Developing a ChatBot System for Cultural Institutions using the example ZKM

Alexander Grote
ucdzx@student.kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Jannik Pries
ukesu@student.kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Janna Kraft
uvdvw@student.kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Marie Reger
uleix@student.kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

Patrick Hemmer
uydjn@student.kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

ABSTRACT

The use of chatbots in various business areas as well as in daily life is becoming increasingly popular. These digital assistants can support users by answering questions without human intervention. In this context, they can not only support multiple users at once but also provide a service which is available twenty-four-seven. However, so far, the potential of chatbots is still rarely used by cultural institutions. In order to demonstrate this potential, this paper presents a chatbot system that we developed in close cooperation with the ZKM - Center for Art and Media Karlsruhe. We developed the chatbot with the open-source framework MindMeld. Having predefined the subject area the chatbot covers to exhibitions, artists, opening hours, ZKM background, the journey to the ZKM, greeting, farewell and jokes, we optimised a three-stage classification model with a machine learning pipeline provided by MindMeld. For the front end, we developed a web application with a focus on guiding the user intuitively through the topics covered by the chatbot. To have the possibility to enlarge the training data, we implemented a semi-online learning approach through which user inputs can be gathered and manually added to the training data. Evaluating the chatbot with 50 questions asked onsite by museum visitors, yielded in an accuracy of 74%. After extending the training data through data augmentation, we performed a second testing iteration. This user testing with 150 questions resulted in an accuracy of 77%. The evaluation results indicate that further substantial improvements can be achieved by enlarging the training data and using a semantically well-maintained database.

1 MOTIVATION

A chatbot is a computer program that simulates human conversations via textual interactions. Such programs are of great potential in cultural institutions such as museums [Schaffer et al. 2018].
In this context, the personal interaction of chatbots can enhance the excitement of museum visits by offering personal information, gamification and support. Notably, in the case of teenagers gaining their attention and interest in cultural content is a challenge. Since young people are already highly adapting the use of social media such as online chat platforms, chatbots are a promising opportunity to arise their cultural interest [Boiano et al. 2018; Gaia et al. 2019]. Furthermore, guided tours are already widely spread in museums and used by many visitors. However, most of the current digital visitor guides in museums only offer one-way communication [Gaia et al. 2019]. Chatbots as digital visitor guides in museums could allow the user to ask specific questions and thereby increase personalization. As a digital assistant, a chatbot offers twenty-four-seven availability. The chatbot can respond to any message of visitors individually. Thereby, a museum chatbot has excellent potential to improve the museum experience by accompanying the visitor as an "expert" throughout the whole visit [Schaffer et al. 2018].
In this context, a chatbot not only offers the opportunity to make the museum visit an exceptional experience. It can also provide the user with useful information required to plan a visit such as opening hours, how to reach the museum, or if parking facilities exist.
Besides, visitors will have the opportunity to obtain information about exhibitions from home after a visit to the museum.

The variety of cultural institutions and their visitors, as well as the changing exhibitions that museums show, are accompanied by various challenges for implementing a chatbot. Each museum aiming to develop such a digital assistant requires the chatbot to cover knowledge around their particular field of content and to stay updated to their current exhibitions. This knowledge has to be provided by the museum itself. It is essential for the quality of a chatbot that this knowledge is not only organized in a standardized way but is also easily accessible. For this reason, we trust to increase the awareness of the importance of well-maintained databases within museums by demonstrating the potential of chatbots.
The diversity of people visiting cultural institutions leads to different topics of interests as well as manners of asking questions. Therefore, a chatbot has to cover a considerable band of questions that can be answered. By involving a broad range of people from different backgrounds in the process of generating training questions, we aim to cover a representative band of possible queries users might ask.
Furthermore, recording the conducted chatbot interactions allows us to extend the covered question set by the newly asked questions.

