Potentials of Chatbots for Spell Check among Youngsters

https://doi.org/10.3991/ijai.v1i1.10999

Jeton Arifi, Markus Ebner, Martin Ebner (✉)
Graz University of Technology, Graz, Austria
martin.ebner@tugraz.at

Abstract—Chatbots are already being used successfully in many areas. This publication deals with the development and programming of a chatbot prototype to support learning processes. This Chatbot prototype is designed to help pupils in order to correct their spelling mistakes by providing correction proposals to them. Especially orthographic spelling mistake should be recognized by the chatbot and should be replaced by correction suggestions stored in test data.

Keywords—Chatbot, spelling, misspelling correction, learning platform

1 Introduction

This publication is about developing and programming a Chatbot prototype for the IDeRBlog learning platform. IDeRBlog is an internet platform that includes a practice database of numerous free exercises licensed under a Creative Commons license. The user of the IDeRBlog learning platform are pupils in childhood who are learning the German language. The current version of the IDeRBlog learning platform helps children to correct spelling errors in texts. The blog entries or essays that are written online by the students are checked for their spelling by an *intelligent dictionary* and afterwards suggestions for correction are offered. Based on these spelling errors, spelling corrections are suggested to the user.

Teachers can access this spelling error analysis for further use in lessons. The developed IDeRBlog Chatbot prototype is intended to provide better correction suggestions for mistakes, while allowing easy interaction with the users, which also encourages them to learn and write. The interaction, input and operation of the chatbot should be very simple for the users. Pupils should also be given the feeling of being able to communicate with a supervisor or tutor who supports them in their learning processes in the best possible way. Therefore, an enhancement of the platform by a chatbot is being considered in order to offer even more motivation and an improved error analysis to the users. In this research work we address the research question: How can a chatbot for children be implemented to assist during their learning experiences in a language learning platform?

2 Introduction of Chatbot

The term "chatbot" consists of the English word "Chat" (interview) and "bot" (abbreviation of "robot"). Therefore, a chatbot is a computer system that allows humans to interact with the computer through natural language-based keyboard or voice recognition. A chatbot works always on repetitive patterns without human intervention and simulates human beings. In doing so, chatbots access a stored knowledge base in which they select actions and answers by finding matches of asked questions with the existing questions or answers previously created by programmers and output them as answers to the questioner.

Chatbots are partially personified and are sometimes used in conjunction with an avatar (human, animal, or mythical creature). With the help of this "living organism" and its naturalness, the use of the computer should be entertaining and effortless [1], [2]. Likewise, chatbots can provide useful services, refer important information and present websites of interest to users in addition to directly answering the question. Thus, the users can find the information they are looking for, with little effort, or the conversation can be guided by the chatbot in a certain direction [3].

Chatbots also perform other tasks besides communication such as customer service in online shops, the management of forums, Internet auctions, information search on the World Wide Web or represent artificial characters in MUD's (Multiuser Dialog). Corporate websites are increasingly being complemented with a chatbot that acts as the virtual host, answering questions about products and services and navigating through the site [4]. The companies hope for stronger customer loyalty and better marketing information through call log analysis. There are also several non-commercial chatbots. These are developed primarily for the fun of experimenting, not least in the hope of creating a truly intelligent program. Every year, the developers of these non-commercial chatbots mainly meet to compete for the Loebner Prize, where the most human-like program is determined in a modified Turing test [5].

The British mathematician Alan M. Turing introduced his essay "Computing Machinery and Intelligence" published in the journal "Mind" in 1950 with the question "Can Machines Think?" [5], [6]. Therefore, the history of the chatbots can be determined back to 1950. In his essay, Turing described the basics of Artificial Intelligence (AI) and proposed to use the Turing test as the benchmark for measuring the intelligence of a computer system [6]. The question „Can Machines Think?“ was finally replaced by the question of the communicative ability of a computer program by Turing. If a computer program succeeds in imitating a person's ability to communicate in natural language the machine should be considered as intelligent. The requirement for this is, that the computer succeeds in independent attempts and permanently deceives.

