# A hybrid chatbot that uses contextual sensors to influence responses.

Kevin Jetten

02-02-2021

Thesis manuscript for obtaining the degree of Master of Science in Artificial Intelligence, Utrecht University
In collaboration with Eindhoven University of Technology
Student number: 6496881
Supervised by: Baptist Liefooghe (UU) and Mathias Funk (TU/e)
Second examiner: Martijn Mulder

# 1 Abstract

AI chatbots are currently booming, and they have made great strides in the last couple of years in overtaking more traditional chatbot techniques. Nonetheless, AI chatbots are still lacking some functionalities that are uncomplicated for rule-based chatbots. This paper proposes a hybrid chatbot architecture that combines an AI and a rule-based chatbot to form a single chatbot that unites their respective strengths to cancel out some of their weaknesses. An AI chatbot is generally not able to permanently update its believes unless the model is retrained. On the contrary, rule-based chatbots can do this. Meanwhile, rule-based chatbots cannot appropriately respond to messages for which no appropriate rule is present, in contrast to AI chatbots. The hybrid chatbot can update its beliefs during conversations and can respond to messages for which no rules are known. Additionally, this ability to change beliefs during and between conversations allows for contextual sensors to influence the hybrid chatbot and provide accurate and current information to the user.

The evaluation of the hybrid chatbot is done using two methods. Firstly, an experiment in which twelve participants talked to the hybrid chatbot. Secondly, an automated experiment to evaluate the effect of the contextual sensor data. The results show that the hybrid chatbot performs better on the metrics of responsiveness, repetitiveness, and conversational depth when compared to the default version of the AI chatbot. Additionally, it is shown that the information provided by the contextual sensors influence the responses generated by the chatbot.

# Contents

# 2 Introduction

## 2.1 Background

The development of chatbots started in the mid-1960s, and they have been evolving ever since. One of the first natural language processing (NLP) bots was ELIZA [1], which used pattern matching and gave the illusion of understanding what the user was typing. It was one of the first chatbots capable of attempting the Turing test [2].

While writing this paper, both AI chatbots and Rule-Based chatbots are used most often. Many articles are written on which one you should use [3] [4] [5], as they both have their strength and weaknesses. Rule-based chatbots have, as the name suggests, a series of rules which decide which message is sent to the user. It still relies on pattern matching to determine what the user is saying or asking. One of the advantages of rule-based chatbots is that they can start off simple with just a small number of rules, and by expanding the set of rules, it can gradually keep growing to meet new needs. Additionally, it is simple to add, remove or change responses to rules when you already have a large rule structure in place.

AI chatbots work differently and do not rely on pattern matching or rules to generate responses. It instead uses machine learning to train a model that generates responses based on an input message. One of these models is DialoGPT [6]. DialoGPT is a dialogue response generation model trained on 147 million conversation-like dialogue Reddit comment chains. It generates close to human responses in terms of automatic and human evaluation in single turn dialogue settings, and the model is publicly available to use [6]. The DialoGPT model can be used to query responses to messages, which allows it to be a chatbot. The unique advantage of AI chatbots is that they can generate a coherent reply to messages they have never seen before. However, the quality and coherence of the responses are heavily dependent on the model's training data.

Regardless of which technique is used to build the chatbot, it is important that the chatbot is aware of the context of the conversation. Being contextually aware is challenging for chatbots as the current context is often implicitly defined and can change in the middle of the dialogue. Without knowing the current context, the chatbot will often be unable to generate appropriate responses.

## 2.2 Problem

AI chatbots have seen enormous improvements over the last couple of years, yet still lack some of the simple abilities that a rule-based bot can provide very easily. For instance, an AI chatbot does not have a knowledge base outside of its pre-trained model to retrieve up-to-date information. If the AI chatbot is trained on conversations with an interlocutor named Peter, it will not permanently change its belief when told that the interlocutor's name is now Randy. While the AI bot will presumably remember the new name for the remainder of that one conversation, as it can use the history of that conversation as context, it will not

remember it for a new and fresh dialogue. Rule-based chatbots, in comparison, are very good at this. They have the ability to update their knowledge base when they encounter contradicting information. However, they are unable to reply in an organic and human-like way to any messages that are not present in their knowledge base, like an AI chatbot is.

Quality control and reliability are also an issue for AI chatbots. They require training on a gigantic set of data, which makes them unpredictable. For example, if the training data is unfiltered and originates from social media, it can create offensive and inappropriate chatbots such as Microsoft's Tray [7]. Even if the training data is verified, there is no guarantee that the generated responses are appropriate or relevant to the initial message sent.

Interpreting messages in the correct context is another challenge for chatbots in general. Taking into account the textual context surrounding the conversation is not trivial, however, it is possible. AI chatbots are generally better at contextualizing as they can use a portion of the entire conversation history to generate a reply. Research shows that most open-domain AI chatbots lack long-term contextual information [8], but that this can be improved by strategies specifically designed to boost information content like GPT-2 does [9] [6]. Rule-based chatbots can also be contextually aware of their dialogue history [10]. This contextual awareness allows them to respond correctly when the user refers to a different subject than is expected based on where the conversation currently is. However, rule-based chatbots are not fitted for open-domain applications, and thus managing its context is much more simple.

It is clear that techniques exist that allow chatbots to understand the textual context and adapt to them. However, in conversations between people, there is often an implicit context that is related to their surroundings, environment and personal knowledge of each other. If someone asks if they should wear a jacket when going outside, the response will vary depending on the season. Another example is if someone says they are bored and asks what they should do. Depending on if they are 10 or 80 years old, the suggestion would be different. This context could be implemented in a rule-based chatbot using a decision tree, but the number of rules and responses required will grow exponentially when you want to use a combination of variables to consider while generating a reply. Chatbots have no way of considering this information unless all these details have been mentioned in the conversation.

The research question surrounding this problem formulated as: is it possible to influence the responses generated by a chatbot by providing it with additional context about the surroundings and its interlocutor? In this thesis, an AI/rule-based hybrid chatbot is proposed, which can update its beliefs throughout multiple separate conversations and use information originating from contextual sensors to influence replies generated by the chatbot. This new hybrid chatbot is developed, and tests have been conducted to evaluate its performance compared to the stand-alone DialoGPT2 chatbot. The idea of a hybrid chat consisting of two types of chatbots has been proposed by Gapanyuk et al. [11], but their proposal uses different kinds of chatbots, lacks proper testing, is not focused on being contextual-aware, and is not suited for open-domain dialogue.

This paper will add new knowledge to the field of AI dialogue systems by demonstrating how open-domain AI chatbots can use context that originates from outside of the dialogue to influence responses. Additionally, this paper proposes an architecture that allows for an additional layer of quality control to reduce the amount of inappropriate and irrelevant responses sent to the user. In the area of testing chatbots, a new testing metric is suggested that is usable with large data sets without the need for additional human evaluation.

