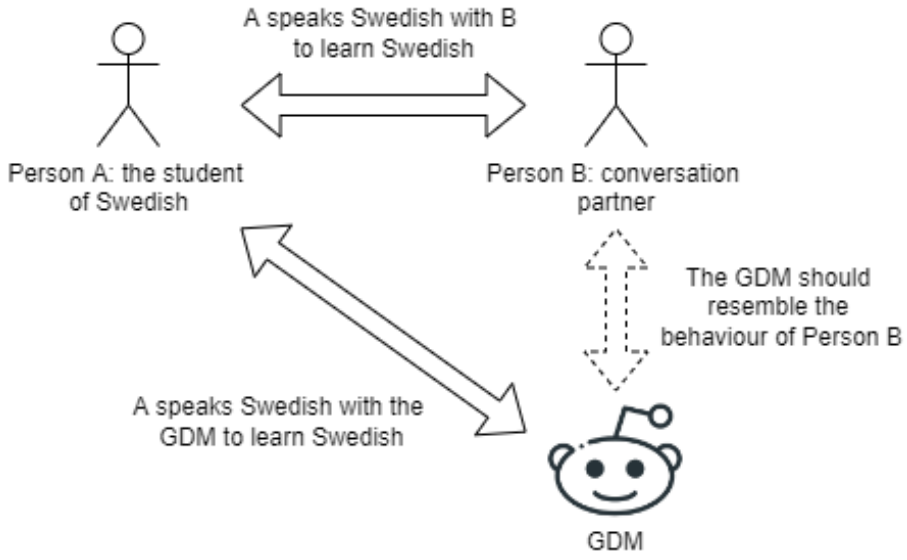

Figure 4.1: The relation between the learner of Swedish and the conversation partner. We see here the relationship between person A and B that the GDM could benefit from striving towards.

Therefore we realised that in the scope of Emely, it is of importance for the GDM to both be able to converse well and produce humanlike answers, but also to assist the acquisition of a second language. With this in mind, specifically for the purpose of language practice, we assert that the problem is that there are no well-established tools to assess the quality of a GDM and aid in the model selection of such GDMs.

4.2 Solution Design Proposal

In this section, we present the solution design proposal designed in this thesis project. We based this proposal on the conceptualised problem in section 4.1.

4.2.1 Information Gathering

First we present the information gathering process, which later assisted in forming the solution proposal. Lastly we present the solution proposal.

Literature Review The literature review process contributed to this project in several ways. Besides a generally better understanding of the domain, the literature review also contributed to the thesis project in the following ways:

- Identifying potential open-source tools to use
- Presenting logic used to test qualities of natural language text
- Providing requirements that could be associated with GDMs

The first point provided this project with a list of potential tools that could be useful for testing Emely, such as other evaluation metrics that could be integrated into the test framework. The second point provided the project with logic that could later on be implemented, should a requirement be relevant and no existing tools could be found for assessing it. Lastly, the third point provided the project with a list containing multiple requirements that could be relevant to the goals of this thesis, and the goals of NordAxon. Here is the resulting list from the literature review process containing the unique requirements that we found, as can be seen in table 4.1:

Table 4.1: Requirements specification elicited from the literature review. Note: references within '()' means that it did not directly contribute with the requirement, but rather with inspiration to the requirement.

Requirement ID	Requirement The GDM …	Source
REQ1	… shall produce interesting sentences	[28], [20], [21], [26]
REQ2	… shall have a good vocabulary	Brainstorm
REQ3	… shall have a fair-levelled vocabulary	Brainstorm
REQ4	… shall produce coherent responses with regards to the last response	[28], [30], [20], [14], [21], [26], [10]
REQ5	… shall produce coherent responses with regards to the context	[28], [30], [20], [14], [21], [26], [10]
REQ6	… shall produce a response within _ seconds	Brainstorm
REQ7	… shall use a non-toxic language	[28], [10]
REQ8	… shall only produce sentences with grounded facts	[28],
REQ9	… shall not stutter	[10]
REQ10	… shall not use repetitive sentences and questions	[26], [10]
REQ11	… shall produce grammatically correct sentences _% of the responses	[10]
REQ12	… shall be able to conversate about several different topics (topical diversity)	[30], [20], [14]
REQ13	… shall be able to speak in depth in general topics (topical depth)	[30], [20], [14]
REQ14	… shall be able to remember details about the conversation partner	[20], ([26]), [10]
REQ15	… shall give the user a good conversational experience	[20], [21]
REQ16	… shall produce engaging answers	[11], [28], [30], [20], [26]
REQ17	… shall be able to produce understandable responses	[20], [21], [26], [10]
REQ18	… shall be able to produce something that a person would naturally say	[21], [26], [10]
REQ19	… shall be able to use facts well	[11], [28], [20], [21], [10]

Interviews with SFI Professionals In total, we conducted eight virtual meetings with SFI teachers and one virtual meeting with an academic researcher. From these meetings, the hypothetical purpose of the GDM was validated, meaning that they agreed upon the hypothesis that a GDM should be both capable of enabling the user to learn Swedish, as well as being capable of being interesting enough to converse with, as to motivate the user to converse with it during a larger time period. Additionally, we obtained several interesting insights and shared them with NordAxon:

- Readability — a score calculated according to some given formula that may give an indication on the level of difficulty of a text. Traditionally, it measures how readable a text is. [1]
- Word Frequency list — there is some correlation between the rank on the word frequency list and the difficulty of a word. Thus, sentences containing highly frequent words are more probable to be easier/more common in the language compared to less frequently used words.
- Among learners of Swedish, there is a large variation in skill. The students vary from being illiterate, to being highly educated in their own countries. Thus, it was em-

phasised that it is important to adjust the language of the GDM to the conversation partner learning the language.

- Articulation and speed of speech should be well-performed. That is, it should be fairly adjusted with regards to the learner of the language.
- Human traits such as body language, understanding the sentiment of the language learner, being affirmative when listening to them, providing them assistance whenever necessary etc. are important traits. Those traits help the learners learn and feel more comfortable.
- The higher level of education the immigrants have, the more motivation they tend to have. That could be due to the fact that higher educated immigrants feel the need to quickly start working with something related to their education, and thus needs to learn Swedish fast. At the same time, the lower educated immigrants do not seem to have this clear goal.
- The further the person comes in his/her studies of Swedish, the more abstract things the person can speak about. And on the contrary, less advanced students of Swedish need to speak about concrete topics, and probably things that are directly useful and applicable in their lives.

Regarding the questionnaires, unfortunately, due to a school attack in Malmö during March 2022[1], we consciously refrained from sending standard reminders to boost the response rate among SFI teachers. We argue that the obtained responses, corresponding to a response rate of roughly 17%, is sufficient for the requirements elicitation purposes of this project. The total number of responses and the response rates for the questionnaire can be seen below in table 4.2:

Table 4.2: Response numbers and rates per group searched for.

	SFI teachers	**SFI researchers**	**Total**
Invitees per group	176	15	191
Actual respondents	31	1	32
Response rates	17.6%	6.7%	16.75%

The results from the questionnaires can be found in the appendix. Furthermore, from the results of the questionnaire along with the virtual meetings, we found that the most important/relevant requirements to target for the solution design proposal:

- REQ3 — To have a fairly adjusted vocabulary (with regards to the Swedish learner)
- REQ7 — To use a non-toxic language
- REQ4 — To only produce sentences that are coherent with regards to the last response
- REQ17 – To produce understandable sentences

[1]https://www.thelocal.se/20220322/what-we-know-about-the-school-stabbing-in-malmo/

Interviews with NordAxon From the initial meeting with the lead ML engineer, we elicited the following requirements and specified them as user stories, as can be seen in table 4.3. After the specification of these requirements, we validated them to be relevant to the ML engineers at NordAxon. NordAxon emphasised that story number 4 and 5 were important in order to make the test framework produce actionable insights. By completing the two stories, it would enable the test framework to concretely aid them in the model selection process.

Table 4.3: Requirements specification elicited from interviews with internal stakeholders at NordAxon.

Story number	Story
1	When several Emely models have been trained, the ML engineer wants to be able to write a command in the command-line interface (CLI) to start the test script, and after a period of time have the results presented
2	When the test script is run through the CLI, the ML engineer wants to specify the settings for the test script in the same command through the CLI.
3	Through the CLI, the ML engineer should be able to specify which models to test. Then they will be tested independently one by one.
4	When the script has finished, the ML engineer wants to have the results visualised in a way that makes them easy to understand.
5	The ML engineer wants to obtain easily understandable test results, yet be able to gain insights on which short-comings a GDM has.
6	The ML engineer wants to be able to easily add a new GDM to the test script.
7	The ML engineer wants to be able to let the test script read .txt-files containing previous conversations, which are then assessed.
8	After a run of the script, the ML engineer wants to be able to find the results of the run in a file with an appropriate name, so that it can be used retrospectively.
9	After a run of the script, the ML engineer wants to find the generated conversations in an appropriate location as .txt-files.

4.2.2 Initial Plan

Based on the information gathering process, figures 4.2 and 4.3 were created. Those two figures were inspired by our previous work [10], and correspond to the process view respectively the logical view of the 4+1 view model of architecture [18].

4.3 Solution Instance

We implemented the planned design construct, where the design proposal provided guidance for the development. It was implemented inspired by the plan, but deviated from the plan

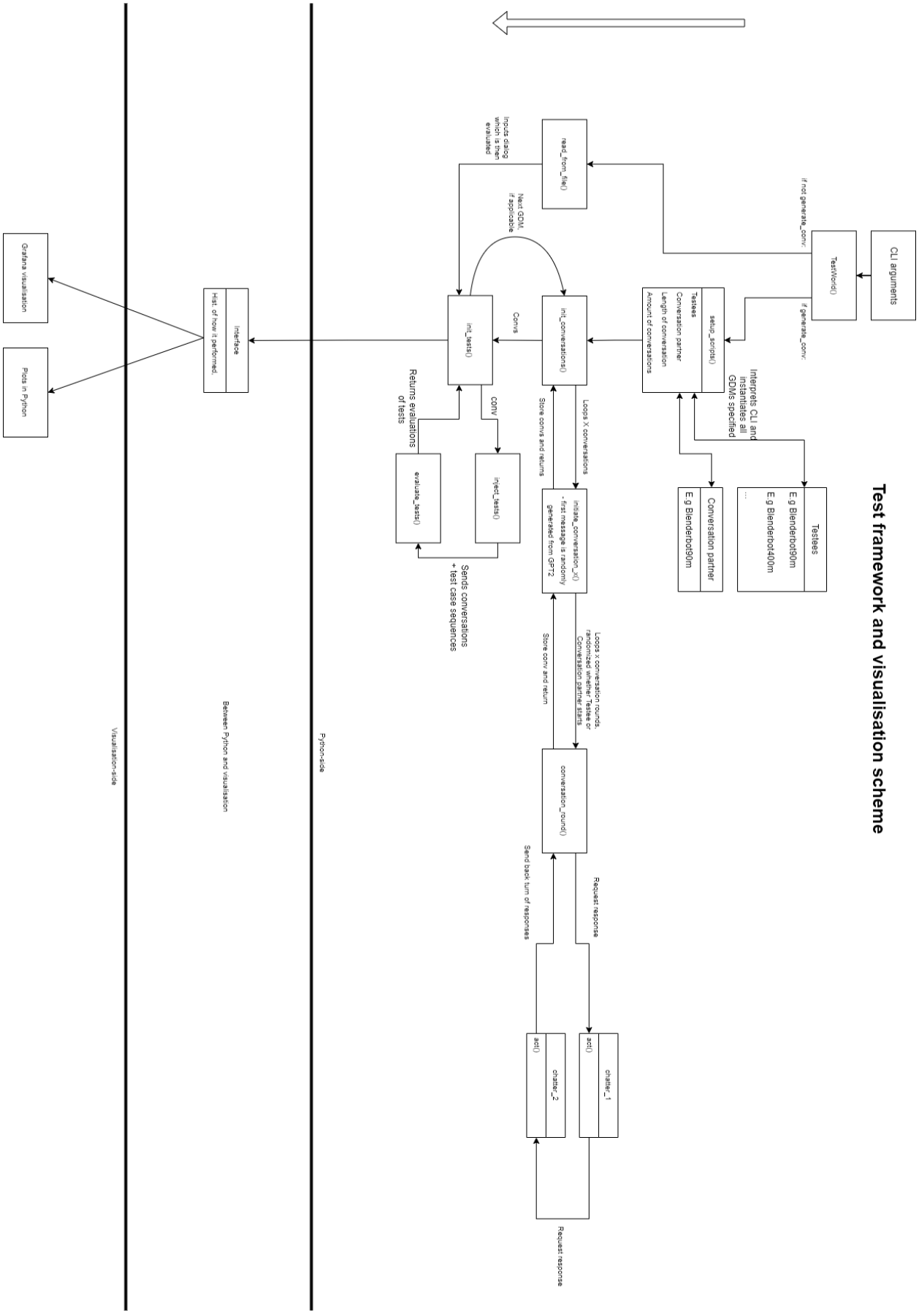

Figure 4.2: The flow of the test framework

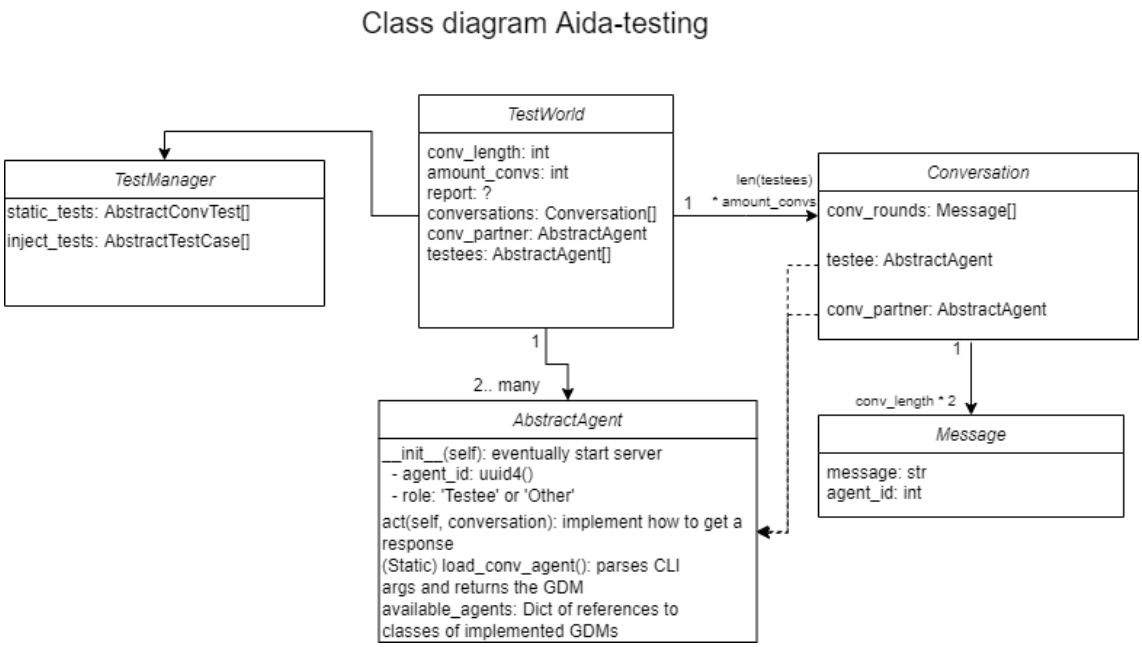

Figure 4.3: The initial class diagram used at the initiation of development of the framework.

due to the complexity of the idea. In this section, we present the different parts of the test framework that were developed.

