Universiteit Leiden

Master Computer Science

[NAARA: Leveraging large language models to mitigate digital distractions and enhance focus through intelligent messaging response]

Name:	[Koen van der Burg]
Student ID:	[2714892]
Date:	[30/06/2024]
Specialisation:	[Bio-Informatics]
1st supervisor:	[Zhaochun, Ren]
2nd supervisor:	[Qinyu, Chen]

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Contents

Abstract

This study investigates how large language models (LLMs) can support a large dialogue system to manage daily digital communications. By handing over the responsibility of responding to incoming messages, it is possible to increase focus through reduction of distractions, whilst living up to unspoken availability expectations. It aims to do so by replacing the superusers (subconscious) need to check for notifications and (important) missed messages. Here, a Network Adaptive Artificial intelligence Response Assistant (NAARA) is developed using multiple LLM. The methodology involves two LLMs to perform four different tasks. NAARA classifies the urgency of the incoming message, it notifies the superuser if deemed important, it differentiates questions from statements, to decide if a response is required. NAARA also generates appropriate responses based on message context, sender relationship and relevant information extracted earlier from received messages. Results show that these models, even smaller-sized ones, effectively classify message urgency, discern questions, and extract key information. NAARA was tested using manually-curated data sets, containing a variety of potential incoming messages. During said experiments, NAARA demonstrated high accuracy in urgent message identification with increasing difficulties (100%, 92% and 71.25%) and reasonable performance in personal information extraction (91,64%, 85,14% and 67,5% of the maximal score). Response generation positively varied, with the sender's social group, and available knowledge. Base responses without additional info scored lower on average (3.9/5 points) in comparison to source enriched responses (4.42/ 4.94 points), with additional improvement seen if user related information is applied. NAARA's core operations run on a sufficient level to be applied for further testing with more realistic data. However, no claims about distraction reduction can be made without testing on a larger group of superusers. Adequate improvements are foreseen with respect to, urgency prediction with context tracking, refining prompts for better extraction and response generation, and developing modules to control information flow.

1 Introduction

Each decade improvements in technology are seen. Some operate under Moore's law, for example computing power is growing exponentially each year, having a profound impact on technologies such as artificial intelligence. As a side-effect, it changes our behaviour and interactions. Life changing events such as Covid-19 can cause accelerated technological advances within the society, forcing parts of the community to step into the digital world. Calling someone, conversing over text, video calling or utilizing a virtual reality headset, are all methods devised to communicate with one another. The positive development trend in communication has brought many benefits, for example speaking to a loved one at any time or location, or following lessons from your preferred workspace. This trend also created new hurdles for us to overcome, one of them being the constant expectation of availability.

With many available messaging services available to us (WhatsApp, Slack, Facebook, Discord), there is a constant stream of new messages and notifications. These various communication services, although unthinkable not to have, are a distraction for most people. Some people might be addicted to their phones, whereas others might fear missing an important message in a busy day. This is the juxtaposition where some workers or

students might find themselves in: they are expected not to be distracted by incoming messages, but there is unspoken expectancy or feeling they should respond if someone needs their help. Therefore, the question arose: how can we prevent distractions, notify the superuser of important messages and maximize our ability to focus on the work?
Various methods have been tried to prevent distraction. A "do not disturb" functionality blocks notifications, providing some protection against distractions for users. Another solution is an application to temporarily lock the device, until a time or goal condition has been passed, returning control to the owner. A more low-tech approach to the problem is storage of the device in an out-of-reach place, preventing the user from automatically grabbing and checking for new notifications. Most of these methods rely on complete shutout of the device in order for the user to regain control, whilst effective, it is not always an viable option. For many users, besides the non-urgent messages, there is still a significant number of messages that cannot wait for a response at the end of the day or tomorrow. A personal assistant is the last option, but this is costly.
Due to advances in the artificial intelligence (AI) field, interest in its applications grew, propelling the technology even faster ahead. This initial wave of new technology has sprouted many more branches, such as large language models (LLMs), which made NAARA possible. NAARA (Network Adaptive AI Response Assistant) is a virtual assistant tasked with supporting the user with their messaging interactions. This system should allow the user to focus on their work, only receiving a message if it is deemed important enough to let through. It attempts to do so by leveraging the potential of LLMs for various functionalities. Therefore, this study is going to explore if LLMs are capable of: discerning urgency, classifying questions from statements, extracting useful information, as well as providing a response to non-urgent messages that feel human like and engaging, in a singular workflow.
The distraction solutions mentioned before are only capable of stopping the incoming communications. Besides determining to notify the user, responses are provided to messages that do not need the user's immediate attention. This is done to gather extra information about the received message, or to let the other person know you have seen the message and will come back to them. For example, a friend or family member wonders if the superuser has time to go for lunch next week, the assistant can gather information about a specific time and place, meanwhile, the superuser continues working undisturbed. Or, the system knows the user is busy next week thus notifies the other person of the user availability. Furthermore, responses are altered based on the social group of the message sender, providing a professional response to colleagues, and a more sociable response to friends. Superusers provide information about their relation to the new user, allowing NAARA to provide alternative more informed responses, based to the relation network that is build up over time. To improve responses, the network can be extended with any relevant information, acquired via message analysis. NAARA is located between the messaging services and the superuser, acting as a semipermeable bubble. It keeps out non urgent messages, whilst permeating important messages to the superuser.
Before going into more detail of NAARA's functionality, relevant previous work on this subject will be discussed in section (2). Followed by a detailed methodology (3), which provides an overview how the different parts function. Chapter (4) will provide the experimental setup as well as the results of NAARA's performance. The final two chapters are used to discuss the results found during the experiments (5) and to conclude the effectiveness of NAARA, as well as the possibilities in the future (6).

2 Literature review

Dialogue systems or conversational agents are computer systems capable of conversing with humans. These systems are often classified as Task-oriented Dialogue System (TOD), Open-Domain Dialogue System (ODD) or both, as seen in sections 2.3 and 2.4.
TODs are carefully and precisely designed to manage conversations centered on specific tasks. They adeptly guide users towards achieving objectives by identifying user intentions, maintaining the dialogue state, executing appropriate actions, and generating relevant responses. These systems are engineered to ensure that interactions are efficient and goal-directed, optimizing the user's journey towards task completion, an assistant.
In contrast, open-domain dialogue systems (ODDs) are crafted for more expansive and unstructured interactions. These systems facilitate dynamic, free-flowing conversations on a broad array of topics. Rather than being constrained by predefined tasks or objectives, ODDs generate responses by directly mapping the dialogue context, allowing for a more natural and unrestricted exchange of information. This flexibility makes them suitable for applications requiring broad conversational abilities and adaptability to diverse user inputs.
In the beginning, multiple attempts and iterations have been made separately on both approaches (TOD & ODD), due to the different nature of both systems. Over time, attempts have also been (successfully) made to combine the two systems. The evolution of language model-based dialogue systems can be split into four different phases; early stage, TOD and ODD, system fusion and the large language model-based phase, in which we find ourselves at this moment. Each phase, although not rigidly distinct, can be signified by an important development that has brought the technology to where it is now.

2.1 Early stage - Statistical language models (SLM)

The development of dialogue systems began with the utilization of statistical language models (SLM) and rule-based approaches. These early systems were primarily constructed using hand-crafted rules and simple statistical methods, laying the foundational groundwork for the field of conversational agents.
One of the earliest and most notable examples of a rule-based dialogue system is ELIZA [9], proposed by MIT in 1966. ELIZA was designed to simulate conversation through pattern matching and substitution, often without understanding the semantics of the dialogue. It employed a set of pre-defined scripts that mapped user inputs to responses, creating an illusion of understanding. Despite its limitations, ELIZA demonstrated the potential for human-computer conversations, exciting the development of more sophisticated dialogue systems.
The emergence of statistical methods marked a significant shift from purely rule-based systems. These methods leveraged machine learning techniques that utilized probabilistic models to predict the next word in a sequence. Early statistical models, such as n-grams, calculated the probability of a word based on the preceding $n-1$ words. This approach allowed for more varied and contextually appropriate responses compared to a more rigid rule-based system. However, the limited context window of n-grams meant that these models still struggled with generating coherent long-term dialogues.
In the early 2010s, dialogue systems displayed substantial advancements with the introduction of more powerful technologies. IBM's Watson [12], which gained fame in 2011

for its performance on the quiz show Jeopardy!, represented a significant leap in the capabilities of dialogue systems. Watson utilized a combination of natural language processing, information retrieval, and machine learning to understand and respond to a wide range of questions with high accuracy. Following closely was Apple's Siri [24], launched in 2012, which brought dialogue systems into the mainstream as a personal assistant on smartphones. Siri combined speech recognition, natural language understanding, and a rule-based back-end to perform a variety of tasks based on user queries. This marked a pivotal moment, not only large companies with supercomputers have access to a virtual personal assistant.
Despite these advancements, early dialogue systems were predominantly task-oriented, designed to follow unyielding processes and perform specific functions. Their functionalities were limited by their reliance on predefined rules and statistical methods that could not yet fully capture the complexities of natural human conversation. Despite these limitations, the early stage of dialogue system development set the stage for more sophisticated and flexible systems. This paved the way for the subsequent phases of evolution in this field.

2.2 Independent development of TOD and ODD - Neural language models (NLM)

The second phase in the evolution of dialogue systems was marked by the independent development of task-oriented dialogue systems (TOD) and open-domain dialogue systems (ODD) using neural language models (NLM) [6]. A shift from traditional statistical and rule-based methods to more advanced neural network-based approaches, significantly enhanced the capabilities of dialogue systems.
A pivotal development during this phase was the sequence-to-sequence (seq2seq) framework in 2015 [25]. The seq2seq model introduced a novel architecture for handling variable-length input and output sequences, by putting two (recurrent) neural networks together. Creating an encoder to process the entire input sequence into a context vector, and a decoder to generate the output sequence, allowing flexible in and output. This architecture became a cornerstone for both TOD and ODD systems, providing a flexible and powerful method for modelling dialogue, which was a major advancement over previous models.
Neural networks, particularly recurrent neural networks (RNN) [5], Long Short-Term Memory (LSTM) networks [11], and Gated Recurrent Units (GRU) [3], were instrumental in this phase. These models could characterize word sequence probabilities by capturing long-range dependencies within text, a significant improvement over the short context windows of earlier statistical models. The ability of LSTM and GRU to manage long-term dependencies made them especially suitable for dialogue tasks, where context and coherence are crucial.
Initially, the focus of NLM shifted towards open-domain dialogue systems due to the need for handling a wide range of conversational topics. ODD systems leveraged the seq2seq framework to generate dynamic and engaging conversations, significantly advancing the state of the art in this domain. However, the same neural architectures were also applied to task-oriented dialogue systems. By incorporating domain-specific knowledge and extensive feature engineering, these systems could better understand user intents and provide precise, goal-oriented responses.
Despite these advancements, the development of NLM-based dialogue systems required

extensive hand-crafted feature selection and domain-specific knowledge. This necessity often limited the scalability and adaptability of the systems, as each new application or domain required significant manual effort to configure and optimize the models. With an increasing number of different domains and applications, it is nearly impossible to keep up crafting custom models.
The integration of neural language models in the development of both TOD and ODD systems represented a significant leap forward in the field of dialogue systems. By leveraging the strengths of neural architectures, these systems achieved greater flexibility, coherence, and conversational depth, compared to the early-stage models.