The ZKM - Center for Art and Media Karlsruhe - is a media museum in Karlsruhe. By connecting new, upcoming art with technology, the ZKM provides an international collection of interactive artworks that enable the visitor to experience installations [ZKM Center for Art and Media 2019]. As the ZKM acts as a pioneer in incorporating

modern, technology affine ideas, they offer the ideal surrounding for implementing and testing a chatbot. Therefore, this paper tackles the question of how to create a chatbot that guides the user in exploring objects and the ZKM itself.

In close cooperation with the ZKM we have developed a chatbot that is able to answer questions within a predefined subject area. It provides an ideal surrounding for us to evaluate the potential of chatbots in cultural institutions.

2 RELATED WORK

The rise of frameworks and platforms for creating chatbots, the improvement of statistical methods for implementing Natural Language Processing (NLP) as well as Dialogue Management and the availability of open APIs offer new opportunities in the application of chatbots.
Both larger and smaller museums now have the opportunity to explore the capabilities of chatbots in their environment with less effort at a low level of costs and resources [Bordoni et al. 2016; Schaffer et al. 2018].
In museums, there are various examples of AI and NLP enabled demonstrators like chatbots, covering different use cases with different intentions [Gaia et al. 2019].

In 2013, the Cooper-Hewitt Smithsonian Design Museum in New York launched their so-called Object Phone. Its initial intent was to enable visitors to receive more information about the objects they currently explore and experience in the museum. Three years later, in 2016, the museum expanded the Object Phone's capabilities by adding a subscription service for daily updates of the museum at home [Walter 2016].

In 2014, the Heinz Nixdorf Museum Forum in Paderborn published their avatar bot Max (**M**ultimodal **A**ssembly e**X**pert). Available via one screen inside the museum, it is possible to communicate face-to-face with the user. The focus of the project was to create a realistic human communication behaviour. The log files of all conversations were analyzed, and it was proven that interacting with embodied conversational agents (ECAs) encourages users to a more human-like communication form compared to interacting with textual or audio bots.
Besides, enforcing a human-like communication style - greeting, saying goodbye, asking for the weather - ECAs encourage people to have more small talk conversations [Kopp et al. 2005].

Intending to attract teenager audiences in a museum, in 2016 the House Museums Network in Milan started a digital project. The main challenge was that young teenagers are generally more interested in social media and online chats than in cultural contents. The solution was a Facebook Messenger based chatbot game. It enhanced the idea of a chatbot as a digital tour guide by offering a treasure hunt - a combination of gamification and storytelling - inside the museum.
Given a fictitious character, visitors had to find clues in the museum collection in order to solve a mystery. Since online chatting is one of the highest forms of social interaction amongst teenagers, the chatbot was an easily adaptable solution for them.
Furthermore, teenagers tend to test all conversational limits of a chatbot. The gamification approach shifted the central focus of the chatbot away from the human conversation towards the actual game. As a result, users became more forgiving when reaching conversational limits.
Moreover, buttons and multiple-choice questions minimized the likelihood of disruptions caused by misunderstandings.
The most challenging part when implementing the chatbot was to ensure that it understands a wide variety of different ways of saying the same [Boiano et al. 2018; Gaia et al. 2019].

In 2017, the San Francisco Museum of Modern Art (SFMOMA) further increased the capability of a museum at home by offering their SMS service SFMOMA Send Me. The SFMOMA Send Me chatbot is able to send the user a requested picture of the artwork. Since the collection of the SFMOMA has overgrown, with this service, interested people could view any piece of artwork any time remotely [Gaia et al. 2019; Mollica 2017].

In 2017, the Anne Frank Museum in Amsterdam initialized a chatbot use case to enforce increased user engagement. By using the Facebook Messenger, the museum enables an interactive, individual discovery of the museum. Via different storylines, the chatbot enables the user to discover the museum via various paths of Anne Frank's story [Gaia et al. 2019].

The museums named in the five studies cover a wide variety of different cultural focuses, artwork collections and ways of representing these. This diversity is representative of museums. Therefore, the questions museum visitors could ask a chatbot, as well as the answers a chatbot should give differ between cultural institutions. As the chatbot requirements are museum-specific, reusable training data for museum chatbots is only rarely available. Thus, every newly developed museum chatbot requires new training data and new data bases. For every cultural institution, own solutions for detecting intents and managing the dialogue have to be implemented [Schaffer et al. 2018]. For the implementation of a chatbot, it needs to be considered that for every museum a specialized, different, individually adjusted chatbot is required.