In this thesis, the architecture of the hybrid chatbot is explained. All of the various modules are talked about individually, explaining their inner workings. Then the evaluation methods to test the performance of the hybrid chatbot will be elaborated. This is followed by the results of the previously mentioned evaluation methods. The discussion part contains an elaboration of the results and the proposal of possible future research topics is. Lastly, a conclusion is drawn.

# 3 Methods

## 3.1 Implementation

### 3.1.1 Architecture

The hybrid chatbot is designed to be modular, as this ensures that any parts of the system can be replaced or improved without directly modifying the other modules. Modularity is preferred because DialoGPT is not the only AI chatbot and, depending on the use case, anything can be a contextual sensor. By creating a modular architecture, the hybrid chatbot can be used in a multitude of different use cases that use different kinds of AI bots, sensors, or knowledge bases.

Figure 1 shows the information flow between the modules. The chat manager is at the center of it all and provides the logic which enables the other modules to work as one unit.

The different chat modules (DialoGPT and rule-based chatbot) are not aware that they are part of a bigger system, as the chat manager interacts with them as if a user requests a response. How this works will be explained more in-depth later on. This way of interacting with the chatbots allows for more chatbots to be added when desired.

### 3.1.2 User interaction

The user can interact with the chatbot through the popular messenger app Telegram (https://telegram.org/). Telegram provides a familiar chat experience for the user that is similar to any other modern chat application. This familiarity ensures that there is no learning-curve to chat with the chatbot. Telegram simultaneously provides an extensive API for developers that allows for creating bots and the ability to send messages from these bot accounts. The Telegram module in the architecture is simply a passageway for messages from the chatbot

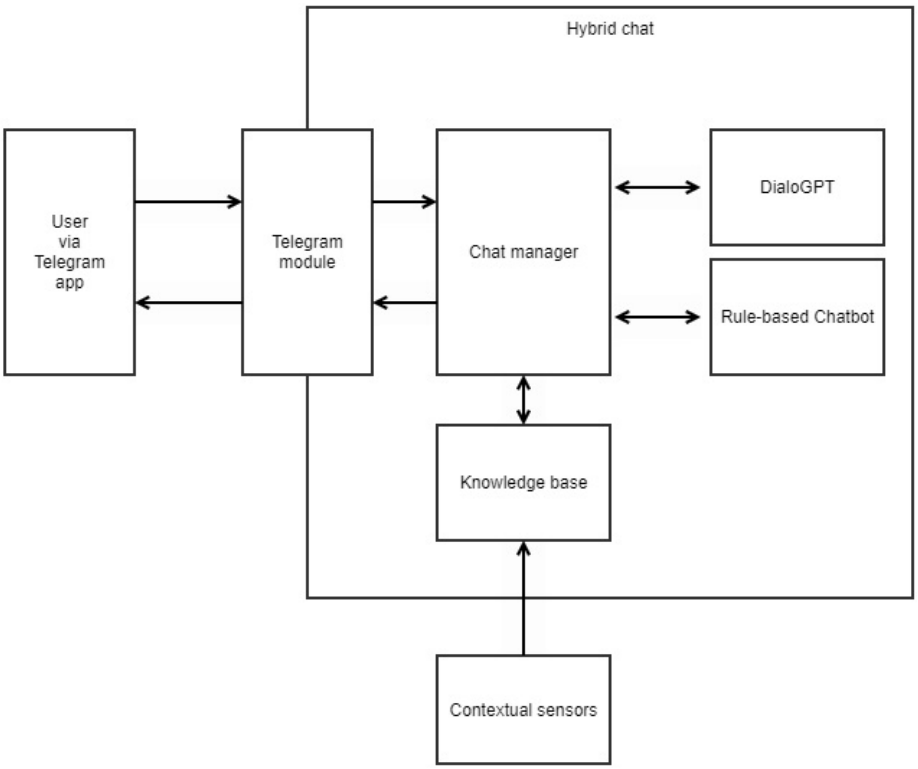


Figure 1: Hybrid chatbot architecture

to the user and vice versa. This module allows for any other chatting module to take its place without changing the chatbot itself.

### 3.1.3 Chat manager

The overall idea of the hybrid chatbot is that the DialoGPT and rule-based chatbot seamlessly work together. The user has no knowledge about the fact that they are, in essence, talking to two different chatbots in the same conversation. The chat modules are not aware of one another either, but they still have to generate messages based on the entire chat history and not only the responses they generated themselves. The chat manager facilitates all of this. To summarize, there are two main functions that the chat manager provides: deciding which message from either chatbot is sent to the user and secondly, the manipulation of messages and the chat history.

When the user sends a message to the chatbot, the chat manager can prompt all under-laying chatbots to generate a message based on the user message. The process that decides which of the generated messages is send back to the user can be as complex as needed. However, in this project, a simple approach is used:

the message from the rule-based bot is used when one of its trigger phrases is detected in the user's message. Otherwise, the machine learning bot is utilized instead to generate a response. As this is a prototype, this simple approach is sufficient to show the workings, and it is easy to conceptualize how to expand this concept. An example of a more complex yet still simple decision-making layer would be the exclusion of trigger phrases depending on the number of messages that are sent by the user. This would prevent 'greeting'-phrases from triggering when the user sends a message such as "Hey! don't say that!" on the fifth turn. The message contains the word 'hey'; although, it is not used as a form of greeting at this stage in the conversation. A much more complex decision-making process, which could potentially be used here, is determining the spatial distance between the user utterance and the responses generated by the various chatbots to determine which message is more appropriate to send to the user. In this project, spatial distance is used inside of the DialoGPT module to improve responses. A more in-depth explanation of this is given in the corresponding subsection later on.

As mentioned before, the manager provides a way to manipulate chat history, which is crucial for the generation of messages by the chatbots. This manipulation allows both chatbots to generate a response based on the entire chat history regardless of whether some messages in this conversation were generated by the other chatbot. Furthermore, this manipulation can be used to provide the chatbots with information from the contextual sensors. It is done by logging all the messages that are sent by the chatbots and the user. When consulting the two chatbots for a reply message, the chat history is transformed in the correct format corresponding to the rule-based or the DialoGPT model. To do this, the default DialoGPT module that is used, called gpt2bot [12], was altered. Instead of the DialoGPT chatbot managing its own history, it now received a turn history as a parameter when prompted to generate a response. This way of managing the chat history also permits the injection of information originating from the contextual sensors.

Because all the messages are handled at a single point, generated responses can be altered and even faked. Information can be obtained from the knowledge base and then, by modifying the response generated by one of the chatbots, be added into the message. This process is called injection. The injection of contextual sensor information adds an extra layer of intelligibility to the responses of the chatbot, as they contain current and accurate statements. An example of injecting contextual sensor data into a message is shown in Figure 2.