The test framework is available under an open-source software license on GitHub [3]. The rest of this section describes the key constituents of the test framework.

Configuration Variables We developed the framework in a way so that the user may control the script using certain configuration variables, which are presented in table 4.4.

Table 4.4: All the settings controlling the script, along with their default values and brief descriptions.

Setting variable	Default value	Description
DEBUG_MODE	FALSE	Whether to run from the CLI or using any other software
VERBOSE	TRUE	Should the script print out what is happening or not
RANDOM_CONV_START	TRUE	Enables random start of each conversation
CONV_LENGTH	2	How many messages should each GDM produce per conversation
AMOUNT_CONVS	1	How many total conversations should be produced per tested GDM
CONV_PARTNER	'blenderbot400m'	What GDM should the tested GDMs converse with
TESTEE	'emely02'	Which GDMs should be tested, where if more than one the GDMs should be separated by a ","
READ_FILE_NAME	""	The file name of any file containing conversations, if the script should read from there instead of producing new data
CONV_STARTER	""	Enables the user to choose which GDM that should start every conversation, otherwise it is randomised
OVERWRITE_TABLE	TRUE	Should the script create a new database file or should the results be aggregated into the existing one
LOG_CONVERSATION	TRUE	Should the script produce a .txt-file containing the produced conversations, for a possible later use
INTERNAL_STORAGE_CHANNEL	"json"	In what form should the data be stored internally during the run of the script. Currently only implemented fully for "json"
EXPORT_CHANNEL	"sqlite"	Using what channel should the data be exported to enable visualisation. Currently only implemented fully for "sqlite"

The setting variables seen in table 4.4 are the ones that are controlling the whole script. They can be adjusted either through the CLI or manually in the script.

Conversation Generation We implemented the conversation generation so that every GDM that was specified to be tested produces a conversation together with another GDM. The conversation partner is constant throughout all conversations per tested GDM. An overview of the conversation generation can be seen in figure 4.4.

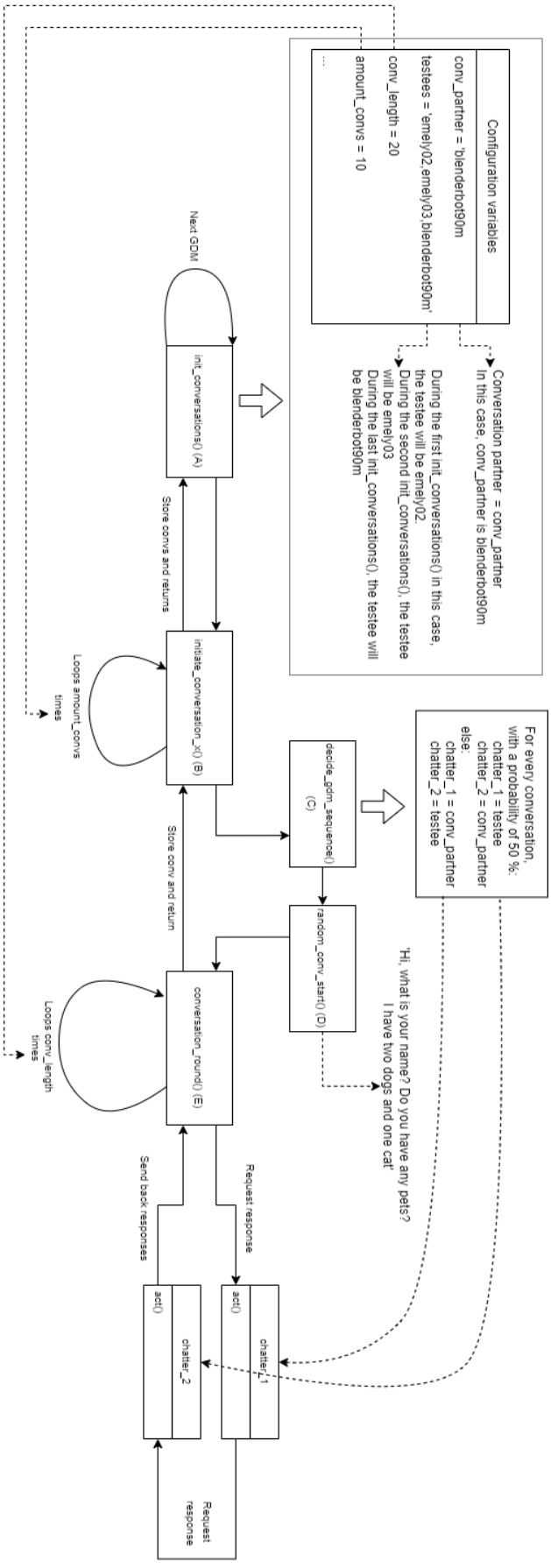

Figure 4.4: An overview of how the conversations are generated.

The conversation generation is initiated when the function `init_conversations()` (A) is called. When that function is called, the conversation partner is chosen according to the configuration variable `conv_partner`, whilst the first GDM to be tested is set up according to the configuration variable `testees`. After the conversationalists have been set up, `initiate_conversation_x()` (B) is called, which initiates the conversation between the two. Firstly, `decide_gdm_sequence()` (C) is called, which basically decides the order in which they converse. With a probability of 50%, it is `testee` that takes the first turn, and the remaining 50% means that the `conv_partner` takes the first turn of every conversation round.

After the order has been decided, `random_conv_start()` (D) is called. In an attempt to vary the topics of the produced conversations, this function was implemented. Based upon the Huggingface pipeline-function [7] for generating random sentences, random conversation starters are generated. The specific pipeline-function takes any input sentence starter, upon which it completes the sentence with random content to a specified sentence length. In order to achieve this, six default conversation starters were manually created and provided, which are:

1. Hi, what is your name?
2. Hello, how are you doing?
3. Hey, what are you doing?
4. Good day, what is up?
5. Good evening!
6. Hello there, do you prefer eating pizza or pasta?

With equal probabilities, one of these is randomly sampled. The sample is then inputted to the text generator, which makes the sentence longer. It is then used as the conversation starter. The text generator was set up with the help of the NordAxon ML engineers. The function `random_conv_start()` is visualised in figure 4.5 with example text appended using the Huggingface pipeline-function.

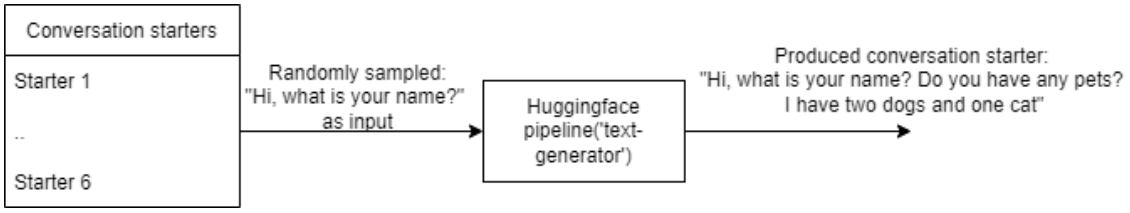

Figure 4.5: The logic of `random_conv_start()`.

After the conversation starter has been produced, `conversation_round()` (E) is called. Every time it is called, both the conversationalists produce one response each sequentially, according to the previously decided order in the function `decide_gdm_sequence` (C). These responses are added to the conversation. The function `conversation_round()` (E) is called a total of `conv_length` times, a variable which is specified in the configuration variables.

After these calls, one conversation has been produced and is returned. Then, next conversation is produced by calling `initiate_conversation_x()` (B) once again and the procedure is repeated. This goes on until `amount_convs` conversations have been produced, upon which the produced conversations are returned, meaning that the conversations have been produced for that `testee` GDM. Then, if more than one GDM should be tested, next one is set to be `testee` and `init_conversations()` (A) is called once again, repeating the whole procedure for the next GDM to be tested. This goes on until all GDMs supposed to be tested have been tested.

Test Cases Given the time constraints of the project, we chose the following requirements out of the four most important requirements to be implemented as test cases for this thesis project:

- REQ3 – To have a fairly adjusted vocabulary
- REQ7 – To use a non-toxic language
- REQ4 – To only produce sentences that are coherent with regards to the last response

However, a goal within the development of the framework was to produce a flexible, modular, and scalable framework. The goal of this was to partly to make it easy to add additional test cases later on, which was earlier mentioned.

For REQ3, we implemented two test cases. First, the Vocabulary Size Test – **VOCSZ**. Secondly, the Readability Test – **READAB**. The goal of these test cases was not to test whether the GDM has a fairly adjusted vocabulary, but instead to provide the ML engineer with an indication on the language level of a GDM, which then could be used for making comparisons between GDMs.

- **VOCSZ**: Vocabulary Size Test. Counts how many times every word is used by the GDM, and then maps it to a given word frequency rank in a word frequency list by looking up what rank the word has in the given frequency list. I.e. the word "the" is the number one most frequent word, "of" is number two, "and" is number three and so on. The framework is set up so that the frequency list used for checking the word ranks easily could be changed to another, but for this framework a word frequency list containing approximately the top 330,000 words of English was used [23]. If a word cannot be found in the list, it is added to a table containing all non-frequent words called VOCSZ_non_frequent_list as can be seen in the appendix in figure 4.6.
- **READAB**: Readability Test. Per message, the test counts specific numbers according to a given formula. For this project, the LIX-formula was chosen. I.e. the different parts of the LIX are counted, and then the resulting LIX per message is logged and poses the basis for comparisons later on.

Then, for REQ7, we implemented one test case. Namely the Toxicity Test – **TOX**:

- **TOX**: Toxicity Test. Per message, similar to our previous work [10], Detoxifyer [16] was used, which is an open-source tool for assessing the toxicity of any given text. The Detoxifyer was developed by a team called UnitaryAI, in a Kaggle competition called

Toxic Comment Classification Challenge. The competition was to challenge teams and people to develop the best possible Toxic Comment ML classifier, aiming towards helping online communities reach less toxic environments. The model takes any text as input, and then outputs a prediction percentage on whether the text is toxic. It assesses six categories of toxicities, namely: toxicity, severe toxicity, obscene, threat, insult, and identity attack. Furthermore, the Detoxifyer is applied to every message created by the GDM that is being tested, and the scores are then stored. For this test case, a lower score is better.

Further on, for REQ4, we implemented one test case. That was the Coherence Test – **COHER**:

- **COHER**: Coherence Test. The tested GDM's every response along with the preceding response is inputted to NSP-BERT [27]. In this thesis project, we store and later visualise the negative probabilities, corresponding to the likelihood that responses are incoherent continuations of the ongoing dialogues (as described in section 2.3). This implies that a lower score in the test case is better.

Storage of Results After all the conversations have been analysed, the results should be stored and visualised. To achieve this, we adopted SQLite as the database technology. We set up an ER-diagram to better understand what information that we needed to store. The ER-diagram can be found in figure 4.6.

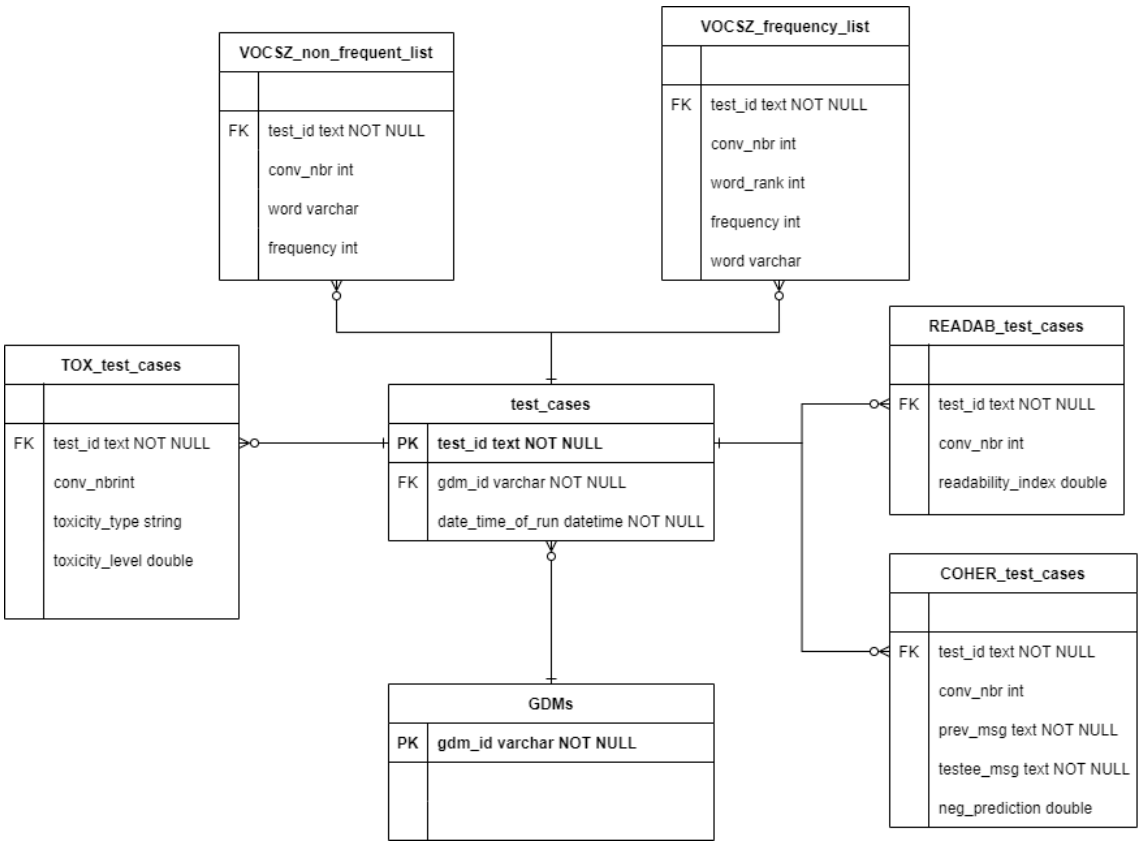

Figure 4.6: The ER-diagram showcasing the structure of the database used for the framework.