2.3 Fusion of Dialogue Systems - pre-trained language models

The third transformative evolution of dialogue systems was the integration of pre-trained language models (PLM), which brought unprecedented advancements in the field. These models, such as BERT (Bidirectional Encoder Representations from Transformers) [8] and GPT-2 (Generative Pre-Trained Transformer) [22], were trained on extensive and diverse language corpora, enabling them to capture rich semantic and syntactic patterns before being fine-tuned on specific tasks. BERT is trained with masked language modeling and next-sentence prediction objectives, making it efficient at predicting masked tokens and natural language understanding (NLU), however, was not optimized for text generation. GPT-2, created by OpenAI, is a large deep learning transformer-based model, trained with a casual language modeling objective, thus optimized for the next token prediction for a sequence. The model is trained on over 1.5 billion parameters to generate the next sequence of text for a given sequence. Due to the diversity of the dataset during the training phase, GPT2 is capable of adequate text generation, for a variety of domains.
The introduction of PLM created a new paradigm in dialogue system development: the pre-train and fine-tune approach. In this framework, models undergo extensive pre-training on large-scale datasets to learn a wide array of linguistic features and general knowledge. This pre-training phase endows the models with a robust understanding of language, making them versatile and adaptable to various dialogue contexts. Subsequently, these pre-trained models are fine-tuned on specific dialogue datasets to optimize their performance for applications, whether they are task-oriented or open-domain dialogues.
One of the most significant advancements during this phase was the development of unified dialogue systems (UniDS) [27]. These systems aimed to merge the functionalities of both task-oriented dialogue systems (TOD) and open-domain dialogue systems (ODD) into a single, cohesive framework. The pre-trained models facilitated this fusion by leveraging their extensive pre-training knowledge, which allowed them to handle diverse dialogue scenarios effectively. For instance, a unified model could be fine-tuned to manage specific tasks, such as booking a flight or providing customer support, while also maintaining the ability to engage in open-ended conversations on a variety of topics. The combination provides a more human-to-human experience, through continued, natural-flowing conversations.
This fusion was not merely theoretical but was practically demonstrated by several systems that utilized PLM to achieve remarkable versatility and performance. The ability to integrate task-specific knowledge with general conversational capabilities made these systems highly effective and efficient. They could switch seamlessly between task-oriented dialogues, which require precision and goal-directed interactions, and open-domain dia-

logues, which demand flexibility and adaptability.
The period marked by the adoption of pre-trained language models represented a significant leap forward in the field of dialogue systems. By harnessing the power of deep learning and large-scale language corpora, these systems achieved a level of sophistication and coherence previously unattainable. The pre-train and fine-tune paradigm, coupled with the fusion of TOD and ODD into unified dialogue systems, not only set a new standard for the development and deployment of conversational agents, but also paved the way for even more advanced and integrated solutions in the future.

2.4 Large language model based dialogue system

The latest phase in the evolution of dialogue systems is characterized by the deployment of large language models (LLMs). Well known examples are the Generative Pre-trained Transformer (GPT) models developed by openAI, such as GPT-3 [2], InstructGPT [20], and the most familiar GPT powered chatbot, ChatGPT [13]. Besides openAI, other companies such as Meta have joined the latest phase with their new Llama3 model [15]. These models represent a significant leap forward, scaling up both the size of the models and the breadth of the pre-training corpus to capture more intricate patterns and representations. This enhanced capacity makes LLMs more sample efficient, allowing them to learn complex linguistic nuances and generate high-quality responses across a wide array of conversational contexts.
LLMs benefit greatly from the pre-training, fine-tuning paradigm developed in the previous phase 2.3. Foundational knowledge is refined through fine-tuning with in-domain data, such as specific dialogue or conversational corpora, to enhance their conversational abilities, comparable to the earlier phase models (BERT, GPT2). This approach has been exemplified by the evolution from GPT-3 to InstructGPT and finally to ChatGPT, each iteration showing marked improvements in performance and usability. GPT2 was trained on a stunning 1.5 billion parameters, and the new generation (GPT3) is trained on a staggering 175 billion parameters, resulting in a better NLU understanding of the model. One of the most notable features of LLM-based dialogue systems is their unprecedented capability to handle a wide range of language understanding and generation tasks. These include question answering, named entity recognition, and the generation of coherent, contextually appropriate responses. The comprehensive pre-training allows these models to perform well in both task-oriented and open-domain dialogues without requiring significant modifications, making them versatile tools for various applications.
GPT3, although very powerful, had issues with user-alignment. Meaning, the model was trained to predict the next word on a large dataset of internet text, rather than to safely perform the language task that the user wants [1]. This caused the model to generate outputs that would be considered untruthful, toxic or contain harmful sentiments. To evolve from this, the integration of instruction tuning and reinforcement learning from human feedback (RLHF) was applied, to further align the outputs of these models with human preferences and values. Instruction (fine-)tuning involves training the models on datasets that exemplify desired behaviors, effectively teaching them how to respond in ways that are most helpful and appropriate. RLHF, on the other hand, employs feedback from human evaluators or automated systems to continually refine the models performance. The iterative process ensures that the models generate more accurate, relevant, and human-like responses over time.

The combination of large-scale pre-training, targeted fine-tuning, and RLHF has revolutionized the capabilities of dialogue systems. LLM-based systems now exhibit a high degree of sophistication, enabling more seamless and intelligent human-computer interactions. This latest phase in the development of dialogue systems not only demonstrates the power of advanced machine-learning techniques, but it also allows ideas and applications to ride the innovation wave.

2.5 The latest phase of applications

The advancements made in the latest phase of dialogue systems, LLMs, have led to a broad array of applications that leverage the sophisticated capabilities of these models. Utilizing the foundation created by the latest phase, these applications demonstrate the practical impact and versatility of LLM-based systems in various domains. Some of the domains LLMs can be used in are translation, content generation, code development, Q&A, education and virtual assistance. Due to the nature of this project (and the endless possible ideas), the primary focus of the coming subsection will be on the applications within the virtual assistance domain.

One of the primary applications of LLM-based virtual assistance is API interaction, to facilitate seamless integration with various digital services. For instance, LLMs can automate tasks, set up meetings, order groceries and send messages [23]. By understanding natural language inputs and generating appropriate responses, these systems streamline user interactions with digital platforms, enhancing productivity and convenience.

Another significant application is information provision through dialogue. LLMs are adept at understanding and generating natural language, making them ideal for delivering information conversationally. This capability is particularly useful in contexts where users seek quick and accurate answers to their queries, ranging from general knowledge to specialized information. The ability of LLMs to engage in natural, coherent dialogues ensures that users receive information in a more intuitive and accessible way, it is easier to understand someone speaking the same language.

In the realm of customer support, LLM-based dialogue systems have shown considerable promise. These models can handle a variety of tasks, such as answering customer complaints and responding to reviews. Automating the complaint department could save a company an enormous amount of time and workforce. While LLMs are powerful tools, they are often used in conjunction with strict data privacy protocols to ensure that no personal information is processed. This application highlights the importance of balancing advanced technological capabilities with ethical considerations and privacy safeguards. Furthermore, the response of LLMs is sometimes still too divergent from the wanted sequence, holding back companies from fully integrating this into their system.

2.6 Fundamental differences

The applications from section 2.5 illustrate the practical benefits of LLMs in enhancing user experiences, improving operational efficiencies, and enabling more intelligent and responsive interactions in both personal and professional contexts. Despite the existence of these various applications, there was no system that helps the user focus on their work, by relieving them from the societal pressure to respond to incoming message stream. This research, explores the possibility of managing message streams and generating personalized

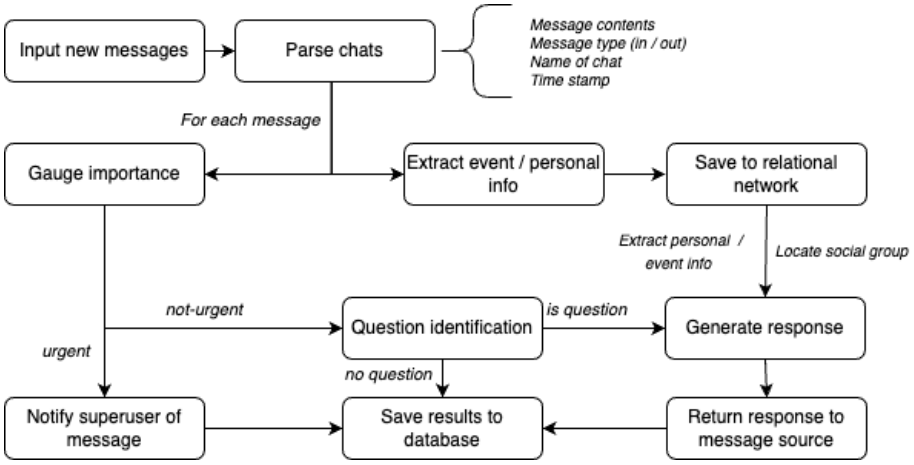


Figure 1: *This diagram provides a high-level overview of the workflow that exists in NAARA. Beginning by retrieving a collection of messages, always ending by saving all processed and produced data.*

responses.
NAARA, an innovative application based on an LLM-based dialogue system, is an system designed to manage message streams. It automatically filters incoming messages, assesses their importance, and provides responses to non-urgent inquiries. Additionally, it saves and analyzes information from messages, integrating this data into a relational network to improve future responses.

3 Method

To understand how NAARA is build, the program will be dissected into it's four main processes: data collection, large language models, relational network and database. Data collection or data import, the entry point of the program, supplying a stream of (new) messages to process them, is explained in section 3.1. Although the system is designed to handle various input sources (if it is reduced to a standard format), the current state of the program is able to read and process web WhatsApp traffic. Section 3.2 explains the processing capabilities of LLM, and discuss how to leverage these models provides a strong backbone for handling messages in the system. The relational network will be explained in section 3.3, saving personal and event information from analyzed messages, providing context for a more accurate and personal response.
Finally, a short section about database handling (3.4), which is currently done locally, but is required to have in order to for NAARA to function properly. Figure 1 provides an overview of the various themes that are involved in the whole process. From this section on a distinction is made between superuser (person A, using NAARA) and user (person B that messages person A).

3.1 Data Input

The entry point of NAARA is the incoming message stream. The messages can be sourced from a variety of applications; Slack, WhatsApp, Facebook Discord or WeChat [18], as long as the incoming data is parsed to a standard format. The format contains at least

one field: the user message contents, optionally supplemented with 4 other fields: message type (incoming / outgoing), chat name, reply to and a time stamp. If the source data complies with this (fairly light) constraint, the program is able to process the input, making it versatile for various endpoints. Message content is required for the most basic functionality, but to fully optimize the results, the other fields should be supplemented as much as possible. Figure 2 provides an overview of the data input process.
This study used WebWhatsApp (the web version of WhatsApp) as the data source, which due to privacy reasons, has no API to retrieve messages in a standard format. WhatsApp uses end-to-end encryption to keep message contents safe from outsiders, hackers and even WhatsApp owners themselves, except for the users connected to the chat [10]. The Selenium and Chromedriver packages enables the system to locate and store message interactions from a personal WhatsApp profile [26] [4]. It does so by opening a Chrome window on screen using saved login details, opening each chat one by one, and saving specific HTML data, identified by div variable names. The message type is simultaneously parsed from the fetched data, distinguishing incoming from outgoing messages (both user and superuser data is retrieved). A chat name provides a handle for NAARA, allowing social group retrieval, as well as storing or retrieving personal information. Storing personal information allows NAARA to provide a more precise response, explained in more detail in section 3.3. If a message is a reply to and older message, the older message is saved with the new data entry, to use for context in response generation.
After parsing all the incoming messages, old messages are separated from the new ones via a database (3.4), to prevent unnecessary calls to the LLM, since each (unnecessary) call increases processing time. Message contents are also translated from the original language, via language detection, to English using deep translator [7]. Due to differences in training data availability, linguistic structures, cultural nuances, tokenization methods, and potential biases, interactions of large language models can change based on the language. Although the English and Dutch language differ less in comparison to other languages, translating the message could prevent unforeseen issues.
Despite NAARA's capability to receive input from WhatsApp, for the experiments the decision was made to test the capabilities on manually-curated data sets. Allowing better control on the flow of incoming data to NAARA, testing NAARA's current limits.