Even today, 50 years after Turing, there is no agreement in AI research on the significance of the Turing Test, but it seems to serve the Chatbot developers as a relevant orientation. It is their goal to develop a system that passes the Turing Test. In the "conversation" the chatbot should no longer be distinguishable from humans and he must perfectly imitate a human conversation partner. In its original form of Turing, the Turing test was never realized. It was only towards the end of the eighties that a

sufficiently powerful computer was developed that could seriously start practical experiments.

Since 1990, when the American Hugh Loebner launched the "Loebner Prize Competition in Artificial Intelligence," a modified Turing test is conducted each year, with various chatbot programs. The first competition took place at the Boston Computer Museum in 1991, and consequently the Turing Test was also conducted for the first time [5]. Various prize money was defined for the competition. The first untraceable program in the Turing Test will be awarded a gold medal and a prize of $ 100,000. In order to win this prize, the program must be able to deal with audiovisual input (recognition of spoken language, facial expressions, gestures, emotions, etc.) and to convince the jury members [3].

Only bronze medals have been awarded till now, although the performance of the participating programs has improved significantly since 1991 [5]. Every year, several programs take part in the competition. Over the last few years, the ALICE project with Richard Wallace has repeatedly appeared and performed best in the years 2000, 2001 and 2004. The Turing test is just as controversial as the Loebner Prize. Loebner himself sees the promotion of AI research as the goal of his competition, but the majority of scientists are critical of him and his test [3].

2.1 How chatbots work

Today's chatbots are modular stimulus-response systems that compare the linguistic input with an internal sample database and then output appropriate answers. The so-called knowledge database is also today the heart of the chat bot, in which recognition patterns, keywords and answers are stored.

The conversation flow is controlled by the actual program and coordinates the input and output, the activation of the knowledge database and possibly other models such as the output of spoken language. Commercial systems provide an editor that makes it easier for the user to set up and maintain the knowledge database. In the log function, which each Chatbot has, all dialogues are stored and are evaluated by the developers. The individual systems are different in size and flexibility of their "knowledge databases" as well as in the performance of their control programs.

Some chatbots only compare a part of an input to the pattern database, and other chatbots analyze all parts of the input that are combined with the answers. Most chatbots protocol their conversations and propose new keywords or recognition patterns based on them. However, the decision is made by humans whether they are included in the knowledge database or not. This method is called supervised learning.

Some systems independently expand their database based on the protocols. If there is no recognition pattern for the input, some chatbots can also use external databases [4]. The success of a chatbot system depends mainly on the quality of knowledge and the amount of knowledge in the knowledge base. Creating the knowledge base is a time-consuming job. However, by improving the algorithms of the system (for example, improved analysis of inputs), this time-consuming work can be reduced [3].

3 Architecture of Prototype

The IDeRBot-Chatbot is a self-programmed webapp based on a client-server architecture. For the development of the IDeR-ChatBot standard technologies HTML, CSS, JavaScript and the Open Source Framework Java Spring were used. The communication between client and server is done by using the WebSocket protocol [7].

The messages exchanged between the client and the server are transmitted as JSON strings. JSON is a file format that can be used to transfer structured data between client and server and to generate content based on this data [8].