3 OWN APPROACH

3.1 MindMeld Framework

Before starting with the chatbot implementation, we had to choose an appropriate framework. For this study, we chose MindMeld, an open-source state-of-the-art conversational AI platform offering tools for all necessary components of a conversational workflow.
Besides, it is platform-independent, which was essential as we operated on different operating systems.
Moreover, it is advantageous that MindMeld is Python-based as it is a popular programming language offering many open source libraries related to the topic of natural language processing [Pedregosa et al. 2011].
Another reason for the decision was that it provides the possibility to host the data locally without third parties having access to it [Cisco Systems 2019a].

The components that are necessary to implement a chatbot are an Interface, a Natural Language Processor (NLP), a Dialogue Manager and a Knowledge Base. Thereby, the NLP consists of the Natural Language Understanding (NLU) and the Natural Language Generation (NLG) component. Figure 1 displays the components of the MindMeld framework.

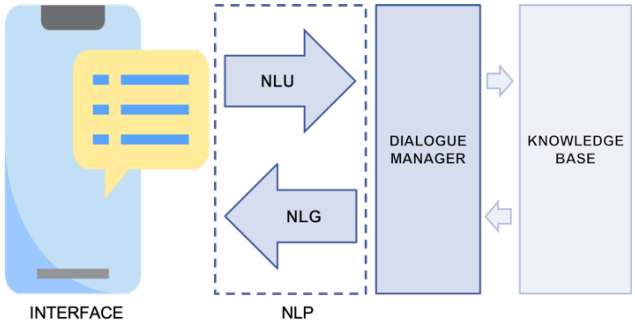

Figure 1: Components of the MindMeld framework (own representation based on [Cisco Systems 2019b]).

In terms of the human interface, MindMeld supports text as well as voice input [Cisco Systems 2019a]. However, voice input requires the integration of a third-party speech recognition system as MindMeld does not offer this function by default [Cisco Systems 2019e]. Due to time limitations, we focused on the text interface, as it decreases the dependency on the user's environment like background noise or sound speaker quality.

The NLU's purpose is to understand the user's input and translate the human language into a computer understandable form. In order to understand the human input, different information has to be extracted from each question. The required information consists of a domain, an intent, an entity and a role. The domain describes the overall knowledge area of the question, for example separating content-related questions from basic, general questions. The intent defines the purpose of the question. The entity enlarges the information value of the intent. It describes all further facts necessary to understand a question. The role is used to specify entities further. In order to extract this information, MindMeld provides a classification component for each stage.

Moreover, MindMeld offers a Language Parser component that clusters entities in meaningful hierarchies in order to point out relationships of words within a sentence. In the course of this study, the implementation of roles and therefore, also role classification was not necessary. Besides, we decided not to implement the Language Parser component as the given use case hardly implied the need to handle several entities within one sentence.

Figure 2 displays the structure of the implemented NLU as well as an example of our use case.

The NLG generates the human language answers that are given back to the user. Due to time limitations, this study focuses on implementing the NLG as a combination of hard-coded and dynamic answers generated by filling placeholders within the sentence.

The Knowledge Base aims to map the correct answers to the user

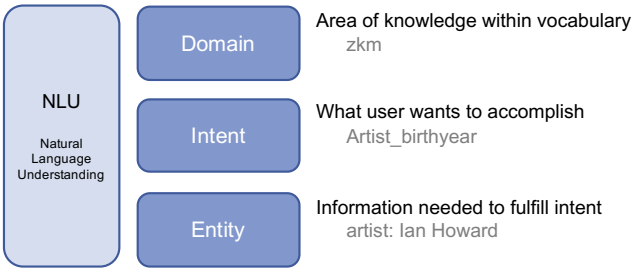

Figure 2: Structure of the implemented NLU (own representation based on [Cisco Systems 2019c]).

questions. It is the repository of all important information for the given use case and is accessed when answering the user questions. The Dialogue Manager guides the flow and structure of a conversation, such as memorizing the context of the dialogue throughout the conversation. It analyzes incoming queries and assigns them to different dialogue states that execute the appropriate logic [Cisco Systems 2019b].