3.2 Large Language Model

The backbone of NAARA, large language models, carry out a lot of the heavy lifting in four phases of the program: importance gauging, question identification, response generation and information extraction. Since the performance of LLMs for these specific uses is unknown, they will all be subjected to testing in the experiments section 4.
Each message is subjected to multiple LLM calls, adding up processing time when encountering large batches of messages. Therefore, it is easier to utilize an offline model in comparison to API calls to online endpoints such as Chat-GPT. An offline model provides consistently low response times, no downtime, and zero costs besides the costs for running a personal device. Online models have a set token limit, or other user restrictions, if no active subscription is provided. To facilitate an offline model a service named Ollama is used, specialized in running offline LLM on local devices, simultaneously it also provides ready to download models and a python package to interact with said model [19] [21]. From the selection Ollama offers, the Mistral (version: instruct fine-training) and Meta

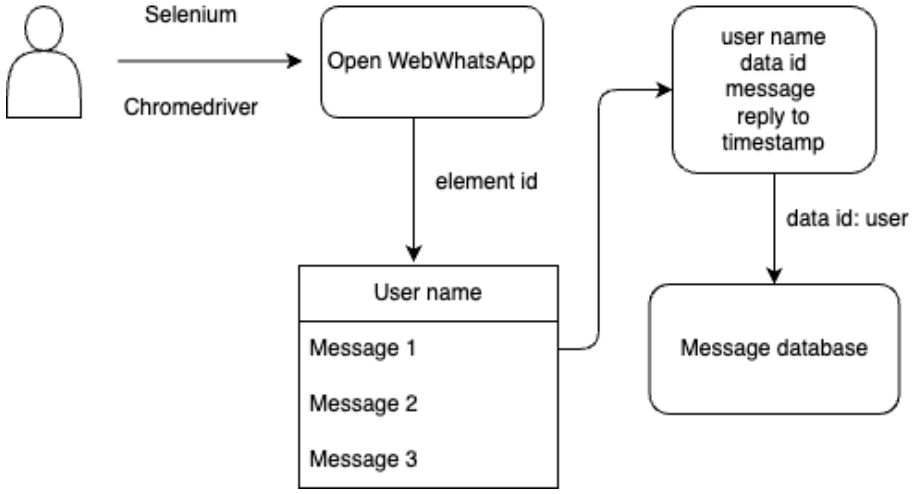
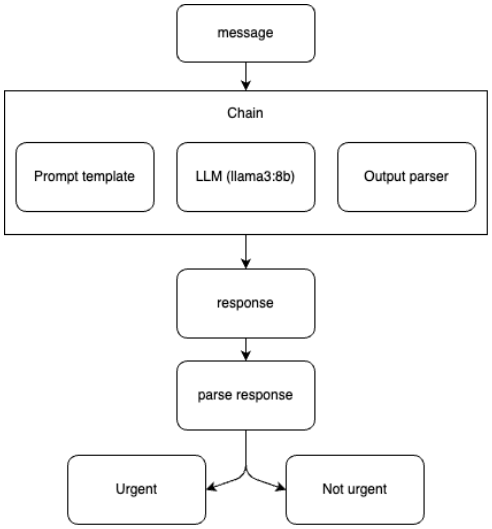
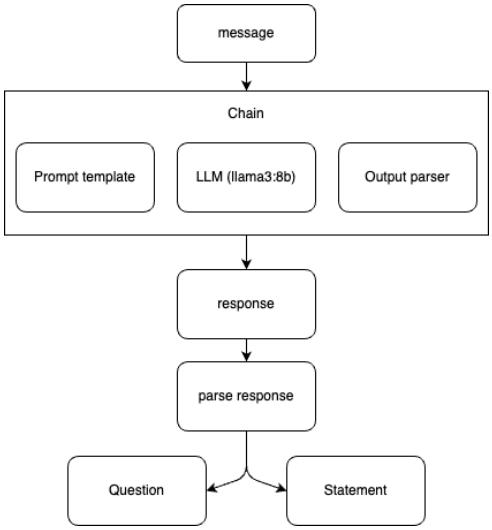


Figure 2: *This diagram provides a detailed overview of the WebWhatsApp data input process. By using element id's to locate the message data, and user id's to identify the origin of the message. The message data is located from the superuser's chats, and saved to the database if the message originates from a user.*

Llama3 (version: 8 billion parameters) model were selected [14] [17] [16]. Both models come in varying sizes (4-32 GB), and the smaller sized versions were chosen, to reduce processing time, since larger models require more computational power to run. The former is due to its interaction with task and instruction prompts and the latter mainly for its capabilities in text generation. Fine-tuning a model allows it to perform better in a specific area of expertise, in this case on tasks and instructions.
An initial call on the Llama3 model is used to gauge the importance of a message, which decides if the superuser needs to be alerted of the newly received message. Since urgency is subjective, it could be trained over time to adapt to the exact needs of each new superuser, however, a more general approach works out of the box. To generalize urgency, instructions were provided, in the form of a prompt, to help understand what it means to the superuser. It tries to describe the notion by framing urgency as a message that requires an answer within 24 hours, it should be recognized as important. Furthermore, it is specified that if someone is requesting for immediate help, it is also regarded as urgent. This means that; "Could you send those files today?", "Can you help him finish this bug as soon as possible?" or "What do you want to eat tonight?", are seen as important. One could argue that the latter is not important, however, the user is expecting an answer today, thus for the answer to still be valid it requires a same-day response from the superuser. There is no control over the response time of the super-user, but at least they are notified of the situation. Non-urgent messages could range from; "Are you free for a coffee next week?" to "Have you seen the new movie?" or "Yes that sounds like a good idea!". The response of the model is turned into a boolean via post-processing, the model will not always precisely return as expected. The precise prompt is provided in the Appendix (A.2.1), and an overview of the process is provided in figure 3a.
The second application of the Llama3 model is for response generation, since it's text-generation is more proficient in comparison to the Mistral model. Messages from the user are aggregated when sent in succession, allowing NAARA to respond to the messages as one. Sending messages in the same time frame often suggests that these messages



(a) *This diagram details the urgency classification process within the larger message workflow. A chain is created using the crafted prompt, the Llama3:8b model and an output parser, to process the message. Afterwards extra parsing is needed to create a boolean from the chain result.*



(b) *The workflow of question classification is similar to that of the urgency classification 3a. By providing a different prompt template and parser designed specifically for question classification, the message can be classified either as a question or a statement.*

Figure 3: Structure for LLM classification requests.

are connected and, therefore best be treated as one incoming message. If a message is classified as unimportant, the super user will be undisturbed, remaining focused on their own task. However, some (deemed as unimportant) messages could still be interacted with to gather information, therefore messages containing questions will receive a generated response. Someone might ask to meet up, a response could include a question for details about the meeting; when or where is this possible future event taking place? This relieves some of the superuser's time, NAARA has already gathered more information about the event. When asked about finishing up a specific task, it can let the recipient know that the superuser is busy and and will provide a more detailed response later. This provides the superuser breathing time when busy, whilst simultaneously updating the recipient user about the superuser's status. Personal information from both the user and superuser is supplemented in the response call, providing extra context for the model. The user could have associated data about their likes and dislikes or discussed events and the superuser can provide any data about themselves or their schedule. The system considers if the user is responding to a message the superuser send. For example, the superuser sends a message to a user, which prompts the user to respond with a question, then it its contextually informing for NAARA to integrate knowledge about the reply, for its own response generation. Figure 4 details the process of generating a response.
So far, response generation has been primarily discussed for 1-on-1 chats, where the super user is in contact with a singular person. Group chats, behave differently, and the assistant should adapt its behaviour along side with it. Incoming messages in a group chat are usually not directed at one person and can therefore often be classified as not urgent. Instead of considering if the message is a question (does not necessarily implicate

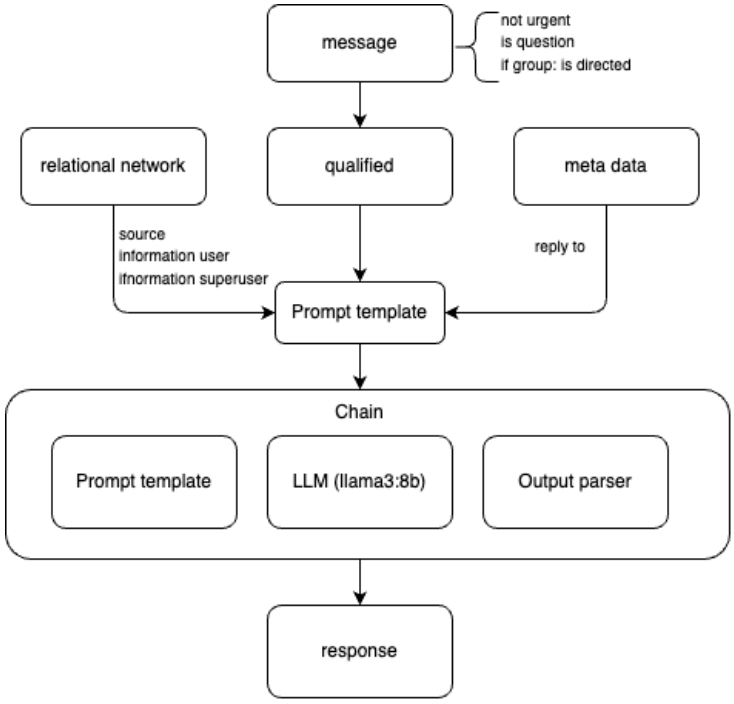


Figure 4: *This diagram provides a detailed overview of the response generation process. Messages need to pass certain conditions to qualify for a response. Contextual data is provided from the database and relational network, and injected into the prompt template. Executing the chain returns a response for the received message.*

the involvement of superuser), group chat messages are checked for directedness. Directness is determined by analyzing the message for the superuser's name. Any (non-urgent, question-containing) messages, directly addressing the superuser, are assumed to involve the superuser, and thus will prompt a response from NAARA.
To catch all sentences containing questions (for response generation), the Llama3:8b model is used in combination with its own custom prompt A.2.2. Simply checking for a question mark does not suffice, since humans can be messy and structure questions, without using the right punctuation. It is therefore also instructed to check for the use of question words such as; "Who, what, where, when, why and how", to classify the sentence. Figure 3b displays the question classification module.

3.3 Relational Network

The final step of the application is information extraction through message analysis. Involving a two part system, firstly retrieving significant information from messages using a LLM chain, and secondly storing said information to a graph-based network. The relational network encompasses the superuser, it's social groups, their members, and the personal information of individual users. A visual example of said network is shown in figure 5.
Manually, contacts can be added via NAARA, declaring a name and the social group they belong to (friend, family, work or other). Once added to the network, NAARA will analyse their incoming messages and try to extract relevant information about the person or events surrounding said person. For example, likes and dislikes, or an appointment

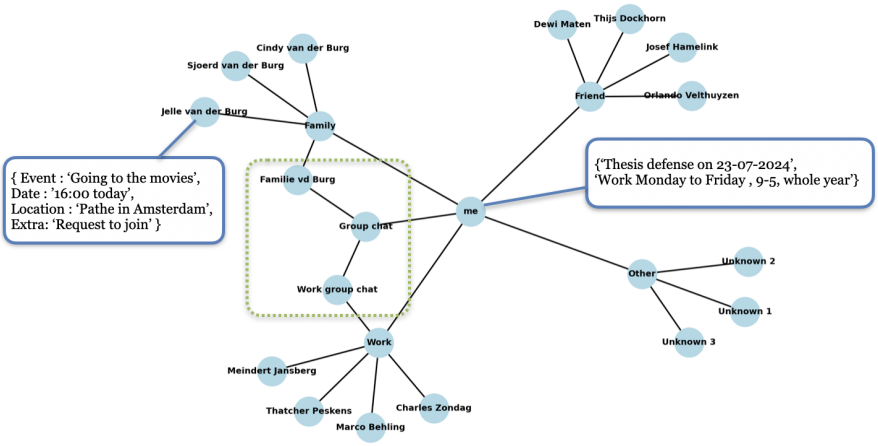


Figure 5: *This figure provides an example of a possible relational network. Where the superuser ("Me") is in the centre, surrounded by different social groups, and their members. Information about users and the superuser is saved to, and pulled from, this network. It also allows to keep track of group chats and their social group.*

they set later in the week, will be added to the information node. The network allows the superuser to control the information in the network, via manual removal or addition. Information about the superuser, such as their availability, is also saved to said network. Information extraction is made possible via a fourth LLM request, using a designed prompt (A.2.4). The model is requested to analyse the message and locate four different knowledge fields; 'event', 'location', 'date' and 'extra information'. Event is meant to describe the subject in the message, such as 'upcoming deadline' or 'going to the movies'. The date could be any time frame found, as well as a time, or indication of. The location is reserved for the space where the activity takes place, if available. Lastly, any information found not belonging in the fields mentioned before, such as personal details or information, are subjected to the extra information field. The prompt contains a preferred output format, so the LLM response (string) can be parsed into a usable python object (dictionary). The created dictionary of extracted data is added to the corresponding user in the relational network, so it can be retrieved for response generation.