The injection procedure seems rather simple, but it enables the DialoGPT to generate more pertinent messages as the conversation progresses. When, in a future moment, the DialoGPT is prompt to generate a reply to the user, it will receive a chat history that includes this information. This information often influences the generated reply noticeably, however, not always positively or expectedly. The reason for this is related to the training set on which the AI chatbot is trained. The DialoGPT parameter for the maximum turn history to consider is set to four (default is two) to ensure that the response to an injected

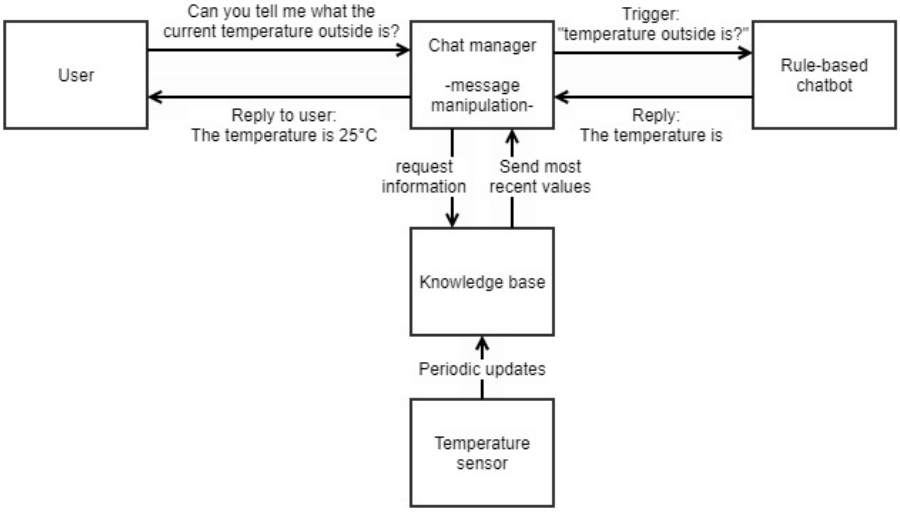


Figure 2: Flow of information injection

message is not always alike.

The injection of messages is not limited to the procedure shown in Figure 2. The chat manager makes it possible to manipulate the chat history without actually sending these messages to the user. An example of this would be the following: instead of the user asking for the current temperature it would ask: "Is it cold outside?" and there has to be a corresponding trigger for this question. However, instead of setting some artificial threshold deciding if the chatbot says yes or no, we inject the sentence "The temperature is *Current-value*" into the chat history as if the chatbot has said this before. This current value is based on real-time sensor data. Consecutively, we request the model to generate a response to the user's question based on this altered chat history. While the user never receives the message, the model will still use this information to generate a response to the question of whether it is cold outside.

In this project, there are three information variables that all modules support: the interlocutor's name, the current temperature, and the weather type (e.g., raining, storming, sunny). This includes triggers for the rule-based chatbot, ways to manipulate the sentences provided by the rule-based bot, and sensors that provide the corresponding information.

The temperature and weather-related information are provided by the OpenWeatherMap API (https://openweathermap.org/). To avoid calling the API every message a user sends about the weather, the knowledge base will act as a mediator. It will only call the API if the current known information is outdated, as OpenWeather only updates its weather model every ten minutes.

To gain knowledge of the interlocutor's name, a series of triggers for the rule-based bot is used. Greetings-triggers are made because most conversations

start with a simple greeting such as "Hey" or "Hello". When triggered, it generates a reply along the lines of "Hey! What's your name?". The chatbot knows that it just asked that question, so the next message from the user will most likely contain their name. The structure of their reply sentence can be anything, and thus a Named Entity Recognition library (www.spaCy.io) is used to extract their name and save it to the knowledge base. If the user starts the next conversation with a greeting, it will create a personalized greeting reply, which includes the user's name. An experiment was held for the evaluation part of this project. In this experiment, 12 participants talked to both the hybrid and a default DialoGPT chatbot. The named entity recognition does not have knowledge about every single name in existence. To make sure that everything went smoothly during the experiment, the names of the participants were added to a custom list used by the named extraction algorithm.

#### 3.1.4 DialoGPT

The dialogue response generation model used in this project is DialoGPT [6]. During the early stages of this project, no official decoders for the model were published yet. Hence, a third-party model decoder called gpt2bot [12] is used as a wrapper to interact with the model.

DialoGPT attains a close to human performance in single-turn dialogue settings [6]. Nevertheless, there is still room for improvement, especially in multi-turn dialogues. Two of the improvements suggested in this paper are regarding the repetitiveness of replies and the countering of training data bias.

In multi-turn dialogues, the DialoGPT model has the occasional tendency to get stuck in generating the same reply repeatedly. When the chatbot is stuck in one of these loops, it is very frustrating for the user. To break out of this loop, the user is required to drastically change the topic of conversation. And even then, there is no guarantee that it will break out. During the experiment, various conversations of different users got stuck in one of these loops. In Table 1 the frustration of the users is easily recognized.

The original paper [6] does not mention this phenomenon or explain why it might occur. However, a likely cause for this behavior is the feedback loop that takes place when querying the model, as it uses the previous messages as context reference for the new reply.

By default, the model generates just a single message. However, it has the ability to generate multiple, mostly different, messages. Generating multiple sentences does require a way to decide which of the messages is sent to the user. The transformation of a word into a word vector gives it the possibility to be put on a point in a multi-dimensional space. Using a pre-trained model, like fastText [13][14], you can compare the distance between two words to determine their similarity. The distance between the word vectors is called their spatial distance. As previously mentioned, spatial distance is used to determine which of the three DialoGPT generated messages is chosen to send to the user. The spatial distance between the user message and the three generated sentences is calculated, and the generated sentence with the least distance is sent. Even though a word can

| sender | message |
|---|---|
| bot | I love you too! |
| user | My hangovers are really bad so I probably shouldn't drink too much |
| bot | I love you too! |
| user | I don't love you |
| bot | I love you too! |
| user | Can you say something else? |
| bot | I love you too! |
| user | So, if you say that one more time, I'll stop talking! |

| sender | message |
|---|---|
| bot | I love lamp. |
| user | i love hagrid |
| bot | I love lamp. |
| user | lavalamp? |
| bot | I love lamp. |
| user | i don't |
| bot | I love lamp. |
| user | what do you have for dinner? |
| bot | I love lamp. |
| user | uwgfgre |

Table 1: User frustration caused by repetitiveness

have a different meaning depending on the context, the collection of words in the form of a sentence will provide clues about its meaning, and the same is expected regarding word vectors [15]. Unlike word vectors from unsupervised models (i.e., word2vec [16]), fastText word features can be averaged together to form good sentence representation [17]. By taking the mean of all word vectors in a sentence and comparing them to those of other sentences, it is expected that sentences that are contextually most alike, are also spatially closest. An example of how words are placed in a three-dimensional place is shown in Figure 3

In addition to the context determination of a sentence, the spatial distance score generated can also be adjusted by other algorithms. This allows for a system that punishes the use of certain words. DialoGPT is trained on a set of data that was gathered from the website Reddit.com. This gathering process results in the overrepresentation of words that are Reddit specific jargon. In this project, sentences were punished by increasing the spatial distance score if it contained any of the following words or phrases: *reddit*, *subreddit*, *upvote*, *downvote*, and *the temperature is*. The last phrase is because when the hybrid bot would send a sentence with the current temperature, the DialoGPT model tented to contradict itself in the next message with a different value.