3.2 Framework Application to the Use Case

After determining the framework, the scope of the chatbot had to be defined. In the case of this project, the knowledge areas that the chatbot is capable to answer were limited to the following four domains: basic, zkm, unknown and fun.

The domain basic covers necessary conversation flows to start and end conversations including the intents “greet” and “exit”. Since the testing phase revealed that potential users tend to start conversations not only greeting the chatbot but also asking for its well-being, the intent “how_are_you” was incorporated.

Another widespread non-topic-related activity of test users was asking the chatbot for jokes as soon as they ran out of specific questions. Jokes are an easy way to increase the fun throughout the chatbot interaction. Therefore, we included the intent “jokes” in the fun domain.

As it is not feasible to cover questions of all fields visitors could possibly ask, we implemented the “unsupported” intent in order to handle unknown questions.

The areas artist, exhibition, opening hours, journey and zkm background form the focus of ZKM related questions - the zkm domain. We chose the first two - artist and exhibition - because the ZKM provided access to their database in these fields. The intents the chatbot covers concerning the topics artist and exhibition are shown in Table 1.

The conversations related to artists and exhibitions mainly cover questions asked during the visit at the ZKM. To extend this use case with questions from outside the ZKM, the two topic areas opening hours and journey were incorporated. These are especially important when planning the museum visit.

Furthermore, to provide the chatbot with the possibility of answering questions about the ZKM's environment and setting we incorporated three intents within the topic zkm background.

For the intents not covered by the ZKM database, we generated a new database. Table 1 displays the scope and structure of the topics the chatbot is capable of replying.

			Entity Type		
Domain	Topic	Intents	Artist	Exhibition	Time
cultural institution - zkm	artist	artist_age	X		
		artist_birthplace	X		
		artist_birthyear	X		
		artist_information	X		
	exhibition	exhibition_artist		X	
		exhibition_current		X	
		exhibition_description		X	
		exhibition_duration		X	
	opening hours	openinghours			X
	zkm background	zkm_general			
		zkm_cinema			
		zkm_content			
	journey	zkm_journey_bus			
		zkm_journey_car			
unkown		unsupported			
basic		greet			
		exit			
		how_are_you			
fun		jokes			

Table 1: Overview of implemented domains, intents and entity types.

3.3 Machine Learning Approach

The chatbot was trained using machine learning algorithms. As shown in Figure 3, MindMeld consists of a multistage classification model comprising of a domain classification, an intent classification, an entity recognition and a role classification model.

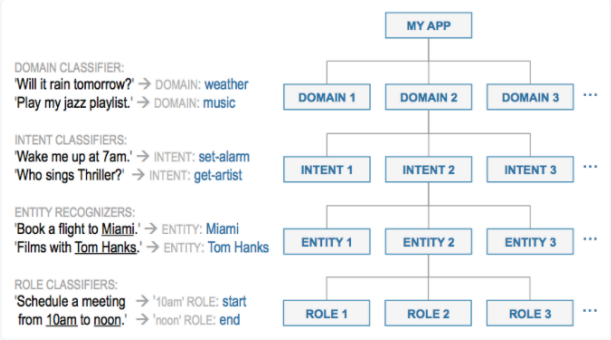

Figure 3: MindMeld multistage classification model [Cisco Systems 2019c].

For the given use case, the first three stages were sufficient, and therefore, we did not consider a role classifier for the implementation.

In the first stage, we use a domain classifier in order to identify the domain the user addresses with his request. Different domains usually differ considering their specific vocabulary. While the basic and fun domains use elementary vocabulary, the zkm domain uses more specific vocabulary related to artists or exhibitions.