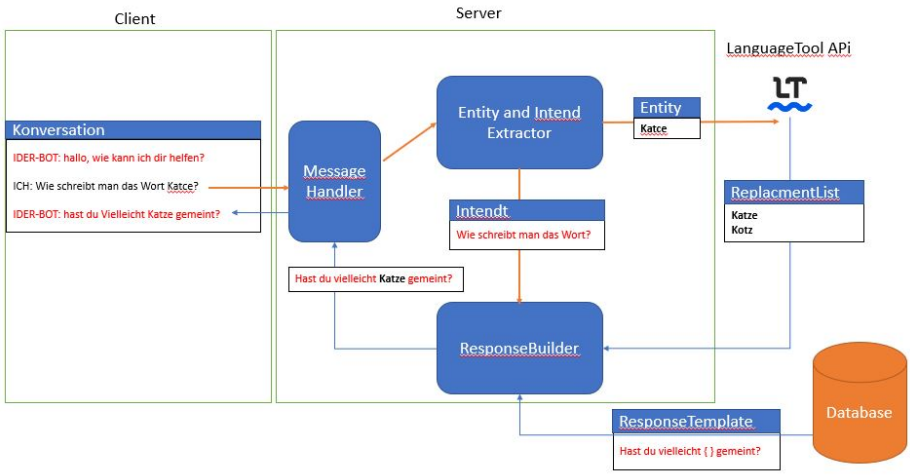

Fig. 1. Architecture of prototype with German dialog

3.1 Client

On the client side is the web frontend user interface, on which the users can write text and messages to the server and receive them from the server.

Figure 2 shows the input and output masks (communication interface) used for the message exchange between the chatbot and the users. The window left labeled as Text Editor displays the input field where the user can write text, but this doesn't mean that the text is correct. Before the user can send text for correction to the tutor or teacher, she/he has the possibility to check the text for misspelling by using Language Tool API, which can provide correction suggestions. In the right side there are windows placed through which user can communicate with chatbot and ask it for corrections if she/he is not sure that corrections suggestion made by Language Tool API are right.

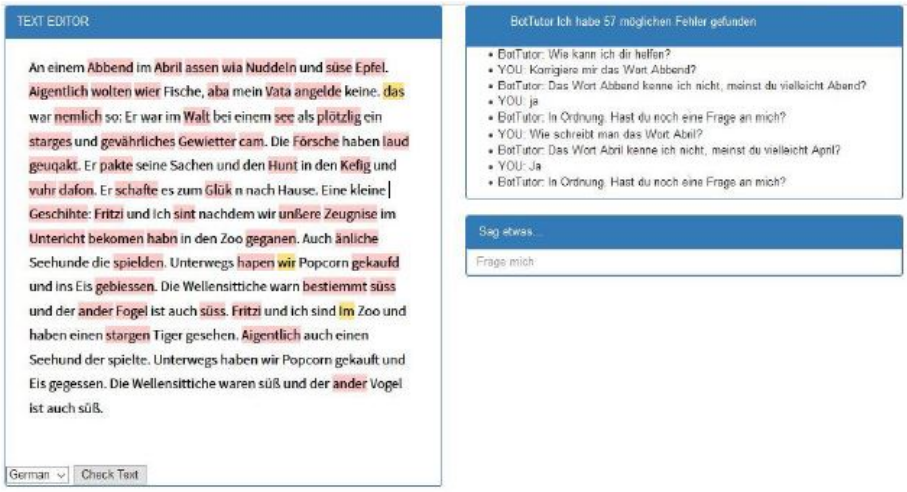

Fig. 2. Webfrontend communication interface for German language

3.2 Backend

To receive messages form users in backend a permanent socket connection must be available for all connected clients. For this implementation the WebSocket protocol of Java Spring Framework was used.[1]

From the received message first full the user's intention must be recognized, this was achieved by using the Levenshtein distance algorithm, the next step is to extract words that contains the spelling mistake from the message. To achieve this the Language Tool API was used. The free open-source application Language Tool helps both non-native speaker and native speakers to recognize spelling and grammar mistakes and supports an appealing writing style [9]. In this Prototype those spelling mistakes are passed from the backend to the Language Tool API endpoint via REST API. The REST API returns a list of possible correction for each spelling mistake as a JSON String to the backend. This list includes, similar words or correction suggestions that can be associated with the spelling error.

The figure 3 shows all processes from receiving messages to generating responses in the backend.