The spatial distance is punished severely when the message generated is the exact same as the users. This punishment is because if a message is identical, the spatial distance is always 0, and it would repeat the user's answer. While not done in this project, the same simple technique could be applied to avoid sending the same message more than once consecutively.

### 3.1.5 Rule-Based chat module

The rule-based chatbot implemented in this project is quite basic, with merely a small amount of triggers. The small number of triggers can be extended

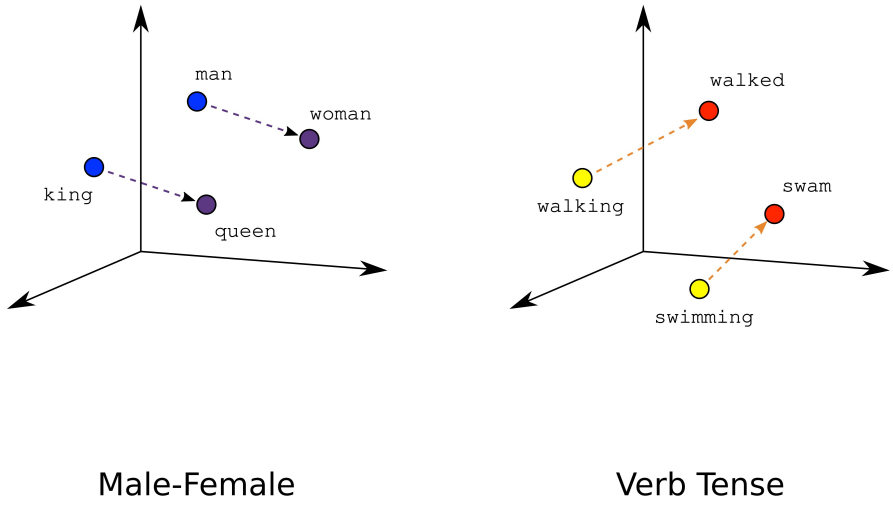


Figure 3: Spatial Distance between words [18]

with ease, but this set sufficiently demonstrates the possibilities. The modules workings are based on the rule-based chatbot design by S. Imran [19]. Pattern matching is applied to detect certain words and phrases in the user's sent messages.

Three types of triggers are present in the chatbot: greetings, weather type, and temperature. A list of synonym words and phrases are set for each of the subjects. When one of these synonyms is detected in the message sent by the user, a predetermined response is returned. The 'greetings' trigger is set off by words like "Hello" or "Hey". The trigger is responsible for returning a greeting and requesting the user's name. If the interlocutor's name is already known, the greeting response is personalized by including their name in the message. For some triggers, there are multiple differently structured replies. When one of these is triggered, a random response from this list is chosen. This randomization is to avoid repetition if a user hits the same trigger multiple times.

Questions about the type of weather or the temperature trigger the querying of the weather API. In turn, this information from the API injects into predetermined sentences that fit the question asked. For example, a user can ask: "What is the weather like currently?". To which the chatbot will reply with: "I see -blank- outside.". The chat manager will then inject the value it receives from the contextual sensor into the message to ensure that the user gets accurate information.

The complete list of trigger phrases and their responses that are used by this

chatbot can be found in Table 2 and 3.

| Category | Triggers |
|---|---|
| Greetings | 'hello', 'hey', 'hi', 'bonjour', 'good day', 'heey' |
| Weather type | 'weather today', 'current weather', 'weather like today', 'weather like currently', 'weather at the moment' |
| Temperature | 'current temperature', 'temperature like today', 'temperature today' |

Table 2: Rule-based chatbot triggers

| Category | Responses |
|---|---|
| Greetings (name unknown) | 'Hey! What's your name?', 'Hello, what's your name?' |
| Greetings (name known) | 'Hey X, what's up?', 'Hello X, how are you?', 'Good day X, what's going on?', 'Great to see you again X, what's on your mind?' |
| Weather type | 'I see X outside.' |
| Temperature | 'The temperature is X.' |

Table 3: Rule-based chatbot responses

#### 3.1.6 Contextual sensors

The definition of context used in this paper is "any information that can be used to characterize the situation of an entity" [20]. The context surrounding a conversation is not always apparent when communicating via text messages, as often implicit information is missing. Providing additional information originating from sensors about the current environment or situation can improve the quality of a conversation.

Sagl et al. [21] provide a list of sensors and their contextual information for smart cities. Almost all of these can also be used to provide contextual information during a chat conversation when the topic is related to that specific field. To show the functionality and test the potential, a weather sensor (in the form of a live weather API) is used. Sagl et al. [21] also mention mobile phone sensing, and this was heavily considered during this project. There even was a working prototype with an Android phone that sent data from its sensors (i.e., ambient noise level) to the knowledge base. But this is much more invasive to the user's privacy so, it was decided not to pursue this idea any further. However, when privacy is not an issue, a person's mobile phone is an extraordinarily rich source of contextual information about the user and their surroundings.

Contextual information can also be gathered from the conversation itself and saved in the knowledge base to be used in a future conversation. While usually

sensors are associated with measuring a physical property, a virtual sensor can provide information on a software level. In this project, a virtual sensor is used to analyze the chat conversation and extract the user's name. This information is used in future conversations, in the form of a personalized message to indicate that the chatbot is aware of to whom it is talking.

All the information gathered in this system, both internally through conversation analysis and externally through contextual sensors, is stored in a single knowledge base. The knowledge base is also designed with modularity in mind. It should support any yet unspecified information structures for future sensors. To accomplish this, a MongoDB database is used. Unlike traditional relational databases, MongoDB does not require a predefined data structure. It is also possible to use a conventional relational database with a table that contains only two columns, acting as a key-value store. However, this does possibly limit future expansion.

## 3.2 Evaluation

There is no single standardized testing technique by which all chatbots can and should be tested and compared. The evaluation metrics should be picked based on the use case of the chatbot [22]. The evaluation metrics used in this experiment are chosen before performing the experiment (with the exception of the Consecutive Repetition Rate), and they are based on the qualities that are deemed important. These qualities are contextual awareness, varied responses (non-repetitive), intelligible responses, and the ability to stay on topic.

It is desired that the hybrid chatbot does not perform worse than the default GTP2 chatbot. It is considered a success if the hybrid chatbot performs the same on the chosen metrics while still being influenced by the contextual information. This result would mean that the injection of information into the conversation does not negatively impact the performance of the chatbot.