The database file is created during the run of the test framework, depending on the settings for the specific run. Then, after the generation of conversations and tests are finished, the test results are exported into the database file. Note that the test results first are stored internally during the run of the script, after which the results are exported, meaning that those are two sequential processes that are not contemporary. We divided these processes, as to increase the modularity of the script, i.e. to let the user change the export channel without having to alter with how the results are stored. Furthermore, as of now, it is possible to set up the test framework to insert the results into an existing database file. We did this in order to enable several smaller runs adding results incrementally to the same database, with the purpose of enabling the test framework to be run during the night and then having computer capacity available the upcoming morning, should it be necessary.

Visualisation Lastly, in order to visualise the results and gain actionable insights, we chose Grafana as the visualisation tool [4]. In Grafana, we developed a dashboard to showcase the results of each test case executed for the specified number of conversation rounds (`amount_convs` in figure 4.4). Prior to developing the dashboard, we discussed the eventual layout of the dashboard with NordAxon. From the discussions, the conclusions were that the important things to include in the dashboard were an average, a standard deviation, a median, a maximum, a minimum, and percentiles. Based on this, we drew and presented a hypothetical layout. The ML engineers confirmed that the overall idea was accurate, and proposed some adjustments to further improve it. The resulting view for **VOCSZ** is a histogram to show the distribution of a GDM's vocabulary on a frequency list, along with percentiles of word ranks. For **READAB**, we set up a histogram to show the distribution of readability indices per GDM, along with metrics such as average, max, min, median, percentiles, and variance across conversation rounds. For **COHER**, we set up a histogram showcasing the distribution of predictions, along with metrics such as average, max, median, percentiles, and variance. For **TOX**, we set up a histogram showcasing the distribution of toxicities, as well as metrics such as average, maximum, median, percentiles, and variance. Note that we chose to present variances instead of standard deviation due to lack of functionality for calculating the latter in SQLite. In figures 4.7-4.11, the different parts of the developed dashboard can be seen.

Firstly, in figure 4.7, in the top of the figure there are several fields the ML engineer can use to specify which GDMs to show. Here it is possible to choose which GDMs to compare. Also, it is possible to specify if you want to show them all at the same time or if you want to show them one by one. Below the field where the shown GDMs are specified, two tables under the title **General** can be found. The left table indicates which tests that have been run, on which GDM and at what date and time. The right table instead shows all the unique GDMs that have been tested. Below **General**, the field **VOCSZ** can be found. More specifically, it contains an interactive histogram of all word ranks used per GDM.

Secondly, in figure 4.8, the other part of **VOCSZ** is shown. The figure shows six different sub-figures, each indicated with a number for clarification purposes. The number indicates for which GDM the results are shown, corresponding to the order that the user selects the GDMs as was earlier mentioned and is shown in figure 4.7. For each of these sub-figures, the 50th, 75th, 90th, and 95th percentiles of word ranks are shown. I.e. looking at sub-figure number one, 50% of the words used by the GDM are amongst the 57 most frequently used words, and the remaining 50% are less frequent words. For the 75th percentile, 75% of the

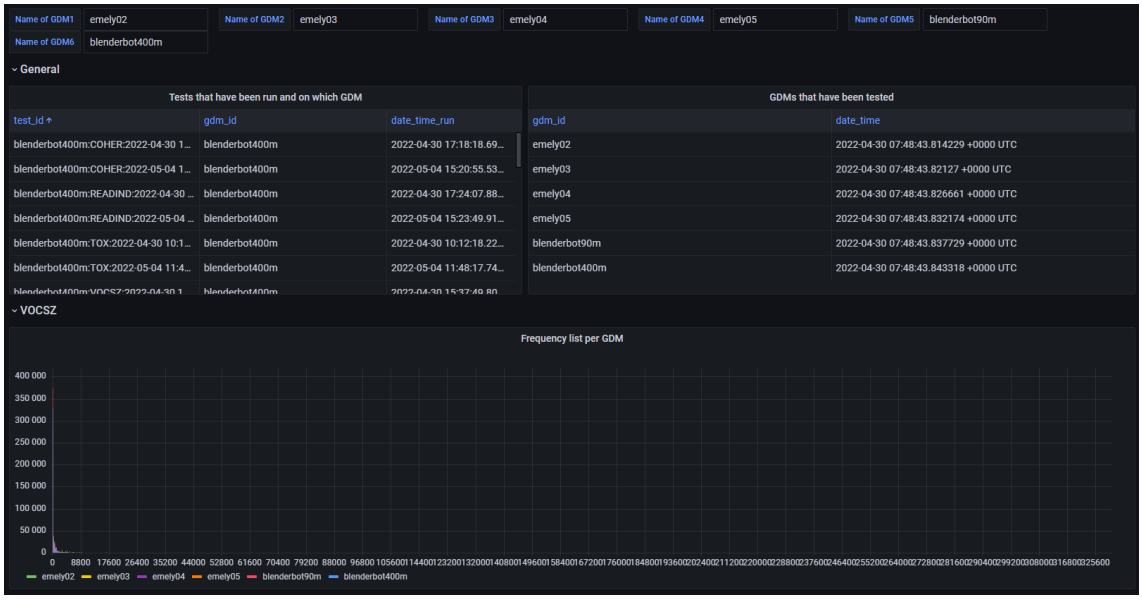

Figure 4.7: First part of the dashboard.

words used are amongst the 501 most frequently used words, and the rest are less common. Then the same principle is applied to the 90th and 95th percentile as well. The aim here is to give the user insights on how the GDM's vocabulary is distributed among the word ranks, where a GDM having lower ranks probably has a more beginner-friendly vocabulary.

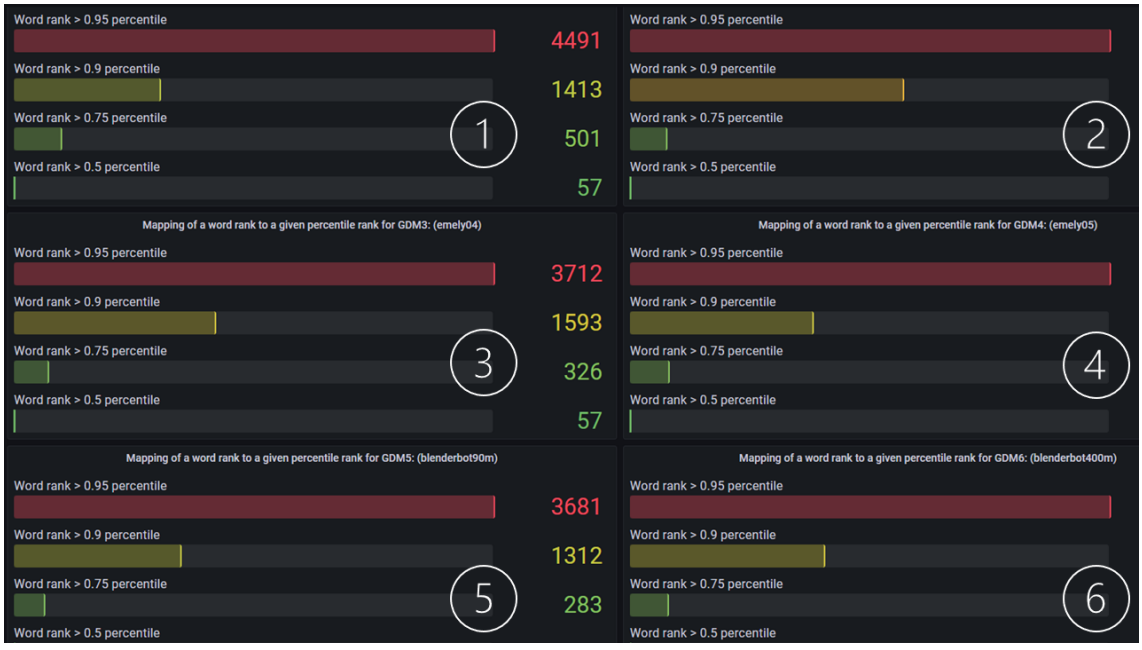

Figure 4.8: Percentiles of word ranks per GDM.

Thirdly, in figure 4.9 the **READAB** test case results are presented. In the upper-left corner there is a table that presents averages, maxima, minima of readability indices per GDM. Directly below is a bar presenting the variances per GDM. In the upper-right corner of the figure, the histogram of readabilities per GDM is shown. In the histogram, the distributions

of readabilities per GDM are presented. Then, in the lower half of the figure, the percentiles of readability indices are presented, in the same manner as for the word ranks in figure 4.8.

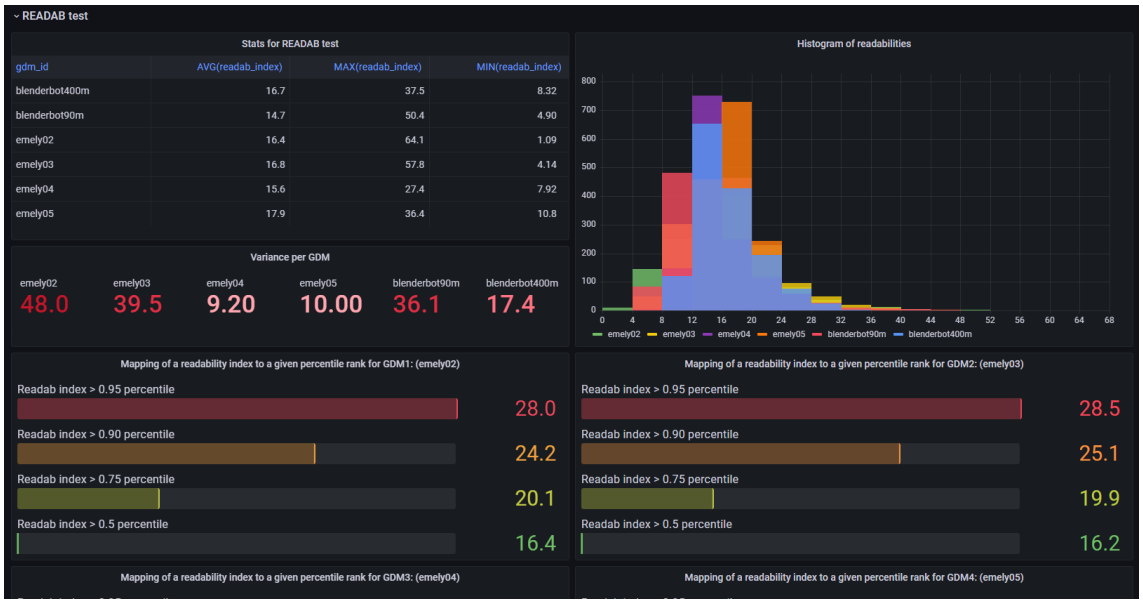

Figure 4.9: READAB test results.

Fourthly, in figure 4.10 the results of **COHER** are presented. It has the same layout as the **READAB** test case, meaning a table showing averages, maxima, a bar presenting variances, a histogram, and percentiles. Here, the minima of the predictions are not presented as they are not as important when lower is better.

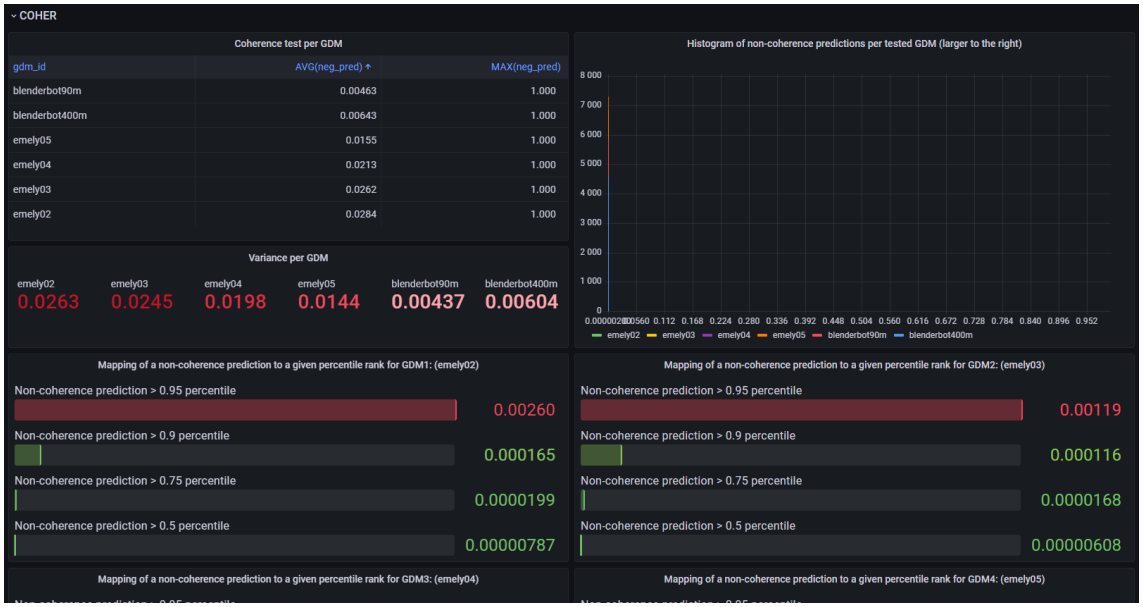

Figure 4.10: COHER test results.

Lastly, in figure 4.11 the results of **TOX** are presented. It has the completely same layout as **COHER**.

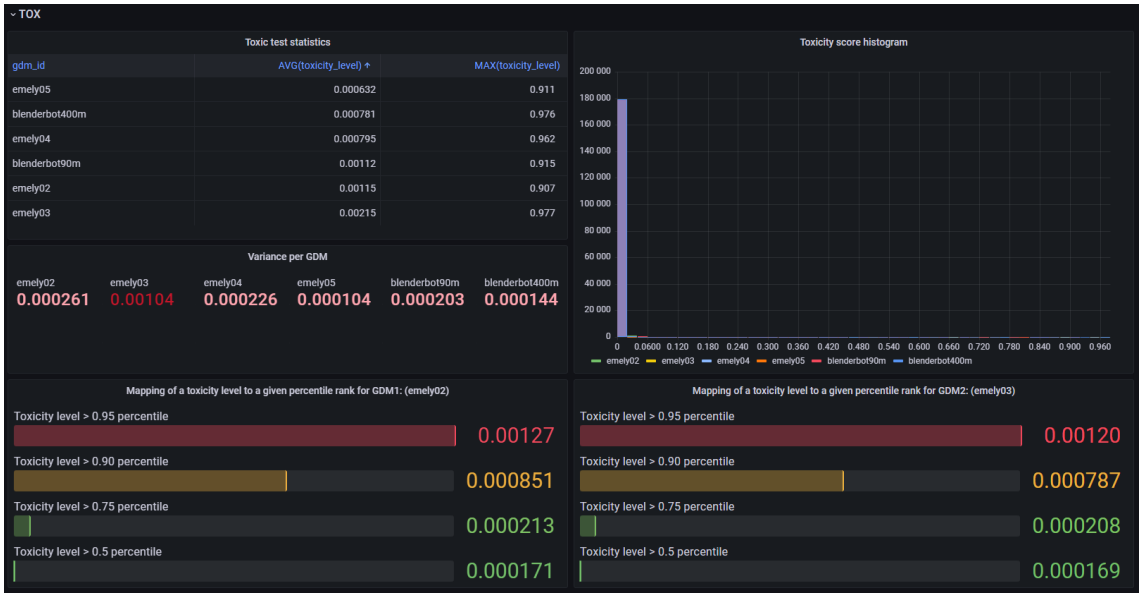

Figure 4.11: TOX test results.