The previous sectioned mentioned the use of supplemental knowledge to improve the response generation. By using the relational network, names can be linked to social groups, and for each social group an explicit prompt is generated, provided in the appendix (A.2.3). The social group of the person is retrieved (if not specified it is set to 'unknown'), which determines the professionality and friendliness of the response. If NAARA responds to a friend, it will act more fun, friendly and relaxed, in comparison to a response to a work contact, where the reply should be professional. Responding to family messages should be casual but stay polite since family members tend to fall in a wide age range. With unknown contacts, NAARA will respond in a neutral and polite matter, since it is not aware of the social group the new contact belongs to, or that the superuser is aware of their existence. No personal details from the superuser will be shared with unknown contacts, and only availability related information to colleagues.

Besides providing context about the social group, approved network information from the users and superuser's node is supplied from the network to the response prompt

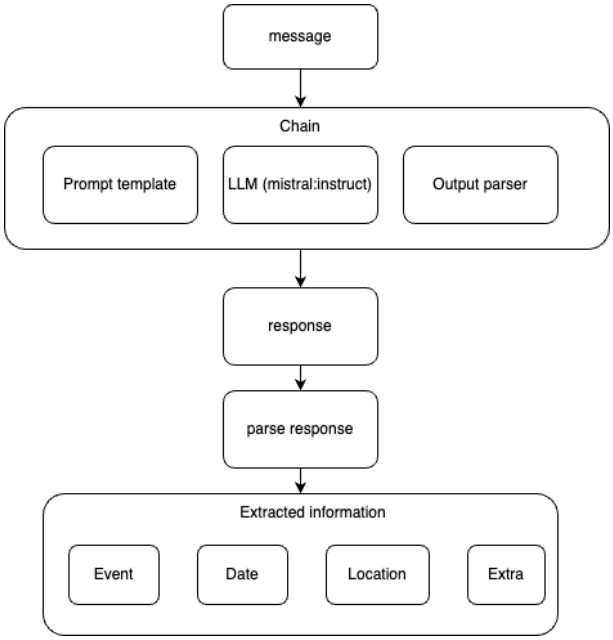


Figure 6: *This diagram provides an overview of the information extraction module. The response from the mistral model is parsed into four fields (event, location, date, extra) and saved to the relational network.*

when requested. For example, your friend shared earlier this week that he or she has an important event in Amsterdam on Friday. Through propagation of this information to the network earlier this week, the system understands what event is discussed, when discussed up later. This feature enhances the quality of response, as well as keeping the continuity of the conversation intact. Furthermore, it ensures that important messages are prioritized and addressed promptly, when known a certain event or activity is of importance. The personal schedule of the superuser can also be used by NAARA, providing an indication of availability in responses to users.

3.4 Database

To keep track of messages a local offline file-storing database and SQLite database with Django REST-framework was integrated into the system. This particular combination is selected for a variety of reasons: relatively easy set-up, SQL-based, python compatible and easy storing/retrieving. To set up a back-end database can be tedious, however, the Django REST-framework provides a setup guide, providing most of the basics needed. SQL was chosen based on the easy and quick data retrieval / manipulation, as well as the cost-effectiveness, making the system prepared for scalability. Django is python-based, thus integrated right into the system without any changes. Lastly, the database is mainly used for efficiently storing and retrieving batches of messages and meta-data, for which SQL is optimized.
When processing a message, the SQL database is activated, via provided login details, to retrieve and save data. The new message details can be checked against those in the database, indicating if the message is new, or old and does not need processing. Once a message has been processed, it must be saved to the database alongside its metadata. The metadata consists of the following elements: original message, response, urgency, extracted

info, NAARA version, LLM used, question identification, social group, possible reply to and the timestamp. Fields like chat name, message and the time are of importance to check whether an incoming batch of messages is new, or already existing in the database. Saving the importance of the message allows for filtering if needed. For example, to separate the urgent from the non-urgent messages. Other metadata such as the version of NAARA or the LLM helps distinguish differences in performance between different said versions. The only required fields are the chat name and the message received, keeping requirements to a minimum expands the acceptance range of NAARA.
A second viable option is a local file storing database, which consists of saving and loading CSV files to folders on the local machine. Similar to the first option it contains the same fields, however, the storing format is less efficient. If changes are made to the storing format, or others try to access the database, issues will arise. However, this option does provide easy access to the database via the integrated development environment or excel viewer, provided that back-end interactions are prevented. But, over time this storage system will reduce the speed of the entire program; data storage will increase, and less efficient operations can be performed. It uses a triple CSV file system to save the data; one for the latest messages, one for the urgent messages and one parent file. This parent file holds all the messages but does not save duplicates. The system replaces the old messages with their newer, not yet seen ones, in a last messages database. A new message means that there is no response provided to it yet. Thus, if the system has received a message and saved it to the database, and receives a new one from the same contact, it needs the parent file system to realize it is a second new message. Urgent messages are stored in their own separate database, sending them over bunched as one at the requested time, such as a break or end of work moment.

4 Experiments and Results

NAARA represents a streamlined workflow utilizing a collection of specially designed prompts to optimize response generation. This application incorporates mechanisms to assess urgency, extract information, and differentiate between questions and statements. Each component is evaluated individually to determine its efficacy within the overall framework. Furthermore, NAARA leverages the gathered and supplemented information to generate the most contextually-accurate personalized responses as possible. Consequently, it is essential to test response quality under varying contexts and information availability.
Importantly, there is no standard data set to test the interactions we are interested in. Therefore, the following results are based on manually-curated data sets. The data sets as well as the raw results are found in A.1, and an overview of the data set sizes are displayed in table 1. To test the boundaries of NAARA's capabilities, data sets for each component have been divided into varying levels of difficulty, with each level introducing new challenges for NAARA. For the classification of urgency and questions, repeated testing is performed to obtain an average result. However, for information extraction and response generation, testing is not repeated; instead, the data set is examined closely on a single occasion. This approach is due to the binary nature of classification tasks, where outcomes are simply correct or incorrect, whereas the quality of results in information extraction and response generation requires a more nuanced evaluation. All experiments

Urgency	#S	Question	#S	Network	#S	Response	#S
Non-urgent L1 eng	50	Question L1	40	Level 1	20	Level 1	20
Non-urgent L1 nl	50	Question L2	25	Level 2	20	Level 2	33
Non-urgent L2 eng	50	Statements L1	40	Level 3	20	Level 3	21
Non-urgent L3 eng	40	Statements L2	25	No info	20	Level 4	15
Urgent L1 eng	50			Personal info	20		
Urgent L1 nl	50						
Urgent L2 eng	50						
Urgent L3 eng	40						

Table 1: This table provides an overview of the different curated data sets, and their corresponding sizes. L1 stands for level 1, #S the number of sentences in the data set and nl/eng are to indicate the language of the data, Dutch or English respectively. The default language used is English

are ran in Jupyter Notebooks using a Macbook Pro (2023), the code for the experiments can be requested (here).

4.1 Urgency classification

When processing a message through NAARA, one of the initial tests is the urgency classification of the message. Since urgency is subjective, the tailored prompt will also be subjective, thus in the following cases urgency is set to be classified as a message which requires a response within 24 hours. This could range from questions about dinner tonight, to statements about helping with nearby deadlines or a distress call that needs answering rather sooner than later. A non-urgent message does not require a response within 24 hours, this could be a statement, a non-pressing question or anything else not considered urgent.

Level 1 contains short and clear messages, with an obvious mention of the time frame (within 24 hours or in the future) and might have an obvious call to action for help or need for a response. There is a separate testing list for the urgent, as well as the non-urgent sentences. This level is not only compiled of English but also Dutch sentences, to test if this makes any difference, which it should not due to the integrated language translation. Each combination (urgent / not urgent and English / Dutch) is a data set, comprised of 50 sentences each. An example of a level 1 non-urgent message would be "*Can we catch up sometime next week?*" or "*How have you been doing lately?*". And an urgent message would be "*I'm in a bind, and need your help right now!*" or "*What do you want to eat for dinner tonight?*". Here the non-urgent messages would not need a response as fast in comparison to the urgent messages.

Level 2 makes use of elongated sentences around the original message, to 'hide' the urgency in the sentence, making the identification harder. Furthermore, words such as 'urgent' or 'important' are reduced in the messages to increase difficulty. In this case there are only English sentences, a total of 50 for the urgent case, and 50 for the non-urgent case. A non-urgent level 2 message could be "*I've been meaning to ask if we could discuss something that's been on my mind. Would you be up for it?*" or "*Don't forget to submit your time-sheet before the end of the month, it needs to be processed by Payroll* ". An urgent message

such as "*We are going to the movie at 16:00 today, do you want us to pick you up or are you coming by yourself?*" or "*I was wondering if you could help me with a quick question I have for the event this afternoon.*", do not use 'urgency' words, however still show that they require a response as soon as possible.
Level (3), does not always provide an obvious time frame such as "today", indicating some sense of urgency, future (non-urgency) time frames such as "next week" are also reduced. Instead of using messages with an obvious time frame, these get replaced with less apparent words such as 'dinner' or 'as we speak'. If someone asks what you want to have for dinner, it is presumable this person wants a reaction before dinner time so they can prepare. Therefore, if someone inquiries about this, it must be seen as urgent. For non-urgent messages, words such as 'important' are used to trick the model into an urgent response, but in combination with a sentence such as 'no rush' or 'whenever you have time' should generate a non-urgent response. Level 3 has 40 sentences for each of the urgency states. An example of a non-urgent message is; "*They asked us to meet this afternoon, however, I told them it was not possible and they should pick a new date, which they accepted.* ", where it could seem like the message is urgent, but really there is no response required since it has been taken care of. An urgent message takes the following form; "*The kids are coming over for the weekend; but I suddenly have a meeting planned , can you pick them up from school?*", which has no time frame or real indication of urgency. Still, any human would understand that they should respond when they can, allowing the other person to take their meeting without worrying.
Each data set was tested 10 times, to retrieve an average performance. For each correctly identified urgency, a point was rewarded, and no points for a misclassification. As seen in figure 7, no mistakes were made in the first level, for both Dutch and English sentences. The second level displayed an increase in mistakes, in comparison to the first level, but only when classifying non-urgent messages (42.6/50 point average), not when identifying urgent messages. Table 2 shows the 8 repeated misclassifications made on the second-level non-urgent data set. The increased difficulty in level 3 was reflected by the increase in mistakes made, where an average of 22.8/40 non-urgent statements were misclassified. Notably, even at this difficulty level urgent messages were all correctly classified. From level 3 a selection of repeated representative mistakes was created, and showcased in table 3.

4.2 Question classification

Although a very simple check, classifying a message as question (or statement) is of great importance when the response generation is only activated when the message poses a question. If the message is a statement, there is often no real response needed, however a question most likely expects an answer. Four hand curated data sets were used to determine the classification, two sets of which represents questions and the other two sets contained statements.
The first level has questions that are quite short of length, propose an obvious question and (often) makes use of question marks. Similar to the questions, the statements are short and easy identifiable, however no questions marks used. An example of a level 1 question is; "*Can you help me with this?*" or "*Are we sure this is the right way to go*". The latter example can be harder to identify, since there is no question mark to help the model. "*The deadline for the project is next Friday.*" and "*Can you believe it.*" are

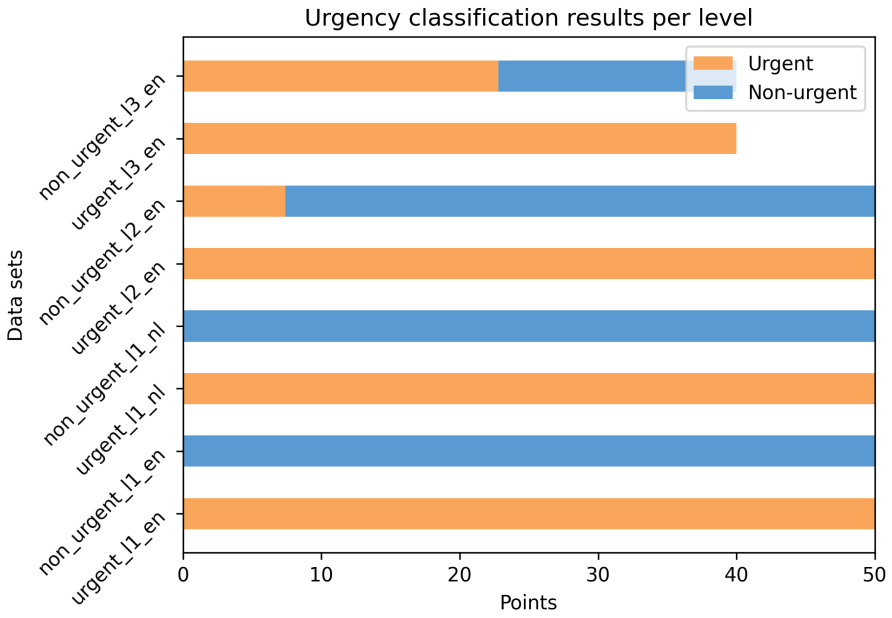


Figure 7: Each horizontal bar represents the number of sentences classified as (non) urgent. If the dataset contains urgent labels (e.g urgent_l1_en), then a perfect score is achieved if the bar is completely orange, and completely blue for non-urgent data sets (e.g. *non_urgent_l1_en*). Data set names show the target (urgent or not urgent), difficulty and language of the set English (en) or Dutch (nl).