A qualitative approach was chosen for the testing phase as this would give a large number of messages to analyze but would still be manageable to be evaluated manually. By reading every message during the analysis, new patterns or remarkable occurrences can found, even if they do not fit in any of the evaluation strategies. This methodology partially resulted in the creation of the Consecutive Repetition Rate. Additionally, some noteworthy exchanges were found and are mentioned in the discussion part.

The experiment was conducted by twelve participants and was entirely online. The participants were between the ages of 20 and 30 years old and had to be able to have a basic conversation in the English language. Due to a global pandemic that occurred during this project, the participants were picked from the author's social circles, mainly: university classmates, work colleagues, and other acquaintances. None of the participants had any prior knowledge about the project or what the goal of the experiment was.

All the participants had three conversations with each of the two chatbots. A conversation is sending, and thus also receiving ten messages. The participants were asked to send a single message at a time and wait for a reply before sending

the next message. It was chosen in advance that participants: 1,2,5,6,9, and 10, would have their first conversations with the hybrid chatbot. The other half of the participants started by chatting with the default DialoGPT chatbot. After chatting with their first chatbot, they were asked to fill in a survey about their conversations. After this, they chatted with the other chatbot and filled in the survey about those conversations. To ensure that the chatbot is tested in an open-domain setting, the participants were told they could talk or ask about anything. They were not instructed to talk about any specific subject. Nevertheless, they were reminded that anything they said would be read and analyzed afterward.

The analysis of the conversations was performed by the writer of this paper and a second assessor who is knowledgeable about the workings of chatbots. The experiment and the procedures to ensure the privacy of participants were conform to the standards set by the Ethical Review Board at the Eindhoven University of Technology.

Ten of the participants triggered at least one of the rule-based chatbot phrases, resulting in the use of both the rule-based and the DialoGPT bot in at least one of their chats. Eight of the participants hit both phrases regarding their names, resulting in getting a personalized response.

### 3.2.1 User survey

| # | **Question** (scale: 1 - 5) |
|---|---|
| 1 | How do you feel about speaking with this chatbot again? |
| 2 | Do you feel the chatbot understood what you said to it? |
| 3 | Did you find the responses of the chatbot appropriate to your messages? |
| 4 | How aware of the context of the conversation was the chatbot? |
| 5 | Did you feel the responses of the chatbot were repetitive?<br>1 being very repetitive, 5 being not repetitive at all. |

The obstacle that is the lack of one standardized testing technique arises again when considering which questions to ask during a user survey. While the other metrics provide a performance measure that can be objectively calculated based on the chat logs, a user survey provides an insight into how the user feels about their conversation. The first question can be considered standardized, as it originates from the Alexa Prize competition, which is held annually [23] [24]. The other questions are constructed to measure the user's opinion regarding the previously named qualities that are deemed important for the performance.

The results of the user survey can be used to paint a broader picture, especially when it is combined with the other upcoming metrics. For example, it can quantify the user's feelings about the chatbot's repetitiveness by comparing the results from question 5 and the consecutive repetition rate. Additionally, the survey can verify the results of the other metrics.

### 3.2.2 Response Error Rate

One of the qualities that are deemed important for a response is its coherency to the user's message. The Response Error Rate (RER) [24] can be calculated from a set of conversations and calculate an overall coherence score. This value is used to compare the performance of different chatbots.

$$\text{RER} = \frac{\text{Number of incoherent responses}}{\text{Total number of responses}}$$

A response is considered coherent when it is relevant to the last message sent by the user. Any other messages are not taken into account [24]. For example, both replies in Table 4 are coherent, even though the chatbot contradicts itself. A lower RER score indicates a better performance.

| sender | message |
|---|---|
| user | Do you like flying? |
| bot | I love flying! |
| user | I'm afraid of flying. |
| bot | Me too, I hate flying. |

Table 4: Contradicting statement by the chatbot

### 3.2.3 Conversational depth

The coherence of replies measured by the RER lacks the ability to evaluate the conversations in their entirety. A natural conversation generally consists of a back and forth on the same domain. Additionally, messages can refer to a previous utterance that happened more than one turn ago.

The amount of messages sent back and forth on the same domain during a conversation is called the conversational depth [24]. By calculating the average conversational depth over a set of conversations, it can become clear how often the user or chatbot tends to change subjects. In general, for generative conversational chatbots, a larger conversational depth means better performance.

### 3.2.4 Consecutive Repetition Rate

Initially, only the conversational depth and the RER were chosen as objective performance measurements. However, after performing the experiment and talking to the participants in an informal setting, many participants made the same remark: it is very frustrating when a chatbot repeats the same messages over and over again. This insight inspired the creation of a new metric: Consecutive Repetition Rate.

$$\text{CRR} = \frac{\text{Number of repeated responses}}{\text{Total number of responses}}$$

The CRR scale is from 0 to 1, the same as the RER score. A score of 0 indicates that a repeated response never occurs, and a score of 1 would mean that a bot returns the same message continuously. Therefore, a low CRR score is desired.

#### 3.2.5 Contextual sensor injection

It would be unreliable to evaluate the sensor injection functionality on the chats from the experiment with participants. It is highly unlikely that a significant amount, if any, of the participants hit the same injection trigger multiple times. As an alternative, this functionality is tested in a brute-force manner.

The goal of the experiment is to observe whether the responses generated by the chatbot are altered solely based on a different injected value. To test this, the same set of questions, which trigger the rule-based bot to insert contextual data into the conversation, are asked multiple times. By mocking the contextual sensors data, a range of values can be tested to see if it changes the generated reply. Table 5 shows a sequence of sentences that are used to test this. The sequence is repeated 100 times. During this, the only variable that is changed is the current temperature; starting from 50 °C to -50 °C. If the second response from the chatbot is not identical for 100 messages, it proves that the DialoGPT model is influenced solely by the injected value from the contextual sensor. Preferably, the second response would make sense based on the current temperature, but this is not required. The DialoGPT model is not trained to *understand* the concept of hot and cold, and thus the message would not have to make sense in order to prove that the model is influenced by the contextual sensor.

| sender | message |
|---|---|
| user | What's the current temperature outside? |
| bot | It's currently X °C. |
| user | Would you say it's cold? |
| bot | -RESPONSE- |

Table 5: Contextual sensor injection testing example

# 4 Results

## 4.1 User survey

First, an f-test is used to determine whether equal variances can be assumed, and the results showed that this is the case. Then a two-sample t-test assuming equal variances was conducted on the results gathered from the user survey. By looking at the survey results, it would seem that the hybrid chat did perform better overall. Unfortunately, by taking a significance level of 0.05, it can be concluded that none of the results of the questions have a statistically significant

| # | **Average score survey results** | Hybrid | Default | p-value |
|---|---|---|---|---|
| 1 | How do you feel about speaking with this chatbot again? | 3,08 | 2,91 | 0,75 |
| 2 | Do you feel the chatbot understood what you said to it? | 2,83 | 2,42 | 0,28 |
| 3 | Did you find the responses of the chatbot appropriate to your messages? | 2,83 | 2,58 | 0,51 |
| 4 | How aware of the context of the conversation was the chatbot? | 2,75 | 3 | 0,52 |
| 5 | Did you feel the responses of the chatbot were repetitive?<br>1 being very repetitive, 5 being not repetitive at all. | 2,83 | 2,25 | 0,15 |

Table 6: User survey results

difference. However, this does mean that the hybrid chatbot did at least not perform worse than the DialoGPT chatbot and, as previously mentioned, is seen as a positive result.