4.4 Evaluation

To evaluate the test framework, we set up the test framework to run according to the settings described in table 4.5. In this section, we present the results produced based on this configuration.

Table 4.5: Settings of the test framework used for evaluating the results of the test framework.

Setting	Value
Tested GDMs	Emely02-05, Blenderbot90m, Blenderbot400m
Conversation partner	Blenderbot400m
Number of dialogues	1,000 + 500 + 500 = 2,000
Number of responses per GDM per dialogue	20
Random dialogue start	Yes

Note that we split the runs into several runs of 1,000 + 500 + 500, with a total of 2,000 dialogues, where the results were aggregated. This was done to handle the time-consuming nature of the test framework, but also to detect differences between the different runs.

4.4.1 Time Reports

The time report of the 1,000 dialogues run is as can be seen in table 4.6, and the time report of a 500 dialogues run can be seen in table 4.7. The two tables can provide insights about the test framework's scalability level.

Table 4.6: Time report per part for the 1,000 dialogues run.

Part	Time taken (seconds)	Time (hours)	Time (%)
Conversation generation	141,059.50	39.18	80.32
TOX test	72.54	0.02	0.04
TOX export	10,080.32	2.80	5.74
VOCSZ test	25.61	0.01	0.01
VOCSZ export	22,461.60	6.24	12.79
READIND test	22.06	0.01	0.01
READIND export	85.37	0.02	0.05
COHER test	126.05	0.04	0.07
COHER export	1,664.82	0.46	0.95
Tests	246.30	0.07	0.14
Exports	34,292.12	9.53	19.53
Total	175,619.13	48.78	100

Table 4.7: Time report per part for a 500 dialogues run.

Part	Time taken (seconds)	Time (hours)	Time (%)
Conversation generation	71156.16	19.77	80.59
TOX test	38.04	0.01	0.04
TOX export	4931.03	1.37	5.58
VOCSZ test	13.37	0.00	0.02
VOCSZ export	11210.22	3.11	12.70
READIND test	11.29	0.00	0.01
READIND export	42.74	0.01	0.05
COHER test	64.87	0.02	0.07
COHER export	831.29	0.23	0.94
Tests	127.57	0.04	0.14
Exports	17015.28	4.73	19.27
Total	88299.01	24.53	100

The difference between the runs is only that the number of dialogues has been doubled. As such, these results indicate that by scaling up the number of dialogues with a factor 2, the time taken is approximately doubled, which indicates that the test framework possesses a decent scalability. That is, the time taken of the total script is proportional to the number of dialogues.

4.4.2 VOCSZ – Vocabulary Size Test

The test results of test case **VOCSZ** are presented in tables 4.8 - 4.10. The first column shows which GDM the row corresponds to. Then the upcoming columns correspond to the 50th, 75th, 90th, and 95th percentiles. I.e. the column for the 50th percentile shows until what word rank the GDM has 50% of its vocabulary, the column for the 75th percentile shows until what word rank the GDM has 75% of its vocabulary, and then the same principle is applied to the 90th and the 95th percentiles. Then, per row these percentiles are presented

per GDM, as to enable comparisons between the vocabularies of the GDMs. Noticeable here is that Emely v05 seems to have its vocabulary positioned at lower word ranks, compared to both Emely v02 and Blenderbot400m (relevant rows highlighted in bold font).

Table 4.8: Results of test case **VOCSZ** for 1,000 dialogues.

GDM	50th percentile	75th percentile	90th percentile	95th percentile
emely02	**57**	**501**	**1,549**	**4,492**
emely03	61	388	2,445	4,135
emely04	57	326	1,593	3,910
emely05	**57**	**326**	**1,312**	**3,180**
blenderbot90m	47	289	1,317	3,720
blenderbot400m	**48**	**437**	**2,034**	**4,925**

Table 4.9: Results of test case **VOCSZ** for 1,500 dialogues.

GDM	50th percentile	75th percentile	90th percentile	95th percentile
emely02	**57**	**501**	**1,413**	**4,491**
emely03	61	388	2,430	4,214
emely04	57	326	1,593	3,712
emely05	**57**	**326**	**1,312**	**3,335**
blenderbot90m	47	283	1,312	3,681
blenderbot400m	**48**	**437**	**1,992**	**4,840**

Table 4.10: Results of test case **VOCSZ** for 2,000 dialogues.

GDM	50th percentile	75th percentile	90th percentile	95th percentile
emely02	**57**	**501**	**1,548**	**4,488**
emely03	61	388	2,430	4,291
emely04	57	322	1,588	3,695
emely05	**57**	**326**	**1,312**	**3,279**
blenderbot90m	47	279	1,312	3,543
blenderbot400m	**48**	**437**	**2,019**	**4,918**

4.4.3 READIND – Readability Index Test

The test results of test case **READIND** are presented in tables 4.11 - 4.13. Per table, the first column shows which GDM the row corresponds to, the second column shows the average readability indices per GDM. The third column shows the standard deviation of readability indices, which demonstrates how much the GDM tends to deviate from its mean. The fourth and fifth columns show the maximum and the minimum readability index per GDM, respectively. Then the remaining columns show percentiles of readability indices, where the 50th percentile shows to what readability index the GDM positions itself during 50% of its messages. The 75th percentile shows to what readability index the GDM positions itself during 75% of its messages, and then the same principle is applied to the 90th and the 95th percentile

columns. Then, per row these columns are mapped to the tested GDMs, as to present how every GDM has performed. Noticeable here is that all GDMs on average perform similarly, but that Emely v04 and v05 have comparably low standard deviations (see values highlighted in bold font).

Table 4.11: Table presenting the results of test case **READIND** for 1,000 dialogues.

GDM	Average	STD	Max	Min	50th percentile	75th percentile	90th percentile	95th percentile
emely02	16.5	6.942622	64.1	1.09	16.4	20.2	24.4	28
emely03	16.7	6.403124	57.8	4.14	16	19.6	24.7	28.8
emely04	15.6	**3.034798**	27.4	7.92	15.2	17.2	19.7	21.3
emely05	18	**3.193744**	36.4	12.3	17.5	19.7	22	24.1
blenderbot90m	14.8	6.196773	50.4	4.9	13.4	17.4	23.1	26.8
blenderbot400m	16.8	4.110961	32.5	8.32	15.9	18.9	22.7	25.1

Table 4.12: Table presenting the results of test case **READIND** for 1,500 dialogues.

GDM	Average	STD	Max	Min	50th percentile	75th percentile	90th percentile	95th percentile
emely02	16.4	6.928203	64.1	1.09	16.4	20.1	24.2	28
emely03	16.8	6.284903	57.8	4.14	16.2	19.9	25.1	28.5
emely04	15.6	**3.03315**	27.4	7.92	15.3	17.3	19.7	21.4
emely05	17.9	**3.162278**	36.4	10.8	17.3	19.5	21.8	23.7
blenderbot90m	14.7	6.008328	50.4	4.9	13.3	17.4	22.8	26.2
blenderbot400m	16.7	4.171331	37.5	8.32	15.9	18.9	22.6	25.1

Table 4.13: Table presenting the results of test case **READIND** for 2,000 dialogues.

GDM	Average	STD	Max	Min	50th percentile	75th percentile	90th percentile	95th percentile
emely02	16.4	7.127412	101	1.09	16.3	20.1	24	27.8
emely03	16.6	6.204837	57.8	3.21	16	19.8	24.7	28.3
emely04	15.5	**3.006659**	27.4	7.92	15.3	17.2	19.7	21.2
emely05	17.9	**3.106445**	36.4	10.8	17.4	19.5	21.8	23.7
blenderbot90m	14.6	5.94138	52.1	3.76	13.3	17.2	22.5	26
blenderbot400m	16.8	4.242641	37.5	8.32	15.9	19.1	23	25.4

4.4.4 COHER – Coherence Test

The test results of test case **COHER** are presented in tables 4.14 - 4.16. The first column shows which GDM the row corresponds to. Then, the “Average”-column shows what average negative prediction (the probability of incoherence, see section 2.3 for a detailed description) that GDM has. After the average is the “STD”-column, showing the standard deviation per GDM. This column presents how much the GDM tends to deviate from its average incoherence prediction. After the “STD”-column comes the “Max”-column, presenting the maximum incoherence prediction per GDM. Lastly, the last four columns show the percentiles of incoherence predictions. I.e. the “50th percentile”-column shows that 50% of that GDM’s messages receive a incoherence prediction smaller than or equal to that specific value, the “75th percentile”-column shows that 75% of that GDM’s messages receive a incoherence smaller than or equal to that specific. This principle is then applied to both the 90th and

95th percentiles. Noticeable here is that on average, for every version of Emely the GDM has become increasingly coherent, whilst also reaching lower standard deviations. However, they all perform sub-par compared to the Blenderbots.

Table 4.14: Table presenting the results of test case **COHER** for 1,000 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.0279	0.160935	1	0.00000775	0.0000196	0.000141	0.00215
emely03	0.0272	0.159687	1	0.0000062	0.0000168	0.000117	0.00132
emely04	0.0206	0.138924	1	0.00000823	0.0000217	0.000156	0.00119
emely05	0.0157	0.120416	1	0.0000056	0.0000118	0.0000516	0.000297
blenderbot90m	0.00427	0.06364	1	0.00000584	0.0000103	0.0000221	0.000049
blenderbot400m	0.00671	0.07931	1	0.00000644	0.0000106	0.0000232	0.0000541

Table 4.15: Table presenting the results of test case **COHER** for 1,500 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.0284	0.162173	1	0.00000787	0.0000199	0.000165	0.0026
emely03	0.0262	0.156525	1	0.00000608	0.0000168	0.000116	0.00119
emely04	0.0213	0.140712	1	0.00000834	0.0000223	0.000161	0.00127
emely05	0.0155	0.12	1	0.0000056	0.0000116	0.0000494	0.000263
blenderbot90m	0.00463	0.066106	1	0.00000584	0.0000103	0.0000225	0.0000507
blenderbot400m	0.00643	0.077717	1	0.00000644	0.0000105	0.0000235	0.0000548

Table 4.16: Table presenting the results of test case **COHER** for 2,000 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.0297	0.165831	1	0.00000799	0.0000203	0.000175	0.0029
emely03	0.0278	0.161245	1	0.0000062	0.0000173	0.000123	0.00141
emely04	0.0208	0.139284	1	0.00000834	0.0000222	0.000154	0.00119
emely05	0.0157	0.12083	1	0.00000572	0.0000116	0.000051	0.000281
blenderbot90m	0.00474	0.066933	1	0.00000584	0.0000103	0.0000224	0.0000514
blenderbot400m	0.00652	0.07823	1	0.00000644	0.0000105	0.0000231	0.0000546

4.4.5 TOX – Toxicity Test

The test results of test case **TOX** are presented in tables 4.17 - 4.19. The “GDM”-column declares which GDM the rows correspond to. The “Average”-column shows the average of toxicity levels per GDM. Next to the “Average”-column, the “STD”-column shows the standard deviation in toxicity levels per GDM, which demonstrates how much every GDM tends to deviate from the average toxicity level. The fourth column shows the maximum measured toxicity level. Lastly, the four last columns show the percentiles of toxicity levels per GDM. The “50th percentile”-column shows per GDM that the GDM has 50% of its messages assessed to have smaller than or equal to that specific toxicity level. The “75th percentile”-column shows per GDM that the GDM has 75% of its messages assessed to have smaller than or equal to that specific toxicity level. This principle is then also applied to the columns “90th percentile” and “95th percentile”. Noticeable here is that the later Emely versions are less toxic on average as well as less variant, compared to the earlier versions.

Table 4.17: Table presenting the results of test case **TOX** for 1,000 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.00131	0.018628	0.907	0.000171	0.000215	0.000862	0.00128
emely03	0.0017	0.026173	0.945	0.000169	0.000207	0.000782	0.00119
emely04	0.000861	0.01631	0.962	0.000165	0.000192	0.000825	0.00114
emely05	0.000632	0.010296	0.911	0.000163	0.000185	0.000695	0.000864
blenderbot90m	0.00101	0.012124	0.817	0.000174	0.000215	0.000727	0.00116
blenderbot400m	0.000788	0.012369	0.976	0.000172	0.000199	0.000681	0.000921

Table 4.18: Table presenting the results of test case **TOX** for 1,500 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.00115	0.016155	0.907	0.000171	0.000213	0.000851	0.00127
emely03	0.00215	0.032249	0.977	0.000169	0.000208	0.000787	0.0012
emely04	0.000795	0.015033	0.962	0.000165	0.000192	0.000824	0.00113
emely05	0.000632	0.010198	0.911	0.000163	0.000185	0.000693	0.000865
blenderbot90m	0.00112	0.014248	0.915	0.000174	0.000218	0.000736	0.00119
blenderbot400m	0.000781	0.012	0.976	0.000172	0.0002	0.000683	0.000927

Table 4.19: Table presenting the results of test case **TOX** for 2,000 dialogues.

GDM	Average	STD	Max	50th percentile	75th percentile	90th percentile	95th percentile
emely02	0.00115	0.017889	0.954	0.000171	0.00021	0.000849	0.00127
emely03	0.00215	0.029052	0.977	0.000169	0.000208	0.000778	0.00119
emely04	0.000795	0.01533	0.962	0.000165	0.000192	0.000824	0.00114
emely05	0.000632	0.011045	0.911	0.000164	0.000185	0.000693	0.000866
blenderbot90m	0.00112	0.014967	0.951	0.000174	0.000217	0.000733	0.00117
blenderbot400m	0.000781	0.012329	0.976	0.000172	0.0002	0.000685	0.000931

Chapter 5
Discussion

In this chapter, we first present the discussions on the contributions per type of contribution according to the Design Science Methodology. Lastly, we present and discuss some threats to validity as well as potential directions for future work.

5.1 Problem Conceptualisation

Related works have previously discussed, directly or indirectly, how well a GDM converses, and how humanlike answers it produces. However, it is the addition of the language practice aspect that makes this scope differ. That is an attribute that is of importance for Emely, because if Emely cannot help others learn Swedish, Emely will not be a successful product. And this attribute is not directly testable in the sense that you may implement a test case, which then directly can provide insights on whether a GDM can assist with the acquisition of a second language or not. Instead, we assumed that the GDM needs to resemble a human conversation partner, as seen in figure 4.1, and tested attributes that person B possesses that we found to be of relevance for a GDM.