[1] https://docs.spring.io (last accessed on 05.04.2019).

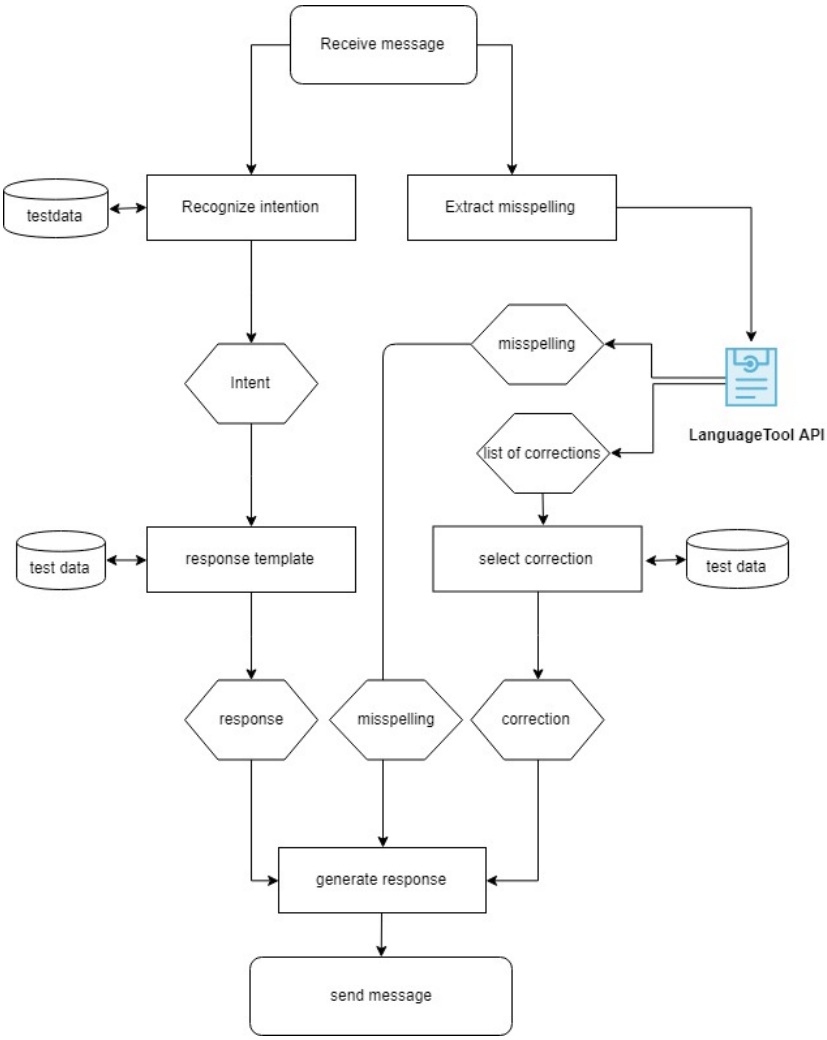

Fig. 3. Backend processes and message flow

3.3 Intent recognition

Recognizing a user intent is the most essential part of the IDeRBlog chatbot. The Levenshtein distance algorithm contained in the Java library (java-string-library) was used to recognize the intent. The Algorithm compares two strings and identifies common patterns when these strings do not match exactly. [10].

To detect user intent from message the distance () function takes the message as a parameter and runs through the list of user expressions from the test data (TrainingPhrases) and selects a TrainingPhrases with lowest distance.

3.4 Extracting misspelling

To extract a word(entity) from intent that contains a spelling error, the following procedure was followed. After the intent of the message has been detected by the Levenshtein distance algorithm, the message should be split into words. The individual words from the message that match the words from the intent would be ignored. The remaining words in the message that do not occur in the intent are candidates that may contain a spelling mistake, for which the user wants to ask the Bot for correction suggestions.