## 4.2 Response Error Rate

| **Coherence evaluation** | Hybrid | Default |
|---|---|---|
| Coherent responses | 248 | 199 |
| Incoherent responses | 121 | 167 |
| **Response Error Rate (RER)** | **0,33** | **0,46** |

Table 7: Response Error Rate

A low RER is desired, and a reduction of 0,13 shows that using spatial distance for determining the response does improve the RER. However, a RER of 0,33 is still high compared to a regular human-to-human conversation, in which it is unexpected nor appreciated to receive, on average, an incoherent response every third message. This score leaves much room for improvement. By reading through thousands of responses that were not picked because of too much spatial distance, it becomes clear that the model regularly generates a response that is more suitable to send than the sentence with the least spatial distance. In the discussion part, an alternative to spatial distance is suggested.

## 4.3 Conversational depth

The average amount of messages sent on the same topic for the default bot is 4,33. For the hybrid bot, this is 5.68, which is an improvement of 1,35. The improved attention span can be partially attributed to selecting the message with

| **Conversational depth** | Hybrid | Default |
|---|---|---|
| **Average Conversational Dept** | **5,68** | **4,33** |

Table 8: Conversational depth

the least spatial distance. However, the greetings trigger that is often activated at the beginning of a chat with the hybrid chatbot guides the conversation in a back and forth about what is on the user's mind. This guidance often results in a sequence of messages on the same topic, which contributes to the improved conversational depth score.

## 4.4 Consecutive Repetition Rate

| **Consecutive Repetition Rate** | Hybrid | Default |
|---|---|---|
| Repetitive responses | 19 | 69 |
| Non-repetitive responses | 350 | 297 |
| **Consecutive Repetition Rate (CRR)** | **0,05** | **0,19** |

Table 9: Consecutive Repetition Rate

When a chatbot's reply is not completely coherent, the user starts to lose trust in whether the chatbot understands the user. Receiving the identical message consecutively, the user will lose trust in the chatbot even quicker and will get annoyed, as shown in the examples of Table 1. The consecutive repetition rate shows that the default chatbot is nearly four times more likely to send an identical message consecutively than the hybrid chatbot. This improvement can be attributed almost solely to the use of spatial distance to determine the response.

## 4.5 Contextual sensor injection

Table 10 contains the six conversations that test the effects of contextual sensor injection. As previously mentioned, the user messages are automated, and the conversations were run multiple times, each time injecting a different value from the value range column. This testing method is preferred over asking the participants to converse specifically about the weather. This way, the same messages can be sent repeatedly, and the only changing factor is the data originating from the contextual sensor. When the response from the chatbot is different, there are no other factors that could be responsible for this. Additionally, the rule-based bot would have to be much more extensive to capture all the different ways someone would try to start a conversation about the weather. All this would make testing this part with the participants a much less effective and efficient way to determine whether the contextual sensor data does influence the messages generated by the chatbot.

| Conversation | Value range | # Unique responses |
|---|---|---|
| user: What's the current temperature?<br>bot : The temperature is X °C<br>user: Would you call it warm? | -50°C to 50°C | 48/101 |
| user: What's the current temperature?<br>bot : The temperature is X °C<br>user: Would you call it cold? | | 47/101 |
| user: What's the current temperature?<br>bot : The temperature is X °C<br>user: Should I wear a jacket when I go outside? | | 28/101 |
| user: What's the current weather?<br>bot : I see X outside<br>user: What should I wear? | it's snowing, mist, it's sunny, it's drizzling | 9/10 |
| user: What's the current weather?<br>bot : I see X outside<br>user: Should I go for a walk? | it's storming, it's raining, thunderstorms, | 10/10 |
| user: What's the current weather?<br>bot : I see X outside<br>user: Can I build a snowman? | heavy rain, a few clouds, a clear sky | 9/10 |

Table 10: Contextual Sensor Injection

There are three different questions related to the topic that is asked for both the temperature and weather type. The difference in weather types influences the replies to be more different in comparison to the temperature, as shown by the number of unique responses presented in Table 10. This result is expected, as sentences containing a different weather type are changing differently compared to sentences with a changing temperature value, which is only a numerical change. Additionally, it is arguably a good quality of the model. Ideally, there would be a variety of answers depending on the temperature range. It is desired that the bot replies similarly to temperatures that are close to each other such as -40 and -41.

It may seem that the same answers are given to the questions about it being warm or cold, as the questions are so alike, and the amount of unique responses is so close to each other. However, there is not a single response that is identical between the two sets. Some kind of temperature grouping is possibly already present in the model, but this is not certain, and further research could give more insight into this.

Noteworthy is that for temperature values, the largest variance in responses always occurs at the transition from positive to negative numbers. It demonstrates that the AI model makes a clear distinction between positive and negative values. It gives the impression that when the same injection technique is applied to a model that is trained on that specific domain, it can generate

relevant responses using the contextual sensor data.

In summary, it is overwhelmingly clear that this way of injecting contextual information into the conversation does influence the messages generated by the AI chatbot.

# 5 Discussion

## 5.1 General

The desire to use contextual sensors to generate responses, which are relevant to the current situation, led to the idea of a hybrid chatbot. It facilitates an environment that grants control over the underlying chatbots. This way, information that originates from the contextual sensors can be injected into the conversation. Having control over the AI chatbot creates the opportunity to manipulate its chat history to include messages which were not generated by itself. This chat history is then used in the generation of future responses and influenced by the artificially created or modified chat history.

The concept of a hybrid chatbot is not tied to the use of only these two chatbots, and hence the design of the surrounding modules is modular. Regardless of which chatbots are used, the challenges and inner workings of a hybrid chatbot stay the same. Solutions to problems encountered in this project are most likely also applicable to similar challenges regarding chatbots in general.

When generating multiple responses to a single message, a decision must be made on which response is sent back. Within this project, the rule-based chatbot's response is always chosen if a rule is triggered. As there are only a few rules present, its replies will most likely be more relevant in comparison to the response generated by the AI chatbot. The DialoGPT model is set to generate three responses to each input message, and here a similar decision must be made. For these three messages, the spatial distance between them and the user message determines which is send. DialoGPTs default decision-making process is to pick a random message from the three generated responses. This process makes sense if you consider its original application as an open-domain chatbot. This way, the chatbot won't keep repeating the same messages when it receives the same messages. However, in this project, spatial distance is used to get the most contextually related response.