In the second stage, the chatbot uses intent classification in order to identify the concrete intention of a user request. For domain and intent classification, MindMeld offers the following four classification models: Support Vector Machine (SVM), Decision Tree, Random Forest and Logistic Regression.

3.3.1 Support Vector Machine. In the binary case, this classification model aims to find the hyperplane that maximizes the margin between two classes [Kotsiantis 2007]. For multi-class problems like domain or intent classification, several binary classifiers are combined. Two conventional approaches are "one-versus-rest" or "one-versus-one". In the first, for each class, a binary classifier is constructed maximizing the distance to all remaining classes. The latter constructs pairwise binary classifiers for all possible pairs of classes [Hsu and Lin 2002].

3.3.2 Decision Tree. The idea of Decision Trees is to break up a complex decision into a union of several simpler decisions. Every node in a Decision Tree represents an attribute test. At the leaf nodes, classification into a particular class is made. In order to choose optimal attributes at each hierarchy level, the entropy, which is a homogeneity measure of the training data, is calculated for every possible attribute. The attribute that maximises the homogeneity of the training data after each attribute-based decision is selected for splitting the training data in the next step [Safavian and Landgrebe 1991].

3.3.3 Random Forest. A Random Forest is an ensemble learning method consisting of several Decision Trees. Each Decision Tree of the forest has one vote. A data point is classified into the class with the most votes by the Decision Trees of the Random Forest. Each Decision Tree draws random samples of attributes and training data. This means that the Decision Trees are trained based on a different data basis. Faster training is possible since each Decision

Tree is smaller than the original one. As the number of trees in the forest becomes large, the generalization error for Random Forests converges [Breiman 2001].

3.3.4 Logistic Regression. This method is a modification of a Linear Regression to a classification task. The idea behind this is to model the probabilities of a specific class. It can be done both for a two-class and a multi-class classification problem. One way to do this is to use the logistic sigmoid function to squash the output of the linear function into the interval (0, 1). In order to find the optimal weights, we can minimize the negative log-likelihood by making use of gradient descent [Goodfellow et al. 2016].

After domain and intent are classified in the first two stages, the entity recognizer initially recognizes the sequence of the sentence containing the entity. For this stage, MindMeld offers three further classification models: Maximum Entropy Markov Model (MEMM), Conditional Random Field (CRF) and Long-Short Term Memory Neural Network Model (LSTM).

3.3.5 Maximum Entropy Markov Model. This approach combines a Maximum Entropy Classifier with a Hidden Markov Model. The model builds on the assumption that explanatory variables are connected in a Markov Chain. Based on observation sequences, the model thereby defines the conditional probability of state sequences. Maximum entropy is used to fit the probability representation of a state given a previous state and an observation [McCallum et al. 2000].

3.3.6 Conditional Random Field. It is a probabilistic framework used for labelling and segmenting sequential data and belongs to the class of discriminative undirected probabilistic graphical model. Conditional Random Fields build on a conditional distribution P(Y|x) with an associated graphical structure. Due to their conditional nature, dependencies among the input variables x do not need to be explicitly represented [Sutton et al. 2012]. Furthermore, they can be trained by making use of an exponential loss function [Lafferty et al. 2001].

3.3.7 Long-Short Term Memory Neural Network Model. It is a special form of a Recurrent Neural Network (RNN) designed to operate on sequential data with a focus on learning long-term dependencies. Using Truncated Backpropagation Through Time, LSTMs tackle the problem of vanishing or exploding gradients in Recurrent Neural Networks. By incorporating a memory cell, LSTM networks can capture long-range dependencies of the training data. LSTMs use multiple gates to address the vanishing gradient problem. While the input gate decides when to write to the LSTM neuron, the forget gate controls whether saved states are still relevant. Lastly, the output gate learns the importance of the cell output at a particular time step. Due to the gating mechanism, LSTM cells have more parameters than a hidden neuron of on ordinary RNN [Lample et al. 2016].

Finally, the entity resolver maps the identified entity to an identification value. This allows the information requested by the user to be obtained from the database [Cisco