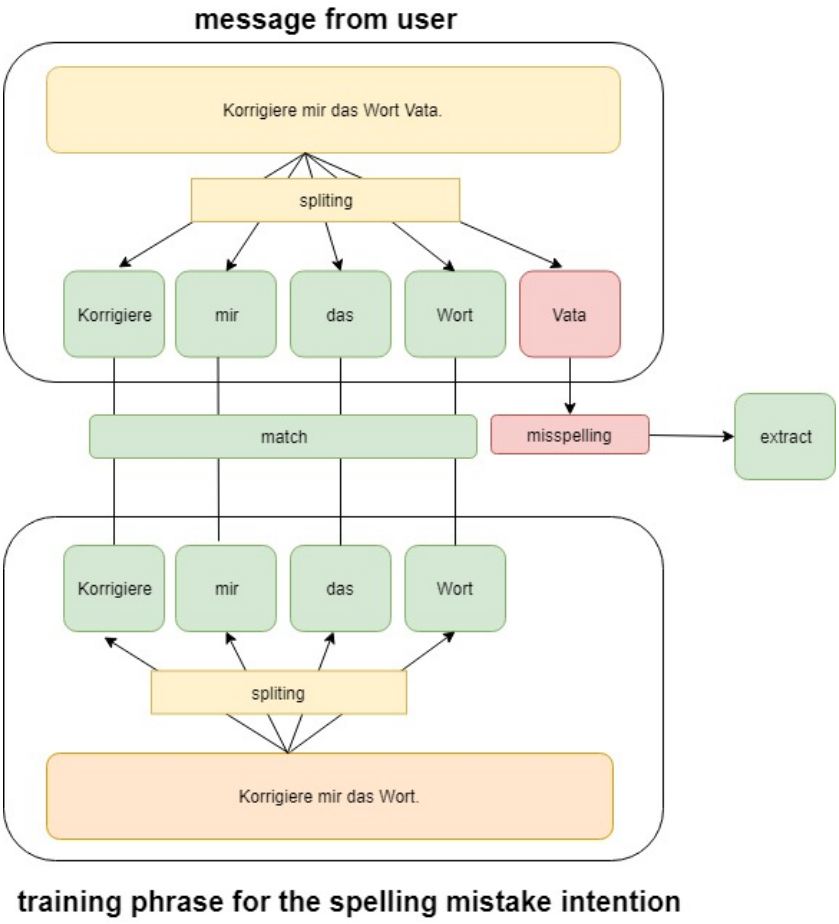

Fig. 4. Extraction of a spelling mistake

The extracted word must now be checked for spelling mistakes. This is done in the next step by using the Language Tool API.

3.5 Proposed corrections

In order to generate correction suggestions for the extracted word, it must first be checked for spelling mistakes. If a spelling error is found, the Language Tool API will also return a list of suggested corrections for the misspelled word. It is possible that wrong correction suggestions are given in the output list, especially for orthographic terms.

The algorithm must compare these correction suggestions with the orthographic terms from the test data and make the right suggestion to the user. To achieve this, the database was filled with test data wich contains some orthographic terms. Each term is associated with a correction suggestion by a frequency factor as shown in Figure 5.