## 5.2 Outcome

The results show that the hybrid chat can facilitate collaboration between the two underlying chatbots without losing any performance compared to the default DialoGPT chatbot. On the contrary, the performance increased. This increase in performance can mainly be attributed to the use of spatial distance in determining which message generated by the DialoGPT model should be sent. However, ten out of the twelve participants activated at least one trigger of the

rule-based chatbot. It shows that this did not negatively impact any metrics, and it even improved the conversational depth.

When taking a closer look at the survey results, it seems that the Likert scale might not have been the best choice of scale, as the participants generally avoided the scores 1 and 5. In the informal conversations which took place after every experiment, almost all participants expressed that they preferred talking to the hybrid chatbot. They could potentially express this preference better if they had a wider ranged scale to judge the performance.

Even though the hybrid chatbot allows for the manipulation and choosing of responses, the biggest challenge still remains: which response is sent to the user? Choosing the response based on spatial distance does improve the response error rate, however, there is still more room for improvement. A RER of 0,33 means that, on average, one in every three messages is not coherent, which is high compared to a human to human conversation and thus can be frustrating to the user. The spatial distance is not entirely to blame the RER, because regularly the AI chatbot does not generate an appropriate response at all. However, occasionally a response is generated that is more fitting than the one eventually sent, and a different message determination technique could improve the RER without modifying the AI chatbot.

The injection of contextual sensor data is successful as it changes the responses that are generated by the AI chatbot. Observed is that some questions posed to the chatbot have much more influence on the generated responses than others. This behavior is the result of the AI model and its training data. The DialoGPT model made a distinction between different temperatures, especially between positive and negative values. It would be interesting to see how a dialogue model trained on data specific to the weather domain would respond to these same questions. The AI chatbot used in this thesis is open-domain, but the contextual sensor provides information about the weather domain. This mismatch limits how well the AI model can generate responses related to the specific topic. Consecutively, it limits how well the results of this testing method show that the model is influenced in a positive and correct manner.

Having common greeting words act as a trigger for the rule-based chatbot can help to guide the conversation in any desired direction. This guidance can occur because greeting words are frequently used to start a conversation, even in conversations with a chatbot. By sending a greeting to the user which contains their name, it assures that the AI bot is also knowledgeable about the user's name in the new conversation.

During the experiment, the importance of training data for the machine-learned model became even more apparent. As the chatbot is trained on Reddit data, the chatbot became fond of certain TV shows and, for an unknown reason, talking in the third person. Some examples of this are shown in Table 11 and 12. The examples originate from the experiment to test the performance of the chatbot. Only the relevant part of the chats are shown. Any names in the conversations are replaced by pseudonyms.

In the first chat of Table 11, the chatbot replies with "I'm the one who knocks!" which is a famous quote from the show Breaking Bad [25]. The second

chat is highly similar to a famous South Park scene. While not exactly the same as in the show, it is clearly based on the "I'm not your friend, buddy" scene [26]. Table 12 gives two more examples of chats in which the chatbot talks in an atypical way while still being somewhat coherent. "Talking in OOC" means talking out of character and is often used in role-playing communities. Remarkably, the chatbot did not start the sentence off with 'OOC:' when correctly explaining to the user what OOC means. In the second chat, the chatbot keeps speaking in the third person but still responds with coherent replies. These results add upon the research done by Zhang et al.,(2019)[6], and demonstrate that the model can generate messages in various linguistic styles and keep using that same style throughout the same conversation.

While many AI enthusiasts would certainly be intrigued by these responses, they are generally not suited as replies to people who might not be familiar with these shows. The hybrid chatbot allows for the penalization of these responses to improve the natural flow of the conversation. With the option to remove these unwanted responses, a new challenge arises as well. Before this experiment was conducted, it was unknown that these responses would even be generated. How can responses be penalized without prior knowledge of their existence? It could be a continuous process of blocking them gradually when they occur. An alternative strategy would be to edit the training data by removing these entries, but, considering that DialoGPT is trained on 147 million conversations-like comment chains, this seems like an impossible task.

| sender | message |
|---|---|
| bot | OOC : I'm not the one asking that. |
| user | Calm down |
| bot | OOC : I'm not the one putting the question mark there either. |
| user | What is OOC? |
| bot | out of character |
| user | Oh did not know that |

| sender | message |
|---|---|
| user | Ah haha, funny guy |
| bot | I'm not your guy buddy |
| user | So do you want to go for some drinks tonight? |
| bot | No thanks, I'm not your buddy pal |
| user | No you are not |
| bot | No, you are not my pal |
| user | But I want you to be my pal |
| bot | I'm not your pal buddy |

Table 11: TV show quotes

In the default state, DialoGPT only allows for the generation of messages based on an initial message from the user. Because the hybrid chat can contain multiple chatbots, it is possible to initiate a chat dialogue instead of waiting for the user to begin. By conversation initiation, the chatbot can suggest the topic of conversation. This initiation allows for user information gathering and the steering of the conversation towards a subject in which the chatbot is knowledgeable. This functionality is useful for applications that rely on regular chats with the user, like a mental health bot that checks in on someone every few

| sender | message |
|---|---|
| bot | Who do you think you are? |
| user | I'm Patrick, who are you though? |
| bot | I'm the one who knocks! I've been waiting for an answer to this! |

| sender | message |
|---|---|
| bot | He nods. Yeah, I'm glad you're here. |
| user | Me too. How is your day so far? |
| bot | He shrugs. It's going okay. |
| user | Good. Who is your friend? |
| bot | He shrugs. A person who looks like me. |
| user | Is he pretty? |
| bot | He nods. Yeah. |
| user | Nice. Does he like look David Beckham? |
| bot | He nods. They look alike. I do too. |

Table 12: Atypical ways of speaking

days.

## 5.3 Future research

In this experiment, the only information sources are text mining of the current conversation and a weather sensor in the form of an API. Depending on the chatbot's domain, there are endless possibilities for providing the chatbot with information in this manner. One prominent idea that arose during this project, but not implemented due to its size and complexity, was the text mining of current events from news websites. The chatbot could ask the user their opinion on a newsworthy event. And if the user has not heard about it, the chatbot could give a summary based on the news article. This functionality would allow the chatbot to suggest talking about current events without ever having to retrain the conversational model.

As previously mentioned, the contextual sensors used in this project do influence the generation of future messages. However, the correctness or appropriateness of these influenced responses is not tested. Future research could test the Response Error Rate of domain-specific AI chatbots in combination with contextual sensors related to that domain. The hybrid chatbot design does not have to change, but only the underlying AI and Rule-Based chatbots require replacement with domain-specific ones. Alternatively, it would be interesting to see how an open-domain chatbot would behave if it had access to a large amount of different kinds of contextual sensors. However, this would be a gigantic project, and it seems like a less efficient way of testing the impact of contextual sensors than the first approach.