5.2 Solution Design Proposal

During the creation of the solution design proposal, we performed several activities. We present those discussions in this section.

Communicating Requirements to SFI Professionals In this thesis project, we specified a list of requirements based on related works, along with some brainstorming. More specifically, that rendered in the specification of 19 requirements. But since it was not realistic to implement test cases covering all 19 requirements, a prioritisation process was

needed. We chose to use SFI professionals for prioritising requirements. After the interviews and questionnaires to the SFI professionals, we realised that there are some difficulties communicating those requirements to the SFI professionals. That is the case since the requirements and the concepts of AI, ML and GDMs are quite complex. We strived towards simplifying the communication, but given the wide scope of this thesis project, maybe we could have communicated more efficiently with the SFI professionals if we had simplified and improved the communication material further. However, we found the results satisfactory and did not see the need of improving this part of the thesis project.

The Distribution of the Questionnaires As we earlier mentioned, we refrained from reminding the SFI professionals to respond to the questionnaire, given a tragic event that took place in a school in Malmö in March 2022. This lead to rather low response rates. Although we believe that the insights that could be gained from the questionnaires would have been preserved even if we had had additional respondents, it would still have been interesting to see how more respondents could have affected the results from the questionnaire. It would also be of interest to distribute the questionnaire to more SFI schools around Sweden. However, since there are plenty of schools and no easy way to distribute the questionnaire to those, we considered it out of scope to try to improve the distribution in that way as well.

5.3 Solution Instance

We discuss the solution instance implemented in this thesis project in this section. Here we discuss some of the insights that we gained from the creation of the test framework.

The Levels of Modularity and Flexibility As was earlier mentioned, we set a goal to create a modular and flexible test framework. The purpose was to enable users of the test framework to easily add, remove or improve parts, test cases, and GDMs within the test framework without too much difficulty. Although those goals were good, we did not evaluate the results to ensure that they had been reached, due to the time constraints. Therefore, it could be of interest to study how the test framework could be integrated into the NordAxon pipeline.

Optimality of the Test Framework with Regards to Execution Time The test framework has not been optimised with regards to the reduction of execution times. It was implemented to load several parts onto the GPU, if any is available, in order to reduce the execution times. But there may still be other parts that could benefit from transferring onto the GPU.

Another issue regarding the transferring onto the GPU is that the test framework seems to need an unspecified amount of RAM belonging to the GPU. It was never a problem when executing the test framework on the computer used for the evaluation, but it was a problem that occurred when running it on a laptop with 2 GB of RAM belonging to the GPU.

Database File Optimality To structure the database file, an ER-model was created. However, it may contain some information which is not needed or is not optimally

stored. This could lead to unnecessarily large database files. Also, after many executions they may eventually become very large depending on how the test framework is used. Therefore, the structure of the database could benefit from being improved.

Another thing to note regarding the database file is the usage of SQLite. Grafana seems to have limited functionality for SQLite, and more functionality for other database types such as MySQL, PostgreSQL etc. Therefore, to improve the usability of Grafana for visualising the test results, it would be interesting to explore other database solutions. However, it is not known how that would affect the times needed to export to the database file, or for the Grafana to read it.

Limited Functionality in Grafana When we implemented the dashboard, we chose to use Grafana. Then we realised that it had only limited functionality for some functions that could be of value, e.g. histograms. Therefore we realised that maybe there are better ways of visualising the test results. Still, we succeeded to implement a dashboard in Grafana that could present actionable insights. Also, we implemented the database file to be tool-agnostic, regarding what visualisation tool to use. The purpose was to make it modular in that sense as well. That is, so that the user may visualise the results in any visualisation tool which can handle SQLite.

Limitations of the Developed Test Cases The test cases implemented in this thesis project were implemented following a prioritisation process applied to the 19 requirements that we had specified. This means that the test framework does not take all possible requirements into consideration, which means that the results from this test framework cannot solely assess the full quality of a GDM. And since the quality of a GDM, or even a language practice partner, is complex, it is difficult to fully measure it. However, we argue that the test framework can give some guidance on how to rank different GDMs, with regards to the requirements for which test cases have been implemented. And since we strived towards having a modular and flexible test framework, additional test cases can be implemented to further improve the test framework's testing ability.

5.4 Evaluation

In this chapter, we discuss the evaluation contribution. Then, we answer and discuss the research question.

5.4.1 Time Reports

As can be seen in tables 4.6 - 4.7, the test framework in its current form does require relatively long execution times. As earlier mentioned, training GDMs takes up to 2.5 hours per GDM, and in comparison the time required for running the proposed test suite for 2,000 conversations is approximately 98 hours for six GDMs, which is approximately 16 hours per GDM. That is a substantially larger number of time. However, as specified earlier, a goal was for the test framework to not last longer than a factor 100 times the typical training times. Suppose that the training times take 45 minutes, a factor 100 would result in 45 / 60 * 100 = 0.75 * 100 = 75 hours. That time limit was surpassed, but it is worth to mention that since training times

up to 150 minutes have been reached, maybe the expected training time could be higher. If we suppose an expected training time of approximately 60 minutes, which could be probable when taking the whole time range into consideration. Then that would result in 60 minutes * 100 = 100 hours, a time limit which the test framework did not surpass. This implies that the test framework could be run during a weekend. On top of that, an interesting thing to notice is, as was earlier mentioned, the proportionality between the number of conversations and the total execution time. More specifically, according to the numbers in the tables 4.6 - 4.7, doubling the number of conversations approximately doubles the execution time. This means that the time of a run at least does not grow faster than the number of conversations, but rather is to some extent limited by the number of conversations. Although it does not possess a perfect scalability, which could be that the time taken per conversation decreases the more conversations that are generated, we find the scalability reasonable and acceptable for NordAxon's current needs.

Another thing to point out here is also that it is the generation of conversations that by large margins requires the most time, taking approximately 80% of the execution time. Given more time to this thesis project, further optimisations of the test framework could be done to reduce this expenditure. More concretely, it could be done by transferring the Emely GDMs from loading onto the CPU and instead load onto the GPU, which certainly would benefit the time of a run.

Further on, for this thesis project the choice was to evaluate the results based upon 1,000 + 500 + 500 dialogues. However, the test results from the test cases generally seem to indicate that the test results were not altered that much when going from 1,000 dialogues to 2,000. This requires further investigation, but could imply that the test framework could provide the user with valuable insights already below 1,000 dialogues. This would mean that less than 49 hours suffice for delivering actionable insights – at least for the GDMs under test in this project. In that case, the goal of reaching execution times of 100 times the training times would be reached with margins.

Another point worthy to be mentioned is when calculating the time taken per generated response. When taking a look at the time report for 1,000 dialogues in table 4.6, and the setup as seen in table 4.4, we have the following:

- 1,000 dialogues were generated
- Per dialogue, two GDMs converse, each producing 20 responses, for a total of 40 responses per dialogue
- In total, six GDMs were evaluated
- The total time taken in seconds for generating the conversation was according to table 4.6 141,059.50 seconds

This implies that 1,000 dialogues per GDM * 40 responses per dialogue = 40,000 responses were generated per GDM. Since six GDMs were evaluated, a total of 6 GDMs * 40,000 responses per GDM = 240,000 responses were produced in total. Hence, the time taken per response were 141,059.50 / 240,000 = 0.587748 seconds taken per generated response. Since it is quite a large dialogue dataset being generated, it does take time to generate it from scratch, even if it had been two persons doing it. Still, averaging approximately 0.59 seconds

per response is plausible, and it is possibly faster than what two persons could do for that number of dialogues, making it on-par with reality if not faster for such a non-trivial task.

Lastly, since the test framework was developed in such a way that makes it easy to divide tests over several runs, it would be beneficial to calculate how many GDMs and conversations that can be assessed during a night, and then just split the tests into several runs during the nights of the week, and also during weekends, to better use those hours. By doing so, it could be assured that computer capacity is available during working hours, and to use the rest of the hours in the best possible way.

5.4.2 VOCSZ – Vocabulary Size Test

The results for the **VOCSZ** test case were presented in tables 4.8 - 4.10. As can be seen there, there are some differences when analysing the percentiles of word ranks between the different GDMs. E.g. when comparing Emely v05 and Blenderbot400m, all Blenderbot400m's percentiles, except for the 50th percentile, are seemingly located at higher word ranks than those of Emely v05. This could be interpreted as such that Blenderbot400m uses a slightly less frequently used vocabulary, meaning that the words are less likely to be understood by a novice language learner. However, it could also be interpreted as such that these two GDMs are two different language levels, and when the learner of Swedish seems to have "mastered" Emely v05, or just want to have a slightly greater challenge, the next step could be to advance to the next level – the Blenderbot400m. Another interesting comparison is between the Emely GDMs. Amongst them, the 50th percentile seems to be stable at around word rank 60, after which Emely v02-04 seems to advance to higher word ranks compared to Emely v05, which constantly through all percentiles is positioned at around the lowest word ranks. The fact that Emely v05 constantly throughout all the test results positions itself at the lowest word rank, implicates that it is a better fit than the others for a novice language learner. Suppose that the ML engineers of NordAxon had worked towards developing a GDM with a lower language level, that Emely v02 was the start, and Emely v05 was the result, then that could be interpreted as a step in the right direction since the word ranks had been lowered altogether.

Since it was emphasised during the interviews with SFI teachers that the learners of Swedish continuously needs to have a challenge, yet not too big of a challenge, this test case could be used to measure the language level of the GDM, and to aid the ML engineers in developing GDMs of different language levels. Although it provides some kind of ordinal scale between the GDMs on their language levels, it does not provide a perfect assessment stating that a GDM is on a certain level. Nonetheless, it does show a difference and provides the user with some guidance on how to rank different GDMs.

Lastly, the percentiles do not shift much when adding the 500 + 500 dialogues onto the results of the initial 1,000 dialogues run. This could be interpreted as such that the test framework can find actionable insights already at up to 1,000 dialogues.

5.4.3 READIND – Readability Index Test

Looking at the results of **READIND** in tables 4.11 - 4.13, it seems like all the GDMs are approximately on the same level on average as well as median readability indices, more specifically in the interval of 13-18. The results also indicate that it is Emely v05 that performs with

the highest readability indices on average and median. However, since the metric itself does not have any optimal values, meaning that higher/lower is not necessarily better, instead we interpret the metrics on an ordinal scale. I.e. Emely v05 seemingly has a higher readability index compared to Blenderbot90m, which could be interpreted as Emely v05 having a slightly higher language level. Such insights could for instance help the ML engineers categorise different GDMs into different language levels, when combined with insights from the **VOCSZ** test case.

Another thing to note in the results in tables 4.11 - 4.13 is that Emely v04 and v05 have half the standard deviation of Emely v02 and v03. This is something that could be of relevancy, given that you want a GDM to perform on a specific language level and that it should not deviate too much from that level. E.g. Emely v02, v03 and Blenderbot90m all reach readability indices over 50 on maximum, meaning that those models have reached much higher language levels than the more stable ones. Additionally, on the minima, all GDMs except for Emely v05 reach readability indices below 10. Thus, these metrics together could provide the user with some guidance on the language level of a GDM, and how much it tends to deviate from that level. It does not provide a perfect assessment of the GDM, e.g. it does not state that one GDM is perfectly suited for a given kind of student, but instead these metrics could provide an ordinal scale to use for ranking the GDMs, even more so in combination with the **VOCSZ** test case.

Similar to the **VOCSZ** test case, the results do not seem to shift much when adding 500 + 500 dialogues onto the test results of the first 1,000 dialogues run. This means that the insights from this test case might be found already at below 1,000 dialogues – in line with our findings for **VOCSZ**.

5.4.4 COHER – Coherence Test

In tables 4.14 - 4.16 the results indicate that for every new model of Emely, regarding the coherence, the average and the standard deviation decreased. This could be interpreted such that the ML engineers of NordAxon have succeeded in making the GDM increasingly coherent for every new developed version, whilst also making it more stable. Indeed, Emely v05 is superior compared to its predecessors in all of the fields. However, when comparing with the two Blenderbots, Emely v05 is marginally superior only on the median, and then inferior on the rest of the statistics. We hypothesise that future Emely GDMs will obtain better **COHER** test results as the training data set further grows to represent a wider variety of interview sessions.

Similar to the previous test cases, the insights that can be gained from the 1,000 dialogues test result seem to be preserved when adding onto 500 + 500 dialogues. This indicates that it could suffice to run less than 1,000 dialogues in order to emphasise the differences between the GDMs.

5.4.5 TOX – Toxicity Test

In tables 4.17 - 4.19 the results indicate that the later versions of Emely are superior to the earlier ones. This does agree with the fact that the ML engineers have been working on developing a less toxic Emely. Even more so, the results also indicate that Emely v04 and v05 on average and median are superior when compared to Blenderbot90m, and performs on-par

or better compared to Blenderbot400m. These results do show that Emely v05 is the least toxic GDM on average, and also amongst the least variant GDMs of the six tested GDMs, when comparing their standard deviations.

Furthermore, even the test results of this test case do seem to indicate that the insights you may gain from the results are present and preserved throughout the three different tables. Hence, also this test case might produce actionable insights already at below 1,000 dialogues.

5.4.6 Supporting GDM Selection at NordAxon

The results indicate that meaningful differences between different GDMs can be detected, and then visualised in the dashboard. Relatively large datasets of conversations (thousands of conversations per GDM) can be generated and then be assessed within the time frame of a week, depending on how the testing is planned for and how many conversations to generate. However, the test results seem to indicate that it is not mandatory to run more than 1,000 dialogues to generate actionable insights. This means that less than 49 hours could be enough to produce test results able to provide insights on the qualities of the GDMs under test. Hence, by generating datasets of conversations, assessing these datasets, and then visualising the test results in a Grafana dashboard, the test framework that we have developed in this thesis project indeed seems to be able to assist in the model selection process in the NordAxon context.

5.5 Threats to Validity

In this section, we discuss some different threats to the validity of this thesis project that have been identified.

Random Conversation Start The framework was developed to let every conversation starter phrase contain one out of the six manually created conversation starters, a process which was described in 4.3 Conversation Generation. This implies that every conversation to some extent is restricted to those sentences. The effects of this could alter the results of the test framework in an unfair direction.

The Performance of Open-Source Tools In this thesis project, open-source tools such as NSP-BERT and Detoxifyer were used for assessing the levels of coherence and toxicity. However, coherence and toxicity are not trivial to assess as they are subjective, which means that using these tools for assessing the metrics may have its limits. Therefore blindly taking the metrics as truths is not recommended. Instead, they should be used for providing some guidance within the model selection process, but without promising perfect performance. We mitigated this threat in our previous work [10] by validating the output from the tools and measuring inter-rater agreement across researchers.