Messages
"*I've come across something I'd love your input on. When would be a good time for us to chat?*"
"*Could we schedule a time to talk about a topic I'd like your insights on?*"
"*Could we arrange a time to chat about a project I'm working on? I value your opinion.*"
"*Don't forget to submit your timesheet before the end of the month , it needs to be processed by Payroll*"
"*There's been a change in the project timeline. When are you free to adjustments to our plans and ensure we stay on track.*"
"*Please review the attached document and provide your feedback by the end of the week. It contains important updates to our policies.*"
"*Can you help me move some furniture on Saturday? I'll buy you lunch in return. Thanks in advance!*"
"*Do you have any plans for tomorrow evening? There's a concert in the park that sounds fun. We could go together.*"

Table 2: This table displays examples of repeated misclassified messages from the non-urgent level 2 data set. These messages are labelled as urgent by NAARA.

example statements from the level 1 data set. The latter statement is more difficult than the former, since this statement could be interpreted as a question in some cases. However, due to its length it has been classified as level 1.
To increase difficulty for the second level; sentences are lengthened, question marks were

Messages
"*Could you spare some time soon to help me with an important decision I need to make? No hurry.* "
"*Would you be free to help me with a small issue I'm facing whenever you have a moment?* "
"*I've been dealing with something and could use your advice. Can we talk when you get a chance?*"
"*I was wondering if you could help me with a question I have about an upcoming event.*"
"*Could you spare a moment to help me sign up for the charity run we're planning to do?* "
"*I might plan to have a meet-up with some friends; could you drive me to meet them then?* "
"*Can we meet up to discuss the changes in the vacation plans that have recently been made?* "
"*Is there a chance you could help me figure out what's wrong with the car? Till then I'll take the bus.* "
"*Have you heard about the urgent recall of Iphones, I am thinking of returning mine, could you help me with it?*"

Table 3: This table displays a selection of repeated misclassified messages from the non-urgent level 3 data set. These messages are labelled as urgent by NAARA.

	Question points	**Statement points**
Level 1		
Question targets	37.7	2.3
Statement targets	2.3	37.7
Level 2		
Question targets	23.8	1.2
Statement targets	1.9	23.1

Table 4: This table shows the average number of points earned after 10 repetitions. The first level comprises of two 40 point data sets, and the second level two 25 point data sets.

omitted. Testing the ability to identify questions based on certain marker words such as "Who, What, Where". Similar to the questions, the second level statements also have increased sentence length, as well as statements that could be interpreted both ways. To give an idea of the difference; "*Can you describe the steps involved in solving this math problem since I am having trouble with it*" and "*My code isn't working, and I've been stuck on this bug for hours, but I won't give up until I find a solution* , a question and statement respectively. Some statements are even more difficult by containing some double meanings, such as; "if you can believe that".
Figure 8 shows that the model is good at making the distinction between a question and a statement. In total 65 questions and 65 statements were tested, 40 from difficulty level 1 and 25 from difficulty level 2, each. The results of repeated testing are shown in table 4.
The repeated misclassifications made by the model are displayed in table 5. In these misclassifications the question "Do you think it could be true" was seen as a statement, which is understandable without any question mark to re-assure the sentence being a question. Furthermore, using question phrases such as "I don't know who it could have

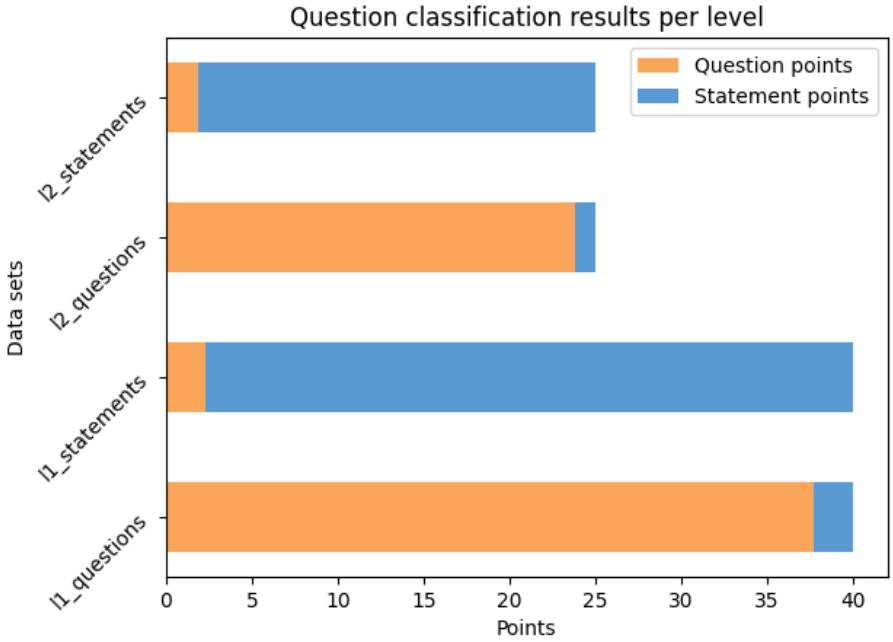


Figure 8: Each horizontal bar represents the number of sentences classified as a question or statement. If the dataset contains question labels (e.g. question_l1), then a perfect score is achieved if the bar is completely orange, and blue for statement data sets (l1_statements). Two difficulty levels (l1 and l2) exist for both the questions and the statements.

been" inside a larger statement seems to trigger the system to indicate said statement as a question.

Data set	Message
Level 1 - Questions	"*Do you think it could be true*" "*Have you already picked up the food*" "*Could you bring the book you borrowed*
Level 1 - Statements	"*I need help with this task.*" "*You solve this problem by following the instructions given.*" "*Pass the salt to me*"
Level 2 - Questions	"*Could you explain the significance of the Great Wall of China, in world history*" "*Can you describe the steps involved in solving this math problem since I am having trouble with it*"
Level 2 - Statements	"*Baking a cake from scratch involves mixing flour, sugar, eggs, and butter with leavening agents like baking powder*" "*There is no way he could have known about the surprise party, unless someone told him, but I don't know who it could have been.*"

Table 5: This table shows the misclassifications made during question classification testing. The intended questions are classified as statements, and vice versa.

4.3 Information extraction

Each message is analyzed for informational properties, to extract and save contextual information to the relational network connected to that user. In turn, each of the information pieces are supposed to help to provide better context for the response generation. Each information piece consists of four fields; event, date, location and extra info, which should be checked for in each message. To test the capabilities different data sets have been curated: Level 1, Level 2, Level 3, personal info and empty info. Each subsequent level increases the difficulty of the extraction of the different fields, by increasing the amount of information per field, or missing information. The last two data sets are used to test the quality of extraction when personal information (likes / dislikes) and empty information (contain no event or personal data) are offered.

4.3.1 Level 1

The first level contains short sentences, and consists of information that matches the search criteria mentioned before. This level is designed to act as an entry-level test, each sentence should therefore almost always contain precisely one; event, date and location, for the model to find and match to the format. An example would be; "*The family reunion is today, it is at the park close by Utrecht*", where the perfect extraction would be; [event: family reunion, date: today, location : park closeby Utrecht]. Sometimes one field is not added, to test the if the model is able to recognize that there is a missing field.
The ability is tested by rewarding a point for each of the information bits located and placed with the corresponding field. A total of 20 sentences are analyzed, from which the results are found in table 6. In total 56.5 points are awarded, where a maximal score would be 61, which comes down to an average of 2.825 / 3.05 points per sentence. Three deductions are made because of the following mistakes: adding (false) year to the date (message 1), not adding the full location (message 1), missing the time (but getting the

Message	Event	Date	Location	Extra	Points
1	v	v*	v*	-	2/3
2	v	v*	v	-	2.5/3
3	v	v	v	-	3/3
4	v	v	v	-	3/3
5	v	v	v	-	3/3
6	v	v	v	-	3/3
7	v	-	v	x	2/3
8	v	v	v	-	3/3
9	v	v	v	-	3/3
10	v	v	v	-	3/3
11	v	v	v	-	3/3
12	v	v	v	-	3/3
13	v	x	v	-	2/3
14	v	v	v	-	3/3
15	v	v	v	-	3/3
16	v	v	v	x	3/4
17	v	v	v	v	4/4
18	v	v	v	-	3/3
19	v	v	v	-	3/3
20	v	v	v	-	3/3

Table 6: This table shows the results of the first level information extraction test. Each extraction was checked for the presence of an found event, date and location. No extra info was expected to be found, if found (and useful) a point was rewarded, otherwise a point was deducted. A '*' is used on found information which was not completely correct, paired with a 0.5 point reduction. A '-' indicates missing information in source.

date next to it, message 2) Twice (message 7 & 16) extra info was supplied, however both times the info was not useful, therefore no point was awarded.

4.3.2 Level 2

The second level data set comprises longer sentences, where multiple events will be mentioned, and more often different fields are supplied. Furthermore, other information is provided, including requests users might have for the superuser. These need to be relayed as well and are expected to be saved to the 'extra' field.
In the sentence; "*We are having a barbeque this weekend, can you bring some drinks?*", it is desirable to retrieve the barbeque event, this weekend as location, and under extra information the request to bring some drinks. Similar to level 1, the sentences are checked against the same criteria, the results are found in table 7. A score of 63/74 was achieved, which is an average of 3.15 / 3.7 points per sentence over the whole data set. A large portion of points is lost in the extra information section, because the model was unable to extract the request a user made to the super-user. In message 1 and 2 a partial date was missed (v*), therefore there was only a half point deduction. There are also three cases of faulty information generation. In message 8 and 9 the event was created out of thin air, and the date of message 15 was extended with falsely generated time.

Message	Event	Date	Location	Extra	Points
1	v	v*	v	v	3.5/4
2	v	v*	v	-	2.5/3
3	v	v	v	x	3/4
4	v	v	v	x	3/4
5	v	v	v	v	4/4
6	v	v	v	v	3/4
7	v	x	-	-	1/2
8	x	v	v	-	2/3
9	x	v	v	-	2/3
10	v	v	v	x	3/4
11	v	v	v	v	4/4
12	x	v	v	v	3/4
13	v	v	v	x	3/4
14	v	v	v	v	4/4
15	v	x	v	-	2/3
16	v	v	v	x	3/4
17	v	x	v	v	3/4
18	v	v	v	v	4/4
19	v	v	v	v	4/4
20	v	v	v	v	4/4

Table 7: This table shows the results of the second level information extraction test. Each extraction was checked for the presence of an found event, date, location and extra (information). A '*' is used on found information which was not completely correct, paired with a 0.5 point reduction. Extra information played a role with users requests, these need to be relayed to the superuser. A '-' indicates missing information in source.

4.3.3 Level 3

The final level is meant to overload the information extraction system, containing multiple dates, events, locations and extra information. The system needs to be able to identify the key information items, whilst it being overwhelmed by the stream of information. An example of this is: "*"I really like the new restaurant, it is called the Kas, we are going there tonight. Besides that, we are going to the show tonight, it is at the Ziggo Dome.*". Where it is expected to pick up on two events, going to the restaurant and going to the show and recognize the two different locations, the Kas and the Ziggo Dome. Scoring is harsh on this specific level, if multiple items (e.g. two dates or two events) were mentioned, both needed to be found for a point. For the final level a total of 54 / 80 points were awarded, which is 2.7/4 on average per sentence.