Responses generated by DialoGPT2 are quite impressive, but its successor GPT3 is already on its way. Unfortunately, it has not yet been made publicly available, so it could not be implemented in this project. It would be interesting to compare the performance of the hybrid chatbot containing the DialoGPT2 model to the same chatbot containing the GPT3 model.

Using spatial distance to establish which message is sent to the user is not completely simple, but there are even more sophisticated ways to create word vectors. It would be interesting to see if the use of an ELMo [27] would be able to determine the context better and make better decisions. This technique, to put it simply, is a much more complex way of creating word vectors based on the other words in the sentence. The distance between these word vectors is then very contextually related [15].

Testing chatbots is a challenge, and this paper suggests an additional metric to test them on. A future research project could test a larger amount of different open-domain chatbots on the metrics chosen in this paper to see how DialoGPT based chatbots perform compared to other chatbots based on other techniques. Additionally, a broader scale should replace the Likert scale. The number of participants should also be increased so the results can be statistically significant. An alternative or addition to the survey could be to hold semi-structured interviews with the participants. Formal interviews with participants could reveal yet unknown elements that users prefer or dislike during conversations with a chatbot. Additionally, these interviews allow participants to give their preference of chatbot based on their gut feeling, which otherwise might be hard to explicitly describe in words or translate to a scale in a survey.

To help with future research on this topic, the code of the hybrid chat is publicly available on GitHub [28].

# 6 Conclusion

The architecture of the hybrid chatbot is highly modular, and the prototype demonstrates how different chatbots can act together as one. By combining various chatbots, the weakness of one chatbot is alleviated by the strength of another. One of the finest examples of this is the ability to update the chatbot's beliefs without the requirement of retraining the entire machine-learned model. In addition, this architecture allows for the manipulation of responses that are generated by any of the chatbots. This manipulation opens up the possibility for the blacklisting of phrases, fact-checking, or any other procedures to assert quality control over the chatbot's responses.

Contextual sensors can provide additional context to chatbots by manipulating the dialogue history, which, in turn, is used for future messages. It requires having all communication streams managed in a single point, and the hybrid chatbot architecture arranges this. The injected contextual data does influence the responses generated by the AI chatbot, but whether or not the responses improve in relevancy is dependent on the *knowledge* of the model.

The hybrid chatbot outperforms the default DialoGPT bot on every met-

ric chosen in this paper. The metrics were chosen before the start of the experiment based on factors deemed important for the performance of an open-domain, generative chatbot. Namely: contextual awareness, varied responses (non-repetitive), intelligible responses, and the ability to stay on topic.

The usage of spatial distance to determine which of the responses generated by the AI chatbot is sent to the user looks promising. Nevertheless, the response error rate is still relatively high using this technique. A more sophisticated way of choosing responses could significantly improve the user experience when talking to a chatbot.

This paper proposes a new metric on which chatbots can be tested: the Consecutive Repetition Rate (CRR). Frustration often occurs when a chatbot keeps repeating itself, and this is unwanted. The CRR can give an objective measurement of how repetitive a chatbot is. Furthermore, it can be calculated without any human evaluation of the messages, which makes this metric great to use on very large sets of data.

# References


[1] Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[2] A. M. TURING. I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, LIX(236):433–460, 10 1950.

[3] Naveen Joshi. Choosing between rule-based bots and ai bots. `https://www.forbes.com/sites/cognitiveworld/2020/02/23/choosing-between-rule-based-bots-and-ai-bots/`, 2 2020. Accessed: 21-6-2020.

[4] Kumar Shridhar. Rule based bots vs ai bots. `https://medium.com/botsupply/rule-based-bots-vs-ai-bots-b60cdb786ffa`, 4 2017. Accessed: 21-6-2020.

[5] Jenna Alburger. Rule-based chatbots vs. ai chatbots: Key differences. `https://www.hubtype.com/blog/rule-based-chatbots-vs-ai-chatbots/`, 5 2020. Accessed: 21-6-2020.

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

[7] Gina Neff and Peter Nagy. Talking to bots: Symbiotic agency and the case of tay. *International Journal of Communication*, 10:4915–4931, 10 2016.

[8] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. *CoRR*, abs/1605.06069, 2016.

[9] A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[10] Ashish Shubham. bot-context. `https://github.com/ashubham/bot-context`, 3 2017. Accessed: 5-1-2021.

[11] Yury Gapanyuk, Sergey Chernobrovkin, A. Leontiev, Igor Latkin, Marina Belyanova, and Oleg Morozenkov. A hybrid chatbot system combining question answering and knowledge-base approaches. In *AIST*, 2018.

[12] Oleg Polakow. gpt2bot. `https://github.com/polakowo/gpt2bot`, 1 2020. Accessed: 16-12-2020.

[13] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. Enriching word vectors with subword information. *CoRR*, abs/1607.04606, 2016.

[14] fasttext. https://github.com/facebookresearch/fastText. Accessed: 20-1-2021.

[15] Noah A. Smith. Contextual word representations: A contextual introduction. *CoRR*, abs/1902.06006, 2019.

[16] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. Efficient estimation of word representations in vector space. pages 1–12, 01 2013.

[17] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[18] Embeddings: Translating to a lower-dimensional space. https://developers.google.com/machine-learning/crash-course/embeddings/translating-to-a-lower-dimensional-space, 2 2020. Accessed: 29-12-2020.

[19] Saif Imran. Introducing conversational chat bots using rule based approach. https://chatbotslife.com/introducing-conversational-chat-bots-using-rule-based-approach-c8840aeaad07, 10 2019. Accessed: 1-8-2020.

[20] Anind K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, Feb 2001.

[21] Günther Sagl, Bernd Resch, and Thomas Blaschke. Contextual sensing: Integrating contextual information with human and technical geo-sensor information for smart cities. *Sensors*, 15(7):17013–17035, July 2015.

[22] Bayan Shawar and Eric Atwell. Different measurements metrics to evaluate a chatbot system. pages 89–96, 01 2007.

[23] Alexa pricze. https://developer.amazon.com/alexaprize, 2020. Accessed: 4-2-2021.

[24] Anu Venkatesh, Chandra Khatri, Ashwin Ram, Fenfei Guo, Raefer Gabriel, Ashish Nagar, Rohit Prasad, Ming Cheng, Behnam Hedayatnia, Angeliki Metallinou, Rahul Goel, Shaohua Yang, and Anirudh Raju. On evaluating and comparing conversational agents. *CoRR*, abs/1801.03625, 2018.

[25] I am the one who knocks. https://www.youtube.com/watch?v=wMEq1mGpP5A. Accessed: 20-1-2021.

[26] I'm not your friend buddy. https://www.youtube.com/watch?v=iH3K2rkkU7g. Accessed: 20-1-2021.

[27] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[28] Kevin Jetten. Context aware personalised chat bot. https://github.com/ubaer/Personalised_context_aware_DialoGPT, 4 2020. Accessed: 5-1-2021.