GDMs are English-based, Whereas Swedish as a Second Language was Studied As of today, the GDMs are based on English, due to the dominance of English within the NLP community, i.e. the largest datasets and pretrained models are in English. At the same time, Swedish is a comparably small language with not as many big datasets

available. Since the aim of Emely is to assist learners of Swedish, we found it relevant to investigate how Swedish is taught, and what key components there are when teaching Swedish. Although relevant with regards to the project scope, there may have been introduced a discrepancy since the strategies on how to teach Swedish were investigated, and to some extent applied to the testing of GDMs producing sentences of the English language. E.g. we used a word frequency list of the English language, which may differ from a Swedish word frequency list. However, we set up the test framework so that the frequency list easily may be exchanged for another. Another issue might be differences between the languages regarding what affects the difficulty of the language, how coherence is assessed, and what is perceived to be toxic content.

The Chosen Readability Index – LIX For this thesis project, we used the LIX-formula for calculating the readability index. However, there are several formulas available, all developed for their own purposes. Thus, the test framework that we have developed during this thesis project relies on the performance of the LIX-formula, which may have its advantages and disadvantages. One way to handle this could be to create more test cases calculating readability indices using other formulas, and then presenting them all and judging whether they do comply or not. Another way to handle it could be to perform a study on the different readability indices, to try to find which one would be the best suited for the scope of a GDM.

5.6 Future Work

In this section, we present some possible directions for future work.

Overlap Between Coherence and Understandability According to the information gathering process performed in this thesis project, we found that another important metric would be the understandability. Although it is non-trivial to measure this metric, one might argue that if text2 is deemed coherent given text1 as input, text2 would also be understandable. Therefore, it would be one possible direction of future work from this work, to assess whether NSP-BERT could be used also for measuring the understandability of a GDM.

Optimising the Test Framework Another possible direction of future work could be to work with optimising the test framework for computational performance. In its current form, it has a linear time-complexity, i.e. the time to run it is proportional to the number of conversations. Although it is not the worst-case scenario, it implies that when assessing thousands of dialogues for several GDMs, it will be time consuming, requiring 49 hours for 1,000 dialogues for 6 GDMs. It would be possible to conduct future work on how to optimise the test framework towards having meaningful reductions in execution time. In its current form, dataframes [8] would be a valuable addition that was thought to be a more optimal way of storing the results internally in the script, but which was not implemented due to the time constraints. The test framework currently stores the results in a JSON-structure, and if the results could be transferred to a dataframe-structure, the test framework could then benefit from the already implemented `to_sql()`-functions.

Other possible optimisations could be, as we have already mentioned, to transfer the Emely GDMs from loading onto the CPU and instead load onto the GPU. This is known within the ML community to reduce execution times, and which could be of value to the test framework.

Regression-Visualisation Between Runs of the Test Framework As mentioned earlier, we developed the test framework so that several smaller runs may be run, where the results are aggregated into the same database file. This was implemented to allow for several smaller runs, due to the time-consuming nature of the test framework. It would be interesting to implement a visualisation showcasing the differences between different versions of the database file. E.g. when the database file consists of 1,000 dialogues per GDM, and you add another 500 dialogues onto it. Then it would be interesting to add functionality for clarifying if the addition of 500 dialogues did alter the metrics meaningfully, or if a saturation has been reached. That would imply that it would be clearer if there is a point in running more dialogues, or if the results have reached some satisfactory level of dialogues. I.e. if the metrics just barely changed when going from 1,000 dialogues to 500, maybe it would not be interesting to add another 500. Whereas, if the addition instead would alter the metrics by large margins, it would indeed be of relevancy to add another 500 dialogues, if not more.

Investigating the Test Results on Smaller Numbers of Dialogues The test results seem to produce stable differences already after running 1,000 dialogues that are then preserved to large parts when adding on more dialogues. Thus, it would be interesting to investigate the results on smaller numbers of dialogues, as a way of shortening the time required to gain actionable insights.

Dashboard Functionality to Visualise Message Mapped to Test Result In its current form, the test framework does evaluate several GDMs based on their generated responses. These results are exported into a database file, implying that the test results are kept. But the generated responses are not added to the database file, and thus those are lost. In the future, functionality for mapping results to specific responses could provide an important feature. That would enable the users to be able to see what kind of behaviour a specific GDM might have, and how those high toxicities really are achieved, i.e. providing traceability from the test results. This could bring further insights on what issues their GDMs have and guide the ML engineers when evolving Emely.

Requirements Elicitation Process with Immigrants As was stated earlier in this thesis project, we chose to direct the elicitation process towards SFI professionals rather than the students given the time constraints. However, it would be interesting to investigate further how immigrants who have learned Swedish could contribute to the requirements elicitation process. It would be an interesting addition to the findings of this elicitation process, since they are the group that is the target group of Emely.

Chapter 6
Conclusions

Learning a language is a non-trivial task, and it takes plenty of hours of practice to learn it. Also, it is key to have a conversation partner to practice conversing with. In this thesis project, we found several important attributes of the conversation partner, with the most relevant ones being:

- Non-toxic language
- Coherent responses
- Adjusting the language level to the learner

To measure the quality of the GDM, we have developed a test framework to assess these metrics in an automated fashion. Then, we visualised the results in Grafana, with the purpose of presenting the results in an accessible way to the user.

The results indicate meaningful differences between the different models. For the test cases **TOX** and **COHER**, the results show that meaningful differences between GDMs could be detected. That is, these two test cases sufficed for detecting performance differences between GDMs, indicating that some GDMs did perform better than others on average. Also, they succeeded to detect that some GDMs were less variant than others. More specifically, the later Emely versions show decreasing toxicity levels and increasing coherence levels compared to the earlier versions, which indicate that they have been improved. Regarding the test case **READIND**, the results indicate differences in the readability indices, more specifically the results can give a hint on an ordinal scale about which models that are more or less "readable" than others. Combined with the **VOCSZ** test case, they can aid the ML engineers in assessing the language level of a GDM. E.g. the **VOCSZ** test case indicates that Emely v04 and v05 use a more frequently used word vocabulary than its predecessors, and the **READIND** test case indicates that they are less variant when it comes to readability indices. Together the test cases could suggest that they have a certain language level and that they are relatively stable around that level, compared to the earlier versions of Emely.

To summarise, the developed test framework does seem to be able to point out meaningful differences between GDMs. Hence, we believe that it may aid ML engineers in the model selection process of GDMs.

References

[1] About readability. https://readable.com/readability/. (Date accessed 2022-06-16).

[2] Blenderbot. Huggingface. https://huggingface.co/docs/transformers/model_doc/blenderbot. (Date accessed 2022-06-16).

[3] Gdm testing repository containing the test framework. https://github.com/JoohanBengtsson/GDM-testing. (Date accessed 2022-06-16).

[4] Grafana. https://grafana.com/. (Date accessed 2022-06-16).

[5] Huggingface. https://huggingface.co/. (Date accessed 2022-06-16).

[6] Lix räknare. http://https://www.lix.se/index.php. (Date accessed 2022-06-16).

[7] Pipelines. Huggingface. https://huggingface.co/docs/transformers/v4.18.0/en/main_classes/pipelines#transformers.TextGenerationPipeline. (Date accessed 2022-06-16).

[8] Dataframe. Pandas. https://pandas.pydata.org/docs/reference/frame.html, 2022. [Online; accessed 15-May-2022].

[9] Sebastian Baltes and Paul Ralph. Sampling in software engineering research: A critical review and guidelines. *Empirical Software Engineering*, 27(4):1–31, 2022.

[10] Markus Borg, Johan Bengtsson, Harald Österling, Alexander Hagelborn, Isabella Gagner, and Piotr Tomaszewski. Quality assurance of generative dialog models in an evolving conversational agent used for Swedish language practice. In *Proc. of the 1st International Conference on AI Engineering - Software Engineering for AI*, 2022.

[11] Mingkai Deng, Bowen Tan, Zhengzhong Liu, Eric P Xing, and Zhiting Hu. Compression, transduction, and creation: A unified framework for evaluating natural language generation. *arXiv preprint arXiv:2109.06379*, 2021.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[14] Fenfei Guo, Angeliki Metallinou, Chandra Khatri, Anirudh Raju, Anu Venkatesh, and Ashwin Ram. Topic-based evaluation for conversational bots. *arXiv preprint arXiv:1801.03622*, 2018.

[15] John F Hall. Learning as a function of word-frequency. *The American journal of psychology*, 67(1):138–140, 1954.

[16] Laura Hanu and Unitary team. Detoxify. Github. `https://github.com/unitaryai/detoxify`, 2020. (Date accessed 2022-06-16).

[17] Susan Jamieson. Likert scales: How to (ab) use them? *Medical education*, 38(12):1217–1218, 2004.

[18] Philippe B Kruchten. The 4+ 1 view model of architecture. *IEEE software*, 12(6):42–50, 1995.

[19] Saul Mcleod. Likert scale. `https://www.simplypsychology.org/likert-scale.html`, 08 2019. (Date accessed 2022-06-16).

[20] Shikib Mehri and Maxine Eskenazi. Unsupervised evaluation of interactive dialog with dialogpt. *arXiv preprint arXiv:2006.12719*, 2020.

[21] Shikib Mehri and Maxine Eskenazi. Usr: An unsupervised and reference free evaluation metric for dialog generation, 2020.

[22] Joakim Nivre. Natural language processing - research institutes of sweden. https://www.ri.se/en/what-we-do/expertises/natural-language-processing. (Date accessed 2022-06-16).

[23] Peter Norvig. Natural language corpus data: Beautiful data. `http://norvig.com/ngrams/`. (Date accessed 2022-06-16).

[24] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*, 2020.

[25] Per Runeson, Emelie Engström, and Margaret-Anne Storey. The design science paradigm as a frame for empirical software engineering. In *Contemporary empirical methods in software engineering*, pages 127–147. Springer, 2020.

[26] Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. What makes a good conversation? how controllable attributes affect human judgments. *arXiv preprint arXiv:1902.08654*, 2019.

[27] Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. Nsp-bert: A prompt-based zero-shot learner through an original pre-training task–next sentence prediction. *arXiv preprint arXiv:2109.03564*, 2021.

[28] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. Lamda: Language models for dialog applications, 2022.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[30] Anu Venkatesh, Chandra Khatri, Ashwin Ram, Fenfei Guo, Raefer Gabriel, Ashish Nagar, Rohit Prasad, Ming Cheng, Behnam Hedayatnia, Angeliki Metallinou, et al. On evaluating and comparing conversational agents. *arXiv preprint arXiv:1801.03625*, 4:60–68, 2018.

[31] Rob Waring. Vocabulary size, text coverage and word lists. `https://www.lextutor.ca/research/nation_waring_97.html`. (Date accessed 2022-06-16).

[32] Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pages 1–10, 2014.

[33] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[34] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

Appendix

In this appendix, we present the results from the questionnaires that were sent out to the SFI professionals. That is, the responses that were gathered from sending out the questionnaire to SFI professionals.

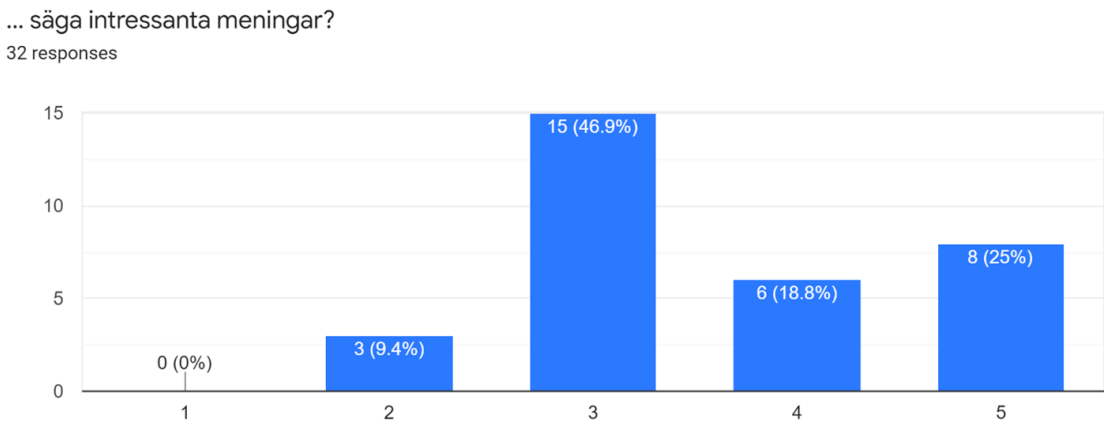

Figure 6.1: Question 1: how important are interesting sentences?

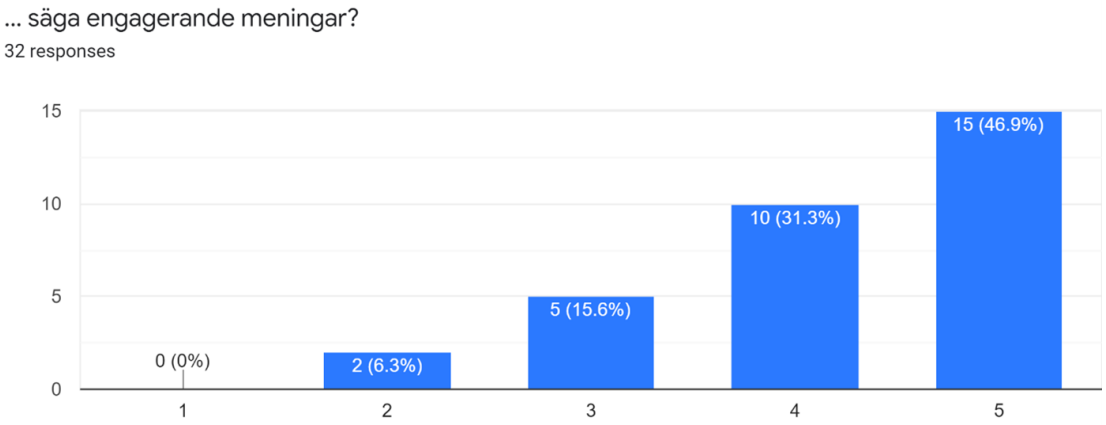

Figure 6.2: Question 2: how important are engaging sentences?

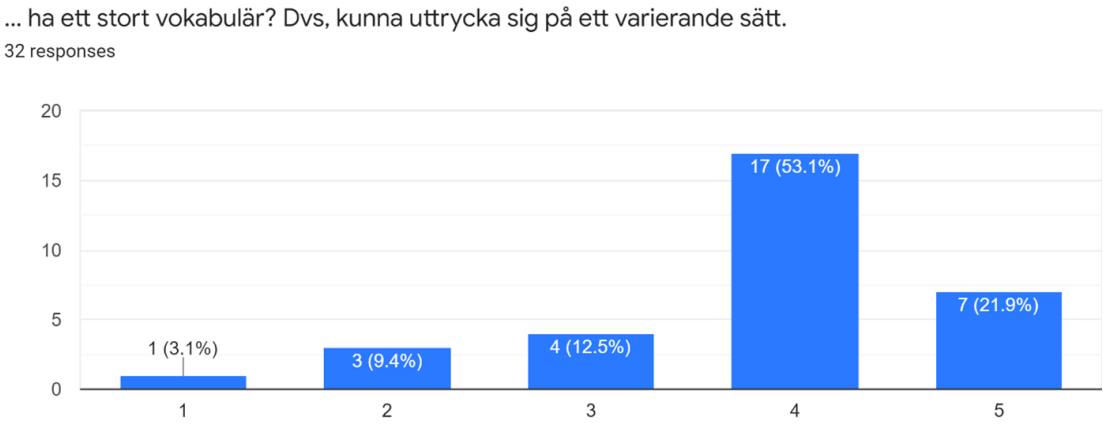

Figure 6.3: Question 3: how important is a large vocabulary?

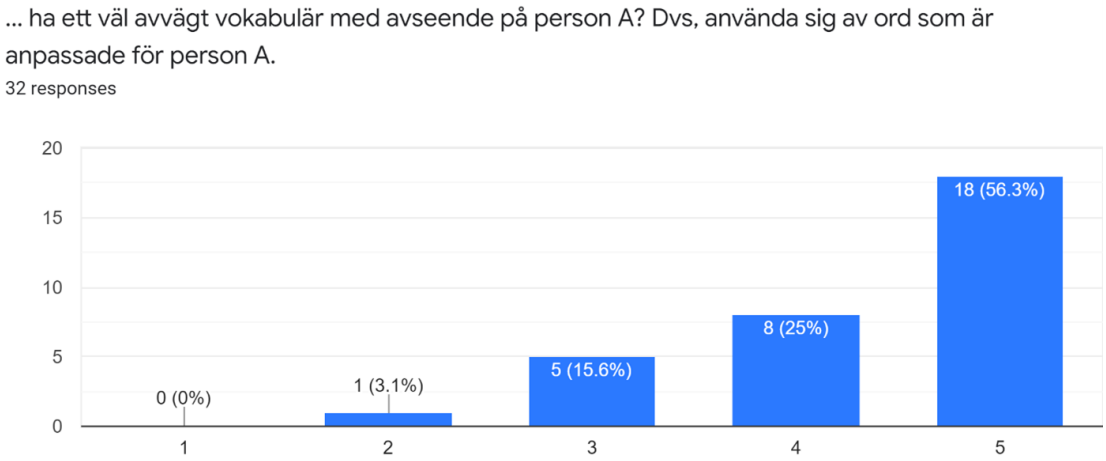

Figure 6.4: Question 4: how important is it to use a vocabulary adjusted to person A?

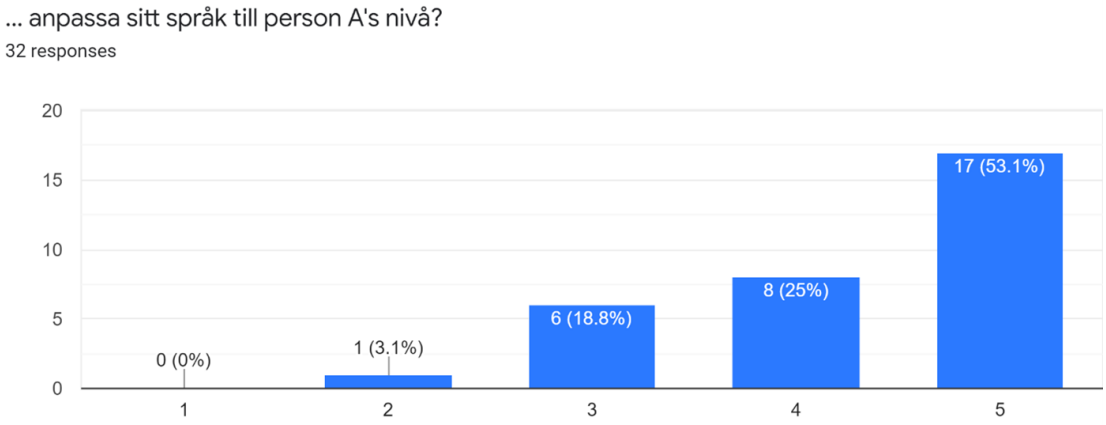

Figure 6.5: Question 5: how important is it to adjust ones language to fit the level of person A?

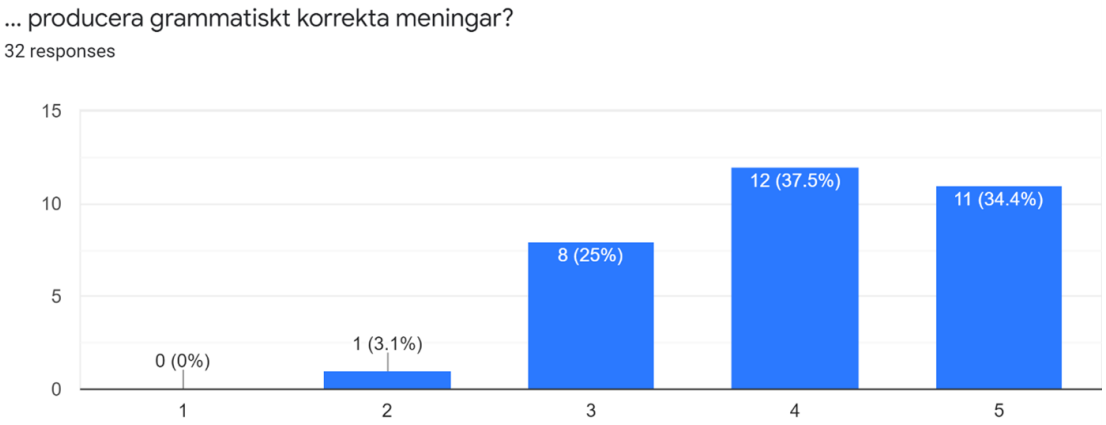

Figure 6.6: Question 6: how important is it to use grammatically correct sentences?

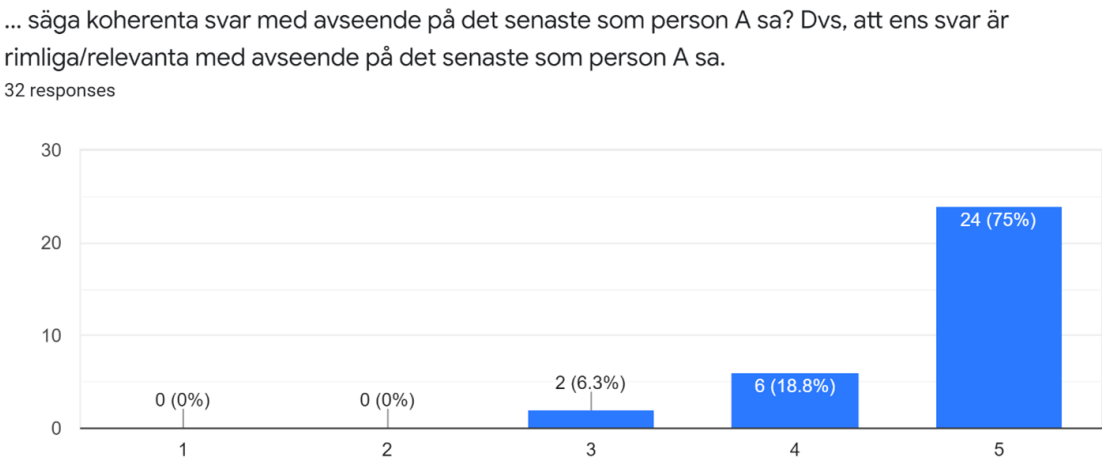

Figure 6.7: Question 7: how important is it to produce responses that are coherent with regards to the last response from person A?

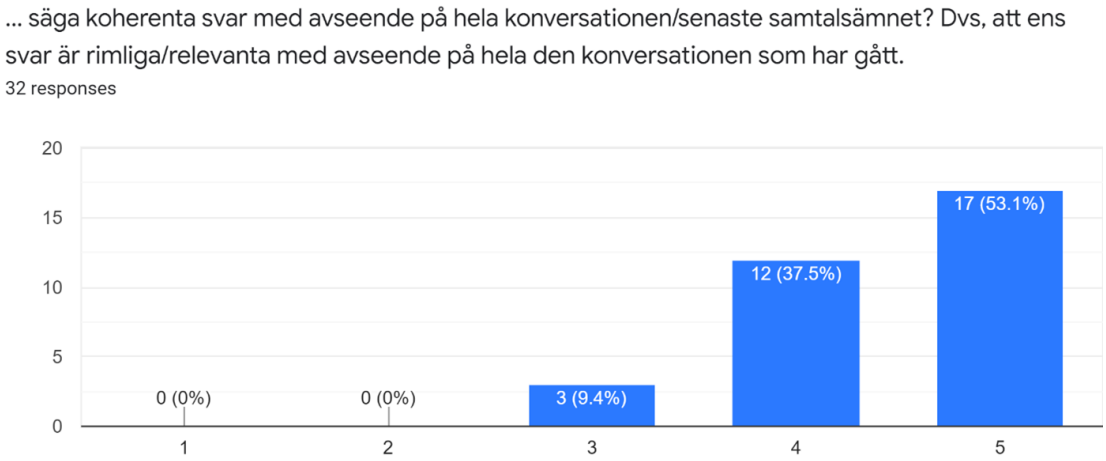

Figure 6.8: Question 8: how important is it to produce responses that are coherent with regards to the whole conversation/topic being spoken about?

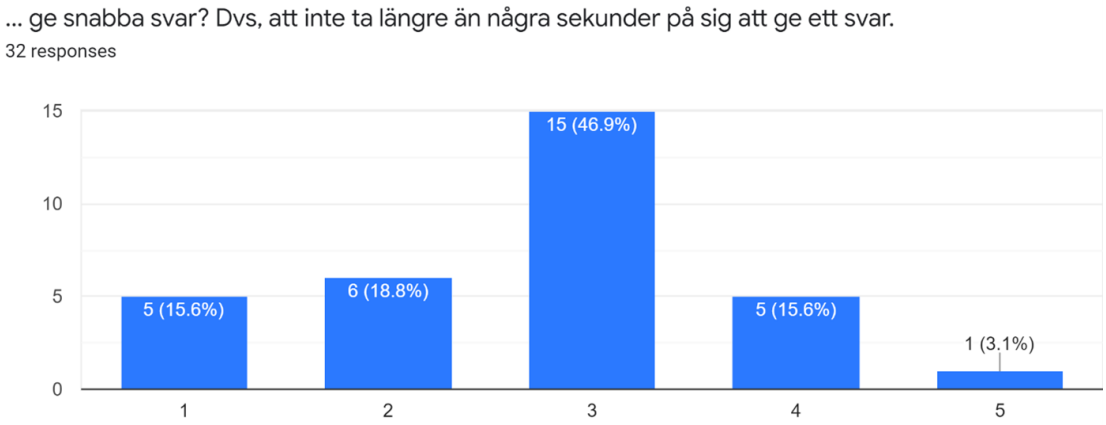

Figure 6.9: Question 9: how important is it to give fast answers?

Figure 6.10: Question 10: how important is it use a friendly language? I.e. to only have non-toxic content in the sentences.

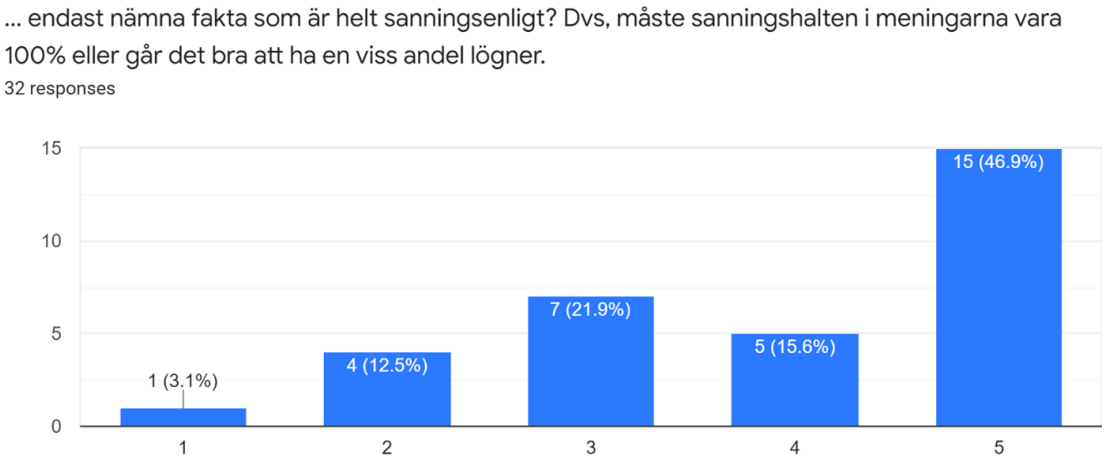

Figure 6.11: Question 11: how important is it to have only well-grounded facts? That is, only using truths.

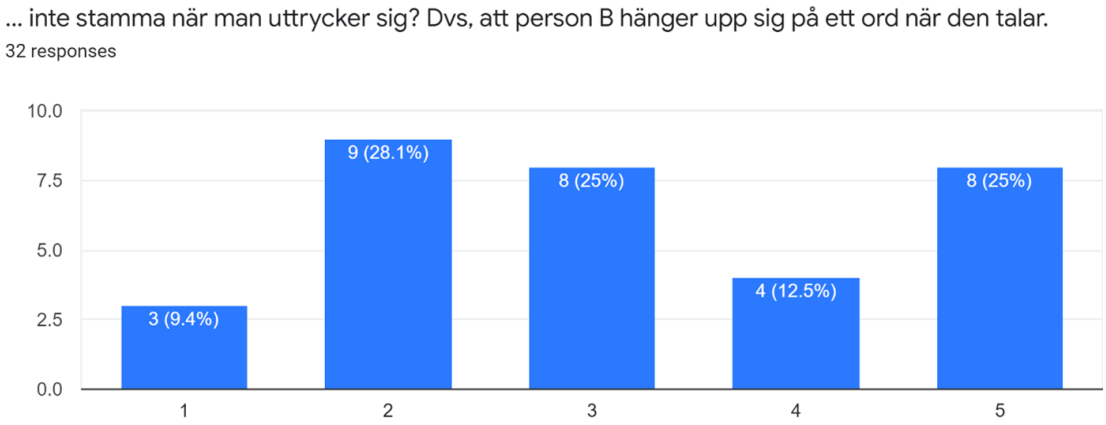

Figure 6.12: Question 12: how important is it that person B does not stutter?

Figure 6.13: Question 13: how important is it to not be repetitive about certain things?