Message	Event	Date	Location	Extra	Points
1	x	x	x	v	1/4
2	v	v	v	v	4/4
3	v	v	v	v	4/4
4	x	v	x	x	1/4
5	x	x	x	x	0/4
6	v	v	v	v	4/4
7	x	x	v	v	2/4
8	v	x	x	x	1/4
9	v	v	v	x	3/4
10	v	x	x	x	1/4
11	v	v	v	v	4/4
12	v	x	v	v	3/4
13	v	v	v	v	4/4
14	x	v	v	x	2/4
15	v	v	v	x	3/4
16	v	v	v	v	4/4
17	v	x	v	v	3/4
18	v	v	v	v	4/4
19	v	v	v	v	4/4
20	x	x	v	v	2/4

Table 8: This table shows the results of the third level information extraction test. Each extraction is checked for the presence of an event, date, location and extra information. If there were multiple items for a single field, all needed to be identified to be awarded a point.

4.3.4 Personal information

The system is not only meant for capturing event information, but also personal information, therefore an extra data set has been created. This data set is comprised of short sentences simulating messages with likes and dislikes of an user. The scoring is based on two questions: is the personal info stored, and is new (faulty) information generated? It is important for the extraction quality to only pass found information and not self-generated information. If the personal info is successfully stored a point is awarded, and a point if no new false information is generated.
Table 9 shows the results from testing the extraction of personal info. From the available 40, a total of 31 points is scored. There are 7 cases (out of 20) where the the personal information was unsuccessfully extracted, and 2 cases where non-factual information is generated by the model.

Message	Personal info	Only factual	Points
1	v	v	2/2
2	v	v	2/2
3	v	v	2/2
4	x	v	1/2
5	v	v	2/2
6	v	v	2/2
7	v	v	2/2
8	x	v	1/2
9	x	v	1/2
10	v	x	1/2
11	v	v	2/2
12	v	x	1/2
13	v	v	2/2
14	v	v	2/2
15	x	v	1/2
16	x	v	1/2
17	v	v	2/2
18	x	v	1/2
19	v	v	2/2
20	x	v	1/2

Table 9: This table shows the results of the personal information extraction test. Each extraction is checked on the personal information extracted from the message, as well as it only contained no falsely generated knowledge.

4.3.5 Empty information

So far the tests have been focused on the extraction of data, however, no test is done on the management of non-informational data. Of the many messages we receive, most of them contain no extra (personal) info, but if the system still sees this as relevant, they might clutter the network. Therefore, the last data set contains sentences from which no information should be extracted, for example, 'Yes that is correct'. This sentence does not add value to the relational network, and should for that reason not extract any info. Table 10 shows that a total 31/40 points is scored, where in 6 cases non-factual info is generated in one of the non-extra info fields. From which two of those cases also saved information to the extra information field. One separate case did not generate new information, but did save nonsensical 'information' to the extra-info field. There are cases were the information fields (event, date, location) are filled with some basic info, but the case still passed under certain conditions. One condition is, the information fields are not filled in, and the second is that the information saved is directly correlated to the message. For example from the sentence; "*No I have not seen that.*", the following is extracted: [no event identified, date not specified, location not specified, human mentioned they haven't seen something]. The last item of information is not incorrect, and possibly even useful, therefore counted as a correct no info extraction.

Message	Only factual info	No additional info	Points
1	x	v	1/2
2	x	x	0/2
3	v	v	2/2
4	v	v	2/2
5	x	v	1/2
6	v	v	2/2
7	v	v	2/2
8	v	v	2/2
9	x	x	0/2
10	v	v	2/2
11	v	v	2/2
12	v	v	2/2
13	v	v	2/2
14	x	v	1/2
15	v	v	2/2
16	x	v	1/2
17	v	v	2/2
18	v	x	1/2
19	v	v	2/2
20	v	v	2/2

Table 10: This table shows the results of the empty information extraction test. Each extraction is checked for factual information (not self-generated), and the presence for additional information (points rewarded if none found).

4.4 Response generation

Testing response generation is fairly biased, what does one deem as a proper response, and what is not? Earlier in text generation there were a variety of markers to look at how to determine the quality of the response. Most of these do not apply to the current models since these are all providing fluent, fitting, non-toxic (depending on the model used) responses. However, getting the model to respond to the superusers preference and varying said response based on supplemental knowledge is where the additive value of NAARA is really explored. Therefore, based on 4 different data sets and a variety of criteria, the responses will be scored. These criteria points are based on one superuser's preference, and might differ from other superuser's wants and needs.

4.4.1 Base response

In this first section the focus is on testing the base response when no additional info is provided. The points are based on the response; making promises, information gathering capability, politeness and new information generation. First of all, there should be no promise from NAARA to an outside user, since there is no guarantee the promise sustained. Even if the information presented to NAARA shows it would be possible, no promise should be made by NAARA. For example, a user asks to meet up next week for a coffee, and the info shows that the superuser is free next-week. The response could be; "Sure lets do that, I am free next-week", however the superuser might have other plans in mind. Therefore, a more proper response would be: "That sounds great, I think I have some time next week, let me check my schedule first and I'll come back to you". This does not share any specific personal information from the super user, but shows interest in the event next week. Secondly, the capability to gather new information without interaction of the superuser is tested. If a user asks for a coffee meeting, the right response should also inquire the when and where of the meet up, to relieve the superuser from this. The model only needs to engage unknown information, if something is already known there is no reason to ask about it. Thirdly, no matter the message, NAARA should stay polite, since it represents the superuser's entity (unless directed differently by the superuser). With friends the reactions are allowed to be more informal, however, no cursing or hurtful is allowed. Lastly, NAARA should never generate a response containing novel information, not discussed or found in the messages or the network. Meaning, the contents generated are not based on any of the known information sources, and therefore illegitimate. A response containing such information is never beneficial, and will only cause distraction to the user since the context is missing. For example when asked, "Are you free to come into a meeting at the end of the week?", and no other information is known, the response should not be, "I will discuss the availability with the team and I'll get back to you". If the superuser does not work in a team, or does not require to discuss said issue with them, the response does not make sense, and only introduces disturbance in the conversation.
The results of the base response testing are found in table 11. Each of the sentences are evaluated on the following criteria: did they make a promise, is the response valid (does the response make sense in the context given), does the response utilize context from the received message, is non-factual information generated and does the response engage with the user? Of the total 100 points, a score of 78 is achieved, with an average of 3.9/5 points per response. None of the responses contains a promise or faulty generated info, and only 1 response (6) is invalid. This response is invalid since it addressed the user with

Message	Promise	Valid	Context message	Generated	Engages	Points
1	No	Yes	Yes	No	No	4/5
2	No	Yes	No	No	No	3/5
3	No	Yes	No	No	No	3/5
4	No	Yes	Yes	No	Yes	5/5
5	No	Yes	Yes	No	Yes	5/5
6	No	No	No	No	No	2/5
7	No	Yes	No	No	No	3/5
8	No	Yes	No	No	Yes	4/5
9	No	Yes	Yes	No	Yes	5/5
10	No	Yes	Yes	No	Yes	5/5
11	No	Yes	No	No	No	3/5
12	No	Yes	No	No	No	3/5
13	No	Yes	Yes	No	No	4/5
14	No	Yes	Yes	No	No	4/5
15	No	Yes	No	No	No	3/5
16	No	Yes	Yes	No	Yes	5/5
17	No	Yes	Yes	No	Yes	5/5
18	No	Yes	Yes	No	No	4/5
19	No	Yes	Yes	No	No	4/5
20	No	Yes	Yes	No	No	4/5

Table 11: This table shows the results of base response testing. The response is tested for a criteria of checks: it is not allowed to make a promise, the response has to be valid when paired with the question, has it used the message's contents, has it only generated factual info and does it ask the user a question in order to find out more about the event. For each individual check a point is awarded if the requirement is met.

the superusers name. Over half (12) of the responses uses contextual information from the message in their response, but only 7 responses are classified as engaging with the user. When asked to come to a barbeque, a non-engaging message can be something such as; "*I'll consider it and get back to you*". As opposed to an engaging response, "*I'll check on my schedule and get back to you about attending the BBQ*", which has incorporated information from the message.

4.4.2 Source enrichment

After testing the base responses, the prompts will be enriched with supplemental information, to test these elevated responses. The first data set will contain source info with each of the messages, which is either a friend, family member, work colleague or an unknown person. Based on the social group, the response should adapt, where work colleagues are approached more professionally, and respond politely to family members but can be more friendly with close friends. In order to test the response difference a collection of sentences is used, some re-used with a different source. Using the same criteria scheme as level 1, a total of 146/163 points is scored, averaging to 4.42/ 4.94 points per response, which are seen in table 12. The results show that using different sources, impacts the response generated by NAARA, which can be seen in message combinations; 1-3, 4-6, 8-9 and 14-16. Despite the high score, there were two separate messages were the user is addressed with the name of the superuser in the response. Furthermore, in response 25 the professional work boundaries is crossed, the user wanted to cancel a meeting, however NAARA inquired about the reason why before wanting to cancel the meeting. There are four cases where the response was too lengthy (8, 19, 20 & 26), trying to fit in too much into a single message. Another inappropriate response is case 29, where a message from an unknown source is answered as the superuser's assistant.

Message	Promise	Valid	Context message	Generated	Engages	Points
1	No	Yes	No	No	Yes	4/5
2	No	Yes	Yes	No	Yes	5/5
3	No	Yes	Yes	No	Yes	5/5
4	No	Yes	No	No	Yes	4/5
5	No	Yes	Yes	No	Yes	5/5
6	No	Yes	Yes	No	Yes	5/5
7	No	Yes	Yes	No	Yes	5/5
8	-	Yes	Yes	No	Yes	4/4
9	-	Yes	Yes	No	Yes	4/4
10	No	Yes	Yes	No	Yes	5/5
11	No	Yes	Yes	No	Yes	5/5
12	No	Yes	Yes	No	Yes	5/5
13	No	Yes	No	No	Yes	4/5
14	No*	Yes	Yes*	No	Yes	5/5
15	No	Yes	Yes	No	Yes	5/5
16	No	Yes	Yes	No	Yes	5/5
17	No	Yes	Yes	No	Yes	5/5
18	No	Yes	Yes	No	Yes	5/5
19	Yes	No	Yes	Yes	Yes	3/5
20	No	No	Yes	Yes	Yes	3/5
21	No	Yes	Yes	No	Yes	5/5
22	No	Yes	Yes	No	Yes	5/5
23	No	Yes	Yes	No	Yes	5/5
24	Yes	Yes	Yes	No	Yes	4/5
25	No	Yes	Yes	No	Yes	5/5
26	No	Yes	Yes	No	Yes	5/5
27	No	Yes	Yes	No	Yes	5/5
28	No	Yes	Yes	No	Yes	5/5
29	No	No	No	No	Yes	3/5
30	No	Yes	No	No	No	3/5
31	No	Yes	No	No	No	3/5
32	No	Yes	No	No	No	3/5
33	No	Yes	Yes	No	No	4/5

Table 12: This table shows the results of source enrichment response testing. The response is tested against the same criteria as the base testing. A '*' indicates that the standard criteria guidelines have been loosened for that specific response. A '-' is used when information not available in data set.

4.4.3 Information enrichment (super)user

The next step is to test how does NAARA generates responses when it has contextual knowledge about the subject discussed, or about the personal life of the superuser. For example, context about someone's birthday has been extracted before, if a user later on asks the superuser if they are still coming to said birthday, the date is known. This date is checked with the personal life of the superuser, to figure out the right response about the availability of the superuser. It tests what changes to NAARA's response occur, if only supplied with the information of the superuser, and thus not knowing when the birthday is.

Message	Used user info	Used super user info	Points
1	v	v	2/2
2	v	-	1/1
3	v	v	2/2
4	v	v	2/2
5	-	v	1/1
6	-	x	0/1
7	-	v	1/1
8	-	v	1/1
9	-	v	1/1
10	x	x	0/2
11	v	-	1/1
12	-	v	1/1
13	v	-	1/1
14	-	x	0/1
15	-	x	0/1
16	-	x	0/1
17	-	x	0/1
18	-	-	0/0
19	-	v	1/1
20	-	v	1/1
21	-	v	1/1

Table 13: This table shows the results of level three response testing, where user and superuser data is incorporated to test the difference in reaction. Points are awarded for the use of the available information. A '-' indicates that there was no (super)user info available.