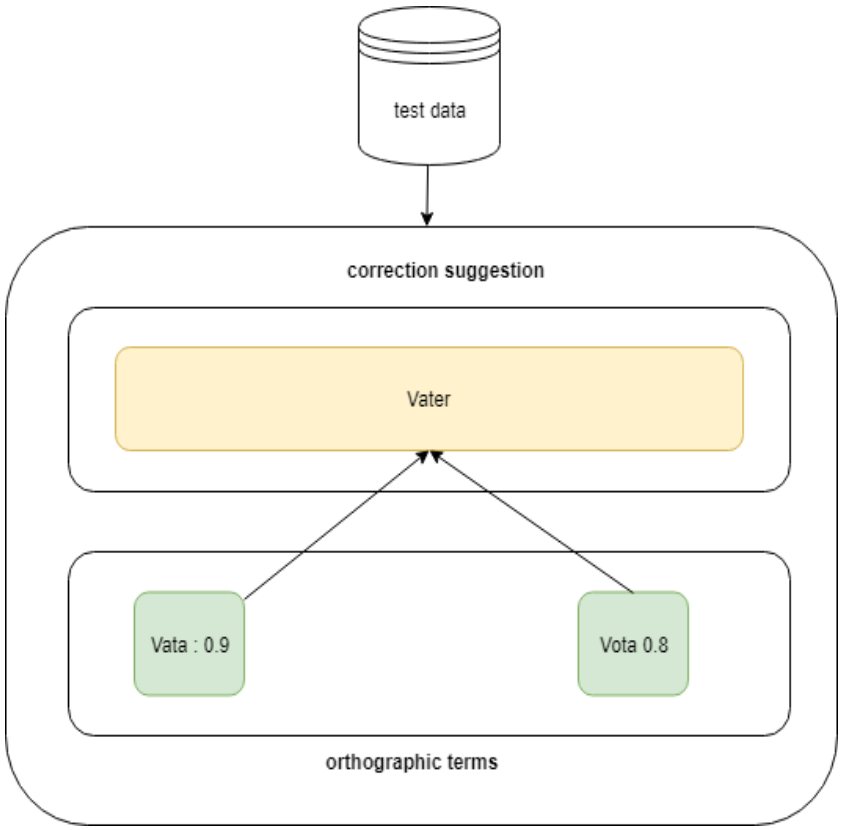

Fig. 5. Frequency factor

The suggested correction with the highest frequency factor is proposed to the user first. If the user accepts the suggested correction, the frequency factor for the suggested correction is recalculated and will be updated in database. This frequency factor ensures more precise suggested corrections for the upcoming proposals.

3.6 Generate message

After the intention of the user has been detected, a response template is automatically selected from the test data in the database. The response template contains the text message (response) and the necessary slots, which serve as placeholders for the spelling mistake and the correction suggestions. The number of slots for a specific response template must be predefined in order to generate an exact response for the user.

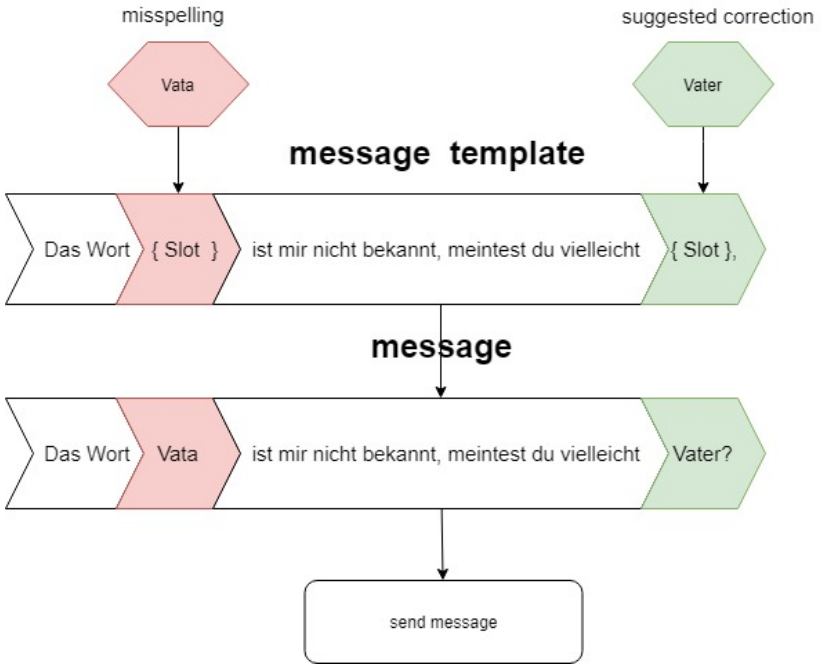

Fig. 6. Message generation using slots

4 Testing the Prototype

Finally, the prototype and the IDeRBlog chatbot were evaluated by experts. Their questions can be separated into three categories: A greeting, small talk, questions about spelling mistakes and questions about spelling mistakes/terms.

4.1 Test category 1: Greeting and small talk

Question to the IDeRBlog chatbot
"Hallo!" (Hello!)
"Wie heißt du?" (What's your name?)
"Wie geht es dir?" (How are you?)

All intentions from the messages were recognized and correct greetings for the test person were generated.

4.2 Test category 2: Questions about spelling mistakes

Question to the IDeRBlog chatbot
"Wie viele Fehler habe ich in meinem Text? "
(How many spelling mistakes do I have in my text?)
"Schreibt man Appril mit zwei p? " (Does Appril have a double p?)
"Habe ich Walt richtig geschrieben? " (Did I spell Walt right?)
All intentions from the messages were also recognized during the spelling error tests. The correct number of errors was detected, and all spelling mistakes were corrected correctly.

4.3 Test category 3: Questions on orthographic errors / terms

Question to the IDeRBlog Chatbot:	Response of IDeRBlog Chatbot:
„Korrigiere mir das Wort **Vata** (Correct me the word Vata?)	Hast du vielleicht **Data** gemeint? (Did you maybe mean Data?)

During the test run, the Language Tool API's correction proposal failed because **"Data"** is not the right correction proposal for the orthographic term or colloquial expression **“Vata”.** Correct it would be “**Vater”** which means “**father”** in German. These orthographic terms or colloquial expressions like **Vata** are used often by children and teenagers when writing texts in schools, especially children who have learned the German language only by listening and do not have German as their native language. After we inserted this orthographic term into our test data, and put this in conjunction with the word father, the next test made the correct suggestion proposal to the user. Each time the user accepts the suggested correction, the frequency factor for this orthographic is increased. This ensures that the suggested corrections are becoming more precise. To conclude, it can be said, that the algorithm seems to work quite well but depends strongly on the amount of test data.

5 Discussion

After testing, the Chatbot IDeRBlog has performed well in detecting intentions and detection of spelling errors has been satisfactory. Correction statements for the selected spelling mistake have been correctly reported to the user. Due to the fact that the Chabot is just a prototype in development, there are still some things that can be done to make the Chatbot smarter:

- Feeding the database with more orthographic terms and linguistic variations, results in a hearing recognition rate
- To increase accuracy, the Chatbot IDeRBlog should also be used frequently.

In summary, it should be noted that the correct generation of responses by a chatbot can is based on the amount of intents that chatbot can detect in a message and the learning ability of the algorithms. At the moment there are no algorithms that can detect intentions without having some training data.

It also turned out that the error detection rate was heavily dependent on the existing test data. A further optimization of the IDeRBlog chatbot would be the error detection of grammatical errors.

6 Conclusion

The aim of this research was to develop a chatbot for the IDeRBot learning platform of the TU Graz and to integrate it. The chatbot should act as a tutor for the spell checker and communicate with the pupils. Through the use of the chatbot, the users should receive an added value in learning the German language. In addition, the interaction should encourage users to write more texts in the German language.

First, an overview of chatbots in general was given by using the literature. It was used to define the term chatbot and to identify tests to determine quality standards. The historical development and functioning of chatbots was also discussed. Areas of application in which chatbots have already been successfully used were also shown. The IDeRBlog chatbots were then discussed by describing the technologies, development environments and libraries used.

The development of this prototype has shown that chatbots can also relieve teachers or tutors in the classroom. These positive experiences encourage us to continue in the research and development in this area. In conclusion it can be sad that, chatbots are quite capable of performing such tasks, but they will never be able to replace a teacher or a tutor completely. Because chatbots are just algorithms and machines do not have the emotional side.

7 Acknowledgements

We gratefully thank the EU-Erasmus+-Project IDeRBlog-ii for financial support and our project partners for the cooperation. The IDeRBlog-ii project are funded by the European Commission in the framework of Erasmus+. IDeRBlog-ii: VG-IN-SL-18-36-047317, 2018-2021.

8 References

[1] Braun, Alexander (2003): Chatbots in der Kundenkommunikation. Springer-Verlag Berlin Heidelberg.

[2] Willmes, Anna (2014): Chatbots: Was ist das und wie funktionieren sie? In: https://familie.wordpress.com/2014/11/24/chatbots-definition-funktionsweise-einsatzgebiete-starken-schwachen/#more-139.

[3] Möbus, Claus; Eißner, Andreas; Feindt, Jan; Janser, Claudia; Krefeldt, Jens; Sieverding, Sven; Sölbrandt, Stefan; Stumpe, Jörg; de Vries, Holger; Willer, Stefan (2006): Web-Kommunikation mit OpenSource. Chatbots, Virtuelle Messen, Rich-Media-Content. Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/3-540-29093-1

[4] Berger, R., Ebner, M. & Ebner, M. (2019) Conception of a Conversational Interface to Provide a Guided Search of Study Related Data. Internationale Journal of Emerging Technologies (i-JET). 14(7). pp. 37-47. https://doi.org/10.3991/ijet.v14i07.10137

[5] Storp, Michaela (2002): Chatbots. Möglichkeiten und Grenzen der maschinellen Verarbeitung natürlicher Sprache. In: https://www.mediensprache.net/networx/networx-25.pdf.

[6] Deshpande, Aditya; Shahane, Alisha; Gadre, Darshana; Deshpande Mrunmayi; Prod. Dr. Joshi, Prachi M (2017): A Survey of Various Chatbot Implementation Techniques. International Journal of Computer Engineering and Applications, Volume XI. In: https://pdfs.semanticscholar.org/8e60/5c49d4a7cba9bf077d97b401ba78aafe693f.pdf.

[7] Weßendorf Matthias (2011): WebSocket: Annäherung an Echtzeit im Web. In: https://www.heise.de/developer/artikel/WebSocket-Annaeherung-an-Echtzeit-im-Web-1260189.html?seite=all (besucht am 25.03.2019).

[8] Ackermann Philip (2018): JavaScript. Das umfassende Handbuch. 2. aktualisierte und erweiterte Auflage. Bonn: Rheinwerk Verlag.

[9] Language Tooler GmbH (2019): Funktionen und Hilfe. In: https://LanguageTool.org/de/ (besucht am 25.03.2019).

[10] Lisbach Bertrand (2011): Linguistisches Identity Matching. Paradigmenwechsel in der Suche und im Abgleich von Personendaten. 1. Auflage. Germany: Springer Fachmedien Wiesbaden GmbH. https://doi.org/10.1007/978-3-8348-9791-6_1

9 Authors

Jeton Arifi is Junior Researcher at the Institute of Interactive Systems and Data Science at Graz University of Technology, Graz, Austria. (e-mail: jeton.arifi@tugraz.at)

Markus Ebner is Researcher at the Department Educational Technology at Graz University of Technology, Graz, Austria. (e-mail: markus.ebner@tugraz.at)

Martin Ebner is head of the Department Educational Technology at Graz University of Technology, Graz, Austria. (e-mail: martin.ebner@tugraz.at). He is responsible for all university wide e-learning activities. He holds an Adjunct Prof. on media informatics (research area: educational technology) and works also at the Institute of Interactive Systems and Data Science as senior researcher. For publications as well as further research activities, please visit: http://martinebner.at

Article submitted 2019-06-07. Resubmitted 2019-07-09. Final acceptance 2019-07-11. Final version published as submitted by the authors.