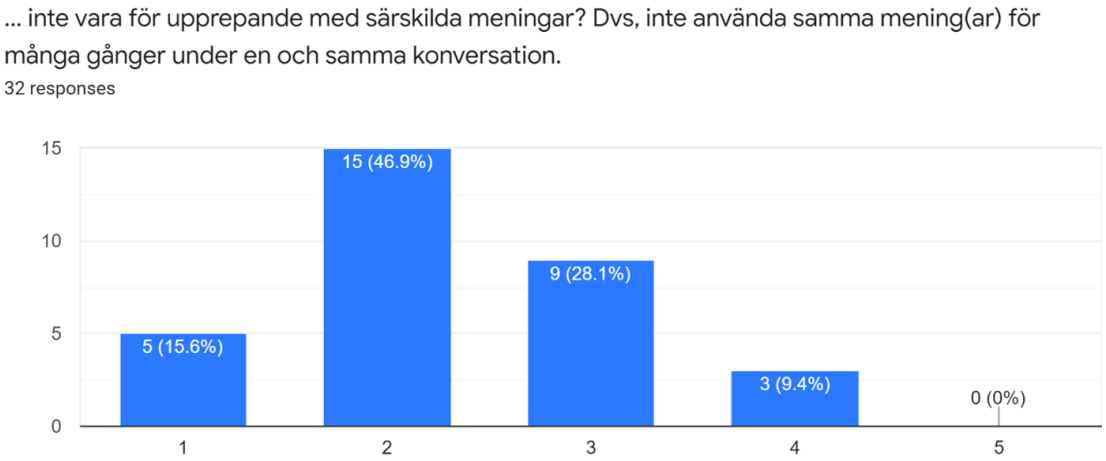

Figure 6.14: Question 14: how important is it to not be too repetitive with certain sentences?

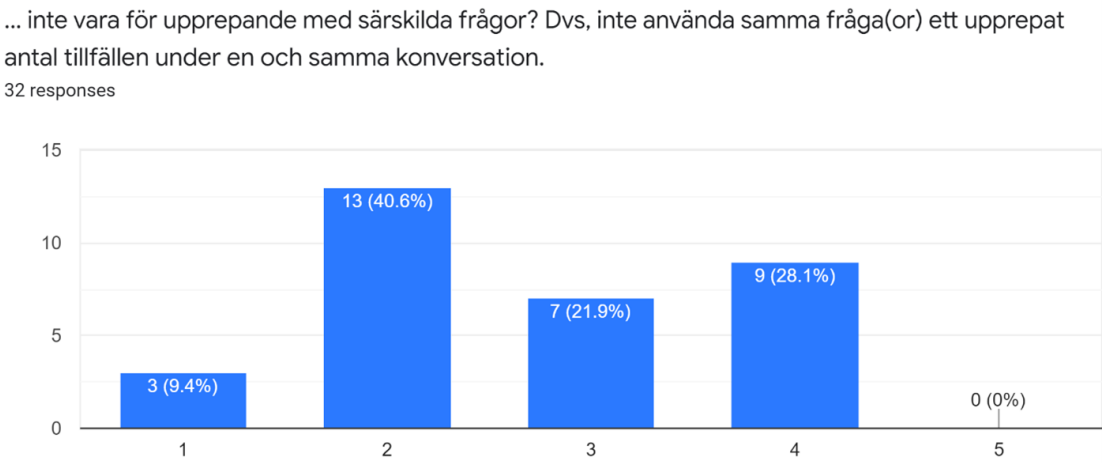

Figure 6.15: Question 15: how important is it to not be too repetitive with certain questions?

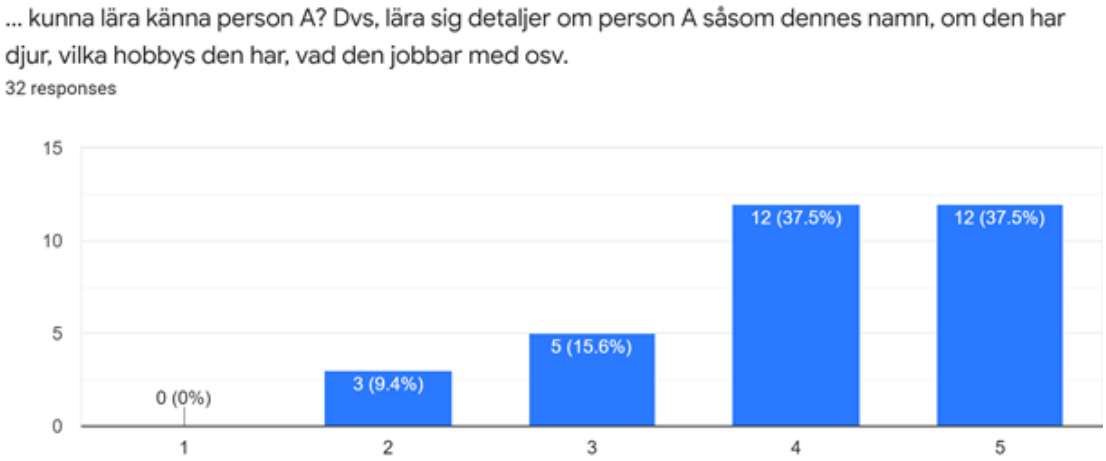

Figure 6.16: Question 16: how important is that person B is able to get to know person A? I.e. the ability to learn personal details about person A.

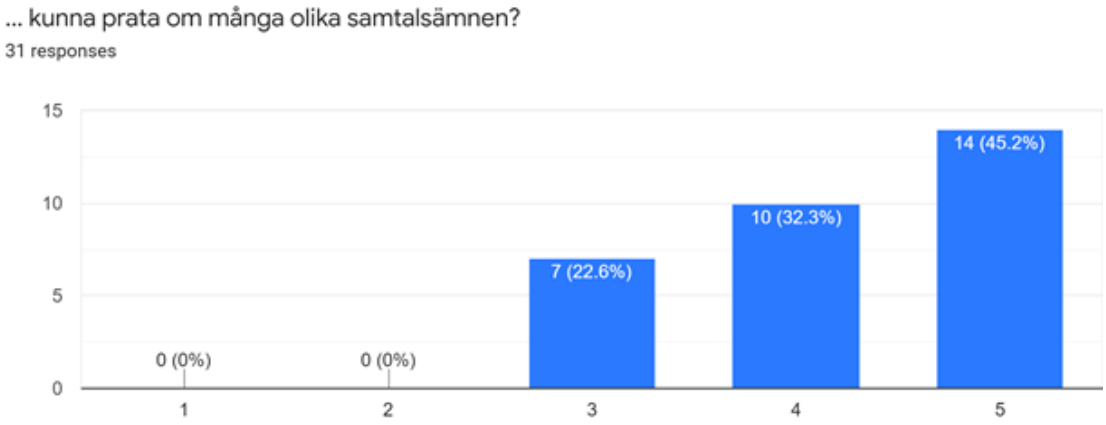

Figure 6.17: Question 17: how important is it to be able to speak about many different topics?

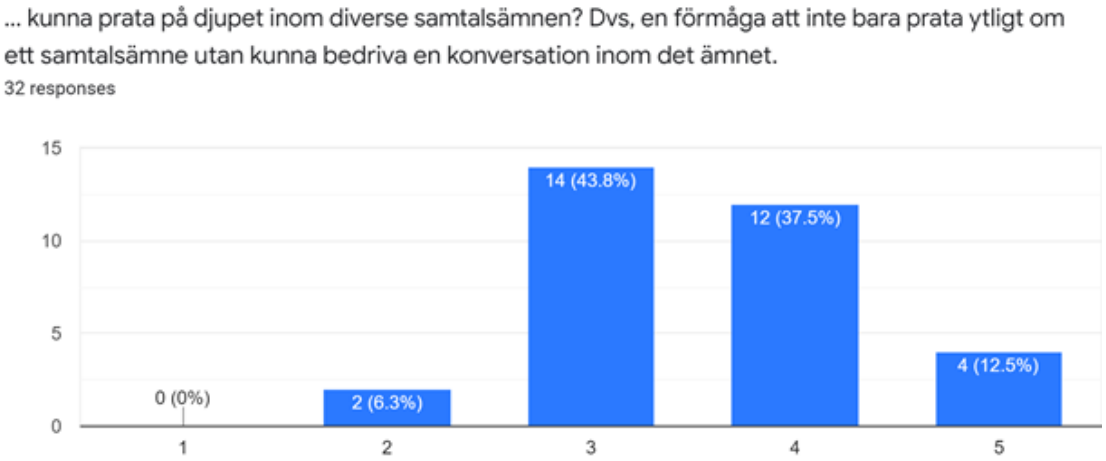

Figure 6.18: Question 18: how important is it to be able to speak in-depth within different topics?

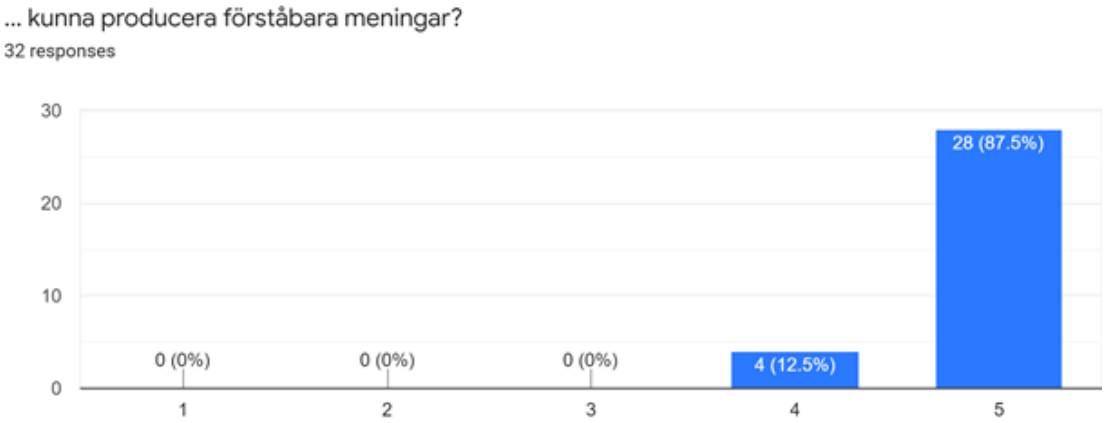

Figure 6.19: Question 19: how important is it to be able to produce understandable sentences?

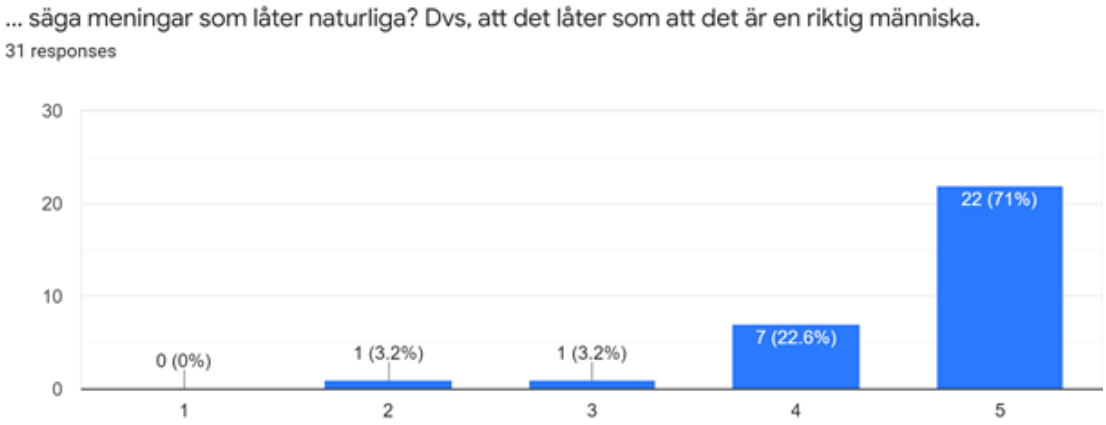

Figure 6.20: Question 20: how important is it to produce sentences that sound natural?

INSTITUTIONEN FÖR DATAVETENSKAP | LUNDS TEKNISKA HÖGSKOLA | PRESENTERAD 2022-06-03

EXAMENSARBETE Quality Assurance of Generative Dialogue Models Used for Language Practice
A Test Framework Used for Measuring the Quality of Generative Dialogue Models in an Automated Fashion
STUDENT Johan Bengtsson
HANDLEDARE Markus Borg (LTH), Alexander Hagelborn (NordAxon)
EXAMINATOR Emelie Engström (LTH)

Hur bra är dialogrobotar på att samtala och lära ut svenska?

POPULÄRVETENSKAPLIG SAMMANFATTNING **Johan Bengtsson**

För att lära sig ett språk krävs det bland annat att du talar språket mycket under en längre tid, men det är inte helt lätt att hitta någon att prata med. För detta ändamål utvecklar NordAxon dialogroboten Emely, som pratar svenska med användaren. Hos sådana produkter är det dock inte helt enkelt att mäta kvaliteten. För att bistå med det har vi utvecklat ett testramverk.

Vid flytt till ett nytt land står du inför olika utmaningar, och att kunna tala det lokala språket underlättar. Men att lära sig ett nytt språk är inte helt enkelt. Det är viktigt att tala språket mycket, något som kräver att man har en samtalspartner vilket inte är enkelt att hitta. För att förenkla detta så utvecklar NordAxon dialogroboten Emely, som svarar med människolika svar på det du säger. Men hur mäts kvaliteten på en sådan dialogrobot?

I detta examensarbete har vi utvecklat en metod för att utvärdera en sådan dialogrobot. Arbetet inleddes med att intervjua och skicka ut formulär till SFI-lärare och relevanta universitetsanställda, för att samla in deras syn på vad som kan vara viktigt vid utlärning av svenska. Resultatet visade att de viktigaste egenskaperna är att språket ska vara koherent, att det inte ska vara ovänligt och att nivån ska justeras efter språkstudenten. Emely har sedan analyserats gällande dessa för att kunna utvärdera kvaliteten.

Det utvecklade ramverket låter dialogroboten producera många konversationer. Därefter, för att analysera koherensen och nivån av ovänligt språk användes två olika maskininlärningsmodeller på varje meddelande. För att analysera nivån på språket användes frekvensordlistor och läsbarhetsindex, som tillsammans antas kunna visa vilken språklig nivå dialogroboten är på. Genom de genererade konversationerna får man en bild av vilket vokabulär dialogroboten har, och hur pass frekvent det är. Till det beräknades läsbarhetsindex för varje meddelande.

Resultaten presenteras i en dashboard som ska hjälpa NordAxon att veta hur bra deras produkt är. Resultaten tyder på att NordAxon lyckats göra Emely alltmer koherent, vänlig och till att oftare ha ett enklare språk. Allt som allt kommer detta bidra till bättre versioner av Emely, som i förlängningen kommer att hjälpa nyanlända till Sverige att snabbare integreras in i samhället.