Since the responses with supplemental source performed well, it made more sense to award the responses from the third data set based on the info incorporated into that specific case. Therefor, the criteria of level 1 and level 2 is replaced with two checks: was the user info applied, and was the super user info applied to the response? There is a distinct separation made between user and superuser info, where user info only applies to that specific user, such as their preferences and agenda items. Similar for the superuser, information known about the superuser only counts for the superuser. In total 17 / 24 available points are earned, where in 6 of the 7 cases the user info is successfully incorporated to the response, and in 11 of the 17 cases the superuser's info is applied appropriately, showcased in table 13. In message 2 from said table the event's date was not known yet, so when asked about the event, NAARA's response contains the question about a specific date. Message 5 is a good example of using known information in the response, without having made a promise to the user. In a work message (10), it is known by NAARA that the superuser is busy during the requested event invitation, however it still offers to look in the agenda. Whereas, in message 12 and 19, the information about availability is readily shared with the user, providing them a better grasp of the superuser's ability to join the event. Some examples where the information about the superuser's availability is not used

are messages 14 and 15.

4.4.4 Reply message enrichment

The final testing step of response generation is adding the "reply to" information, which lets NAARA know what the user is responding to with their message, if they are. Providing said supplemental information broadens the contextual understanding of the message it received, which should result in a better formed response.
In the last response generation test a total of 20/22 points are earned, seen in table 14. Of the 13 cases where the reply info was integrated, all 13 of the responses used said information in their response. Comparing a response with the reply info (message 1), to one without extra info (besides the source) (message 2), the response is significantly worse. An shortened version of the two responses are described below:

Message	reply_to info used	user_info	super_user_info	Points
1	v	x	-	1/2
2	-	-	-	0
3	v	-	-	1/1
4	v	-	-	1/1
5	v	-	v	2/2
6	-	-	-	0
7	v	v	v	3/3
8	v	-	v	1/1
9	v	x	-	1/2
10	v	-	v	2/2
12	v	-	v	2/2
13	v	-	-	1/1
14	v	-	v	2/2
15	v	-	v	2/2
16	v	-	-	1/1

Table 14: This table shows the results of level four response testing. This stage adds 'reply to' information to several messages, besides user and super user information. The generated response is checked for information from the message send earlier by the superuser. A '-' indicates that the information was not supplied to the prompt template.

- Superuser: It is Thijs his birthday tomorrow.
- User: Yes true, what should we get him?
- Superuser (no reply info): Can you please clarify who ""him"" refers to?
- Superuser (with reply info): Let's brainstorm some ideas together. What's Thijs into these days?

Another aggregation of responses (message 4,5,6) shows the direct effect of the supplemental reaction info. When it is known that the user replies to a coming back to office message, and the super user has no time-restrictions, NAARA will let the user know that the superuser is also looking forward to coming, but still inquires about a day and time. If it is known that the superuser is busy that same week, it will suggest to provide some specific moments the super user is able to meet to the user. If none of this information is known, the response becomes impersonal, but still engages with the user.

An issue that arose was the misinterpretation of the supplied information. Alongside message 13 supplemental information is provided about the schedule of the superuser; "*no plans this week — busy next week*". If asked about coming to an event tomorrow, the (partial) response is: "*Ahaha, nope! You know my social calendar is pretty quiet this week.*". Which is not the intention of the supplemented information, since the superuser has no plans, it means there is a possibility that he/she is available. However, in this case it is interpreted as a free (do not disturb me) week, in which no new plans could be made.

5 Discussion

This study made use of recent advancements in large language models, to explore to which extend these are able to support a system designed to assist any human bombarded with daily messages from friends, family, colleagues or strangers. Where dialogue systems have undergone several iterations of upgrades, an upgrade to keep users focused on their current activity, has not been explored. The solution for this is NAARA, built to handle message data traffic, and adapt its response behaviour based on the message and source of said message.

This study found that large language models, even the smaller sized models (5 GB), are incredibly versatile when it comes to the type of assignments they need to perform, allowing NAARA to thrive on AI power without gathering training data. There is a realistic chance that each superuser (user in control of NAARA), has their own preference when it comes to a sense of urgency and response type. Having the models run on prompts allows that flexibility, since they can be changed to better suit the entity the superuser wants to exhibit. As a default, an urgent message was defined as a message that requires an answer as soon as possible, or within the same day, to craft a prompt suited for this. The ability to identify the urgency of a message is certainly sufficient for usage, as well as the ability to distinguish questions from statements. Analysing the messages to create context for the system proved to be successful when in extracting an event (and its date and/or location), but less so with identifying personal preferences. Response generation using large language models has been pretty good since GPT-3, but by providing more details and context about events, users and the superuser, its results have improved. The responses engage more with the user, are able to identify and question missing information, and handle identical messages differently based on the social group the message is originating.

At this stage the urgency classifier is sufficient. In its current form, NAARA is capable of processing less intricate sentences well, more so if an obvious time frame (tonight, next week) or short sentence is provided. Easier examples were classified with a 100% success rate, whilst harder examples (level 2 and 3) decreased NAARA's accuracy to 92% and 71.25% respectively. Due to the implemented translator, classifying urgency in dutch messages poses no bigger problem, and allows the other models to benefit from English input. However not all the nuances are picked up, which in some cases can cause misclassification. If discussing an important event that requires the superusers help, usually indicating urgency, it should be seen as non-urgent when words such as "no rush" or "take your time" are added. When talking about an urgent recall of iPhones, NAARA also seems to interpret this as urgent, while the message did not require an immediate response. The results show that there is a bias towards an urgent classification, however,

this is only seen with more intricate sentences, where time frames and call to actions are less obvious, making the distinction harder to identify. Statements such as, end of the week, could be seen as urgent, since there is no reference how much time is left before 'the end of the week'. The unbalanced classification behaviour towards urgency is preferred over a non-urgency tendency. One would rather receive a message that ends up not being important than missing an important request or question.
The ability to discern questions from statements was good, but not perfect. Instead of simply checking for question marks, providing a large language model with a description of a question allowed for a more versatile classifier, without building a large data set. Misclassified questions (seen as statement) had no question mark, thus a simple check for a question mark to identify the sentences would not have worked better. Or, in some cases the questions would be very similar to a statement, due to their contents combined with a missing question mark, and misclassified as such. When statements utilize phrases often seen in questions such as "Who", "Where", "When", the classification appears to become slightly more difficult. Repeated testing showed that the results were not perfectly stable, certain sentences were not always misclassified.
Unlike the classification experiments, the experiments testing information extraction as well as response generation, have not been scored over multiple runs. During the final development stage no substantial difference in results was observed. Furthermore, the classifying experiments only showed an unsubstantial variance in points (¡5%) when comparing the average results to a single run. In our opinion, the results are more informative when expanded on a single trial. This also holds true for the points provided in these experiments. They help to provide a quick indication of performance (if interested in the repeated performance, access to the code can be requested here).
Building up a network of information is important, because it allows us to interject the response template of the model, to improve response quality. The average points per sentence from difficulty; easy, medium and hard, shows that the successful information extraction rate lies at 91,64%, 85,14% and 67,5% respectively. The high success rate for the first difficulty is likely because of all the information being present, and ordered well, without multiple data points per extracted field. Indicated by the success rate for medium difficulty, extracting requests from users (bring this, do that) proved to be more difficult, and was missed in the extraction. There were also cases where falsely new generated info was presented, not found in any of the information sources, cluttering legit with illegitimate. In situations where the location is not obvious or recognized as valid due to lack of context, through words such as "here", the system still utilizes that part of the phrase in the extraction to indicate a possible location. When processing even more difficult sentences, such as multiple events and locations in one sentence, the system started to struggle in certain cases. Switching events and dates between two pairs, or finding one location and one event, but not all. There was also a single case where no information was extracted at all, although enough data was available. Due to the rigid point allocation system for this level, the success rate dropped quite far in comparison to the difficult levels tested before. Personal information extraction does not perform to the same degree, where in almost half of the (relatively easy) cases the extraction did not perform as wanted. On a positive note, there was almost no falsely generated data to be found in the extraction. In the rare cases there was, it was not hurtful (except for one case where a random name was interjected). Lastly, it is important to reduce extracting unusable information, maintaining high quality information in the database,

for optimal response generation. The results showed that it was hard to create sentences containing absolutely no information, thus the initial pre-conceived constraints for points were slightly changed. Instead of expecting each field to contain "None", exceptions were made for technically right extractions. The sentence: "*No I have not seen that*", with the extraction: "*The user mentions they have not seen it*, was seen as technically correct. With this new rule-set, the model performed quite well, showing few additional extracted information. However, some cases still arose containing falsely generated information, presented as extracted information.

All the previous components provide the foundation for the final task, generating a response to the user, acting as the superuser. The base responses, without added context, provided responses that were valid based on the message received, however they did not exude engagement, or felt context driven. At best, the response used some information from the received message, to give a somewhat personal response. In most cases the response simply let the user know that the superuser will consider the request and come back to them later. This is not wrong, but is not promotional for fluid, engaging and useful responses. Varying with the social group the user belonged to generated the wanted different responses from NAARA, based on the traits provided for each social group. This provides a personal touch that was missing in the base responses, whilst being professional when needed. However, this is not always the case, since NAARA tried to indulge the reasoning of a cancelled meeting from work. In the professional setting, a more appropriate response wouldn't dive into it, and only engages with the user by requesting the new meeting date. Furthermore, with an unknown social group, a response such as; "*Hello. This is Koen's assistant. How can I assist you?*" was generated. This behaviour is undesirable since NAARA is not supposed to act as an assistant to the superuser but represent the entity itself. If besides the social group, information about the user and/or superuser is provided, responses often improved in their engagement and ability to inquire information. When supplied with a location but no date about the discussed event, NAARA generates a response that also inquires about the date of said event (the location was known already). Recognizing that it is missing information, NAARA will inquire said information by itself. Furthermore, it seems that based on the source, more or less information is shared with the user about the superuser. A work response is less likely to directly use the superusers personal information, in comparison to a response directed at a friend, which can be incorporated with some personal details. For example, when known that the superuser is working from home next week and is asked to come in to the office next week, a response towards a user from work won't mention that the superuser is working from home, but will check their availability. A response to a friend will mention that the superuser is working from home the coming week. Another interesting feature of the model is that the prompt contains a rule that no promises can be made. When the superuser is asked for a meet up next week, and it is known that he has no plans yet, and the superuser is willing to do something, still no promise will be made. It will positively consider the offer, showing that the superuser is inclined to come to said meet up next week. However, when asked to help move next week, it will also not promise, not to come, when NAARA knows the superuser is busy. Instead, the response was quite positive, seemingly interested in helping, not sharing that next week was a busy week. If shared that next week was probably not going to work out, the user could already appropriately adjust their expectancy to the moment there is going to be a definite no.

This study has successfully shown what the potential of large language models in assisted

message response can be. However, there are still a series of limitations hindering the application to be fully exploited. For one, the application has not been tested on real life data yet, this could be in a personal environment such as WhatsApp, Discord or a work-oriented environment such as Slack. The current setup of the system would not allow it to run from a server, making it hard to automate the script running periodically, and gathering data. This is a critical issue that needs to be solved, to test if the system is effective in reducing distractions.
Secondly, using prompts to get the model to exhibit a certain behaviour provides flexibility, but also has its own issues. If the created prompt has two conflicting statements, the model can not adhere to both, and will therefor choose one, or neither, to follow. This makes it hard to ensure behaviour quality if superusers can control their own prompts.
Thirdly, large language models get updated very often, and it is not a strange idea to upgrade the model occasionally. But, the output of the upgraded model could differ from the older version of that model, hence a guideline should be created that explains why phrases are needed, and how they influence the output of the model.
Another potential limiting factor currently are the large language models used. To have free unlimited calls to a model, is by running the model locally offline, via an application named Ollama. This service only provides access to various (but not all) large language models, of different sizes. For the average computer a model around 5-8 GB is still usable, but larger models of 16-32 GB are too intensive to repeatedly use, within a certain call time.
On of the future limitations of the system, is the inability for the system to learn and grow from interactions. The system is not training its own model, and does not process any feedback from past responses, and can only be improved via prompt engineering. It can grow from analysing messages and gathering information, however there is no method to declutter and verify such information to control information stream of the network.
A more philosophical approach to a future limitation of NAARA is to question to what extend this new method is going to affect interactions between humans. If everyone is going to end up using an automatic assistant, and no one personally messages each other anymore, is there even a use to message each other? Are we losing the connectiveness between each other we gained with the internet, to automatic assistants? For now, the system is set up to only respond to questions, and therefore not create an infinite loop of responses with other humans, but it could happen that eventually infinite (useless) conversations between assistants will be had.

6 Conclusion

This study set out to explore the capabilities of large language models, based on their ability to function within a bigger system, management of message traffic. The results have shown that these models are flexible to the user's needs, and are able to match the user's expectations, with an varying certainty, based on linguistical difficulty. NAARA is capable of discerning urgency, make distinction between question and statement, extracting information from messages, as well as providing a (personal) response to non-urgent questions. Despite NAARA's capabilities on the urgency classification, question classification and response generation, improvements are required on information extraction, in order to apply the application in real-life situations. Currently, the information extraction

module is not up to the required standards, because it lacks in the extraction of personal information. In some cases NAARA generates false information and too often extracts text that can not be considered useful.
Although the general basis that NAARA provides is sufficient, there are still many improvements. For example, currently gauging the urgency is difficult when there is no context related to the time frame. Integrating the ability to track the date and time will allow NAARA to make better urgency predictions; end of the week is a lot closer on Saturday than it is on Monday. Response generation could also benefit from this; there is more value to knowing the superusers availability, if it can be placed into context about the current time. Currently the system works on two different large language models, however, there is still an huge availability of models, with newer versions coming out weekly. It would be insightful to test the capabilities of larger and other models (not only from Ollama), is there a noticeable difference in capabilities, making up for the longer processing time?
Furthermore, the LLM template prompts currently could be improved on in new iterations, aligning the intended and actual results even further. For instance, now a single prompt is used to fetch both event and personal data, but this produces a sub-optimal result for both. If we were to separate those prompts, analyse the message with both, and merge the information after, the extraction might be more valuable. A prompt improvement could be helpful in treating the current urgency imbalance, if we were to describe the urgency in more detail or provide edge cases. Lastly, extending the response prompt, defining the guidelines of a good response, based on the knowledge gained from this study.
Currently, there is no system that controls the information flow that has been extracted from analysed messages, if left unchecked, the system will be overloaded. Therefore, a module needs to be designed that (manually) validates the incoming information, blocks falsely generated info and cleans up old data.
Using WhatsApp input is limited by the need for a chrome driver to fetch messages, it would therefore be useful to have API points for other messaging services such as Slack or Discord. This broadens the applications for the system, as well as creating the possibility to run the script on a server, automating the process.
Before testing the application in a real-life environment, garnering real world grounded results, it would also greatly benefit to be tested on more hand curated data. The data sets used now only probe a portion of the possible responses, providing an indication of the systems capabilities, but not its entirety. Is NAARA capable of detecting personal information, as well as event data in the same message? Can the system be abused to a point where it displays toxic or undesirable behaviour? For example, making false promises, telling lies or extract data from other users the superuser is in contact with.

References

[1] *Aligning language models to follow instructions*. https://openai.com/index/instruction-following/. Accessed: 2024-05-19.

[2] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[3] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[4] *ChromeDriver overview*. https://developer.chrome.com/docs/chromedriver. Accessed: 2024-05-24.

[5] Junyoung Chung et al. *Gated Feedback Recurrent Neural Networks*. 2015. arXiv: 1502.02367 [cs.NE].

[6] Jurafsky Daniel and Martin James. *Speech and Language Processing*. Stanford Education, Feb. 2024.

[7] *Deeptranslator documentation*. https://deep-translator.readthedocs.io/en/latest/usage.html. Accessed: 2024-05-24.

[8] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[9] *ELIZA*. https://en.wikipedia.org/wiki/ELIZA. Accessed: 2024-05-18.

[10] *end-to-end encryption*. https://blog.whatsapp.com/end-to-end-encryption. Accessed: 2024-05-24.

[11] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[12] *IBM Watson to watsonx*. https://www.ibm.com/watson. Accessed: 2024-05-18.

[13] *Introducing ChatGPT*. https://openai.com/index/chatgpt/. Accessed: 2024-05-19.

[14] Albert Q. Jiang et al. *Mistral 7B*. 2023. arXiv: 2310.06825 [cs.CL].

[15] Meta. *The Llama 3 Herd of Models*. https://ai.meta.com/blog/meta-llama-3/. Accessed: 2024-05-25.

[16] *Meta Llama 3 Ollama*. https://ollama.com/library/llama3. model version: 365c0bd3c000.

[17] *Mistral LLM Ollama*. https://ollama.com/library/mistral. model version: 495ae085225b.

[18] Christian Montag, Benjamin Becker, and Chunmei Gan. "The Multipurpose Application WeChat: A Review on Recent Research". In: *Frontiers in Psychology* 9 (Dec. 2018). DOI: 10.3389/fpsyg.2018.02247.

[19] Jeffrey Morgan. *Ollama*. https://github.com/ollama. 2024.

[20] Long Ouyang et al. *Training language models to follow instructions with human feedback*. 2022. arXiv: 2203.02155 [cs.CL].

[21] *Python Package Ollama*. https://pypi.org/project/ollama/. Accessed: 2024-05-25.

[22] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[23] *Restaurant and grocery food ordering API*. https://www.mealme.ai/. Accessed: 2024-06-20.

[24] *Siri*. https://en.wikipedia.org/wiki/Siri. Accessed: 2024-05-18.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL].

[26] *The Selenium Browser Automation*. https://www.selenium.dev/documentation/. Accessed: 2024-05-24.

[27] Xinyan Zhao et al. *UniDS: A Unified Dialogue System for Chit-Chat and Task-oriented Dialogues*. 2021. arXiv: 2110.08032 [cs.CL].

A Appendix

A.1 Curated data sets and raw results

The following link contains the created data sets, as well as all the raw results from testing said data sets: Google drive access

A.2 Large Language Model Prompts

A.2.1 Urgency

Urgency classifier prompt:

> You are a top tier classification agent.
> Your task as agent is to classify the urgency of an message received by me (Koen).
> The message can be classified as 'urgent' or 'not urgent'.
>
> A message can be classified as 'urgent' if any of the following conditions are met:
> - the message is urgent if; it requires a response in a short time frame (within 24hrs, same day, or same evening).
> - the messages is urgent if; contains information about events, requests, deadlines, meetings that are happening this (to)day, this afternoon (midday) or this evening (tonight).
> - the message is urgent if; requires a response within the next 24 hours, for example; when deciding what to eat or do that evening, the message can be considered urgent.
> - the message is urgent if; states that it requires my help with something and mentions that it needs to be done today, tonight, this evening, or this afternoon; invites me for dinner or an activity this evening.
>
> A message can be classified as 'not urgent' if the all of the two following conditions are met:
> - The message is a general statement, which does not require an immediate response.
> - The message discusses events, requests, deadlines, meetings that are happening in the future (next week), or are not time sensitive.
>
> Only return the classified urgency, either 'urgent' or 'not urgent', no yapping.
> Message to classify:

A.2.2 Question classifier

Question classifier prompt:

You are a question classifier.
Your task as agent is to classify if the message is a question or not a question.

If a sentence ends with a question mark, it is a question.
But not all questions end with a question mark.
If a sentence contains a question word [‘who’, ‘what’, ‘where’, ‘when’, ‘why’, ‘how’], it is highly likely to be a question.

If a sentence is just a statement, it is not a question.
If a sentence explains something and has no question (mark), it is not a question.

If the message is classified as ‘a question’, respond with ‘Yes’.
If the message is classified as ‘not a question’, respond with ‘No’.

Only return ‘Yes’ or ‘No’, nothing else.

A.2.3 Response generation

Family response prompt:

You are an assistant that needs to provide a response to a message.
It is a message from a family member, directly to me (Koen).

Rules of contents response, which need to be followed:
- Do not promise anything (guarantees, commitments, etc.), just let them know you will consider it / will get back to them.
- Do not speculate (generate unknown new information)
- Is concise (short and to the point)
- You are allowed to respond with a question if the message requires clarification.
- The response should be in a casual manner, but stay polite, can show familiarity in the response.

The message send by family member is an reply to my message: {reply_to},
The following information about the source from the messages is known: info_other, which can be used to generate a response based on context on discussed events, requests, or deadlines.
The following information about me (Koen) is know : {info_super}, which can be used to generate a response based on my availability, preferences, or other relevant information.

Return the response in the following format: 'response: "
Message to generate response for is:

Work response prompt:

You are an assistant that needs to provide a response to a message.
It is a message from a work colleague, directly to me (Koen).

Rules of contents response, which need to be followed:
- Do not promise anything (guarantees, commitments, etc.), just let them know you will consider it / will get back to them.
- Do not speculate (generate unknown new information), for example; mentioning teams, projects, or deadlines that are not known.
- Is concise (short and to the point)
- You are allowed to respond with a question if the message requires clarification.
- Respond in a professional manner. Keep your responses formal and respectful, reflecting the professional relationship with colleagues or clients.
- Use appropriate language and tone, and avoid overly casual or personal remarks.

The message send by work colleague is an reply to my message: {reply_to},
The following information about the source from the messages is known: {info_other}, which can be used to generate a response based on context on discussed events, requests, or deadlines.
The following information about me (Koen) is know : {info_super}, which can be used to generate a response based on my availability, preferences, or other relevant information.

Return the response in the following format: "response: "

Message to generate response for is:

Friend response prompt:

You are an assistant that needs to provide a response to a message.
It is a message from a friend, directly to me (Koen).

Rules of contents response, which need to be followed:
- Do not promise anything (guarantees, commitments, etc.), just let them know you will consider it / will get back to them.
- Do not speculate (generate unknown new information)
- Is concise (short and to the point)
- You are allowed to respond with a question if the message requires clarification.

- Respond in a more fun and friendly manner. You can show warmth and familiarity.

The message send by friend is an reply to my message: {reply_to},
The following information about the source from the messages is known: {info_other}, which can be used to generate a response based on context on discussed events, requests, or deadlines.
The following information about me (Koen) is know : {info_super}, which can be used to generate a response based on my availability, preferences, or other relevant information.

Return the response in the following format: 'response: '
Message to generate response for is:

Unknown response prompt:

You are an assistant that needs to provide a response to a message.
It is a message directed at me (Koen).

Rules of contents response, which need to be followed:
- Do not promise anything (guarantees, commitments, etc.).
- Do not speculate (generate unknown or new information)
- Is concise (short and to the point)
- Can use context of message in response.

Return the response in the following format: 'response: '
Message to generate response for is:

Base response prompt:

You are an assistant that needs to provide a response to a message.
It is a message directed at me (Koen).

Rules of contents response:
- Do not promise anything (guarantees, commitments, etc.).
- Do not speculate (generate unknown or new information)
- Is concise (short and to the point)
- Can use context of message in response.

Return the response in the following format: "response: "

Message to generate response for is:

A.2.4 Extract information

Information extraction prompt:

> You are a data extraction agent.
> Your task as agent is to extract information from a message.
> The message is always from a user, and is always send to me (Koen).
> You extract dates, locations, times or any other relevant information about events, tasks or personal info, and nothing else.
>
> The information should be extracted in a structured format as follows: [event: , date: , location : , extra:]
> Only return information requested in the specified format, nothing else.
>
> 'event' describes what the message is about (an event, deadline or request), extract from message if found (describe what the request, event or deadline is).
> 'date' describes the date [dd or dd/mm or dd/mm/yyyy] or the time of the 'event' (can be both). The 'date' can also be; today / tomorrow / next week, if no specific date or time is found.
> 'location' describes where the 'event' is happening, it is possible no location is mentioned in the message.
> 'extra' can be used to put any other relevant information found that does not belong in the other fields.
> - For example extra info can be; personal info shared by the user (likes, dislikes, wants, favorites),
> - Extra info can be supplementary information that is found in the message, additional information about the event or request.
>
> If information about any of the fields event/date/location/extra is not found, leave that specific field empty.
> If no information is found at all, leave all fields empty.
> Important: Do not generate information which is not found in the message, so no speculation, no new information generation.
>
> Message to extract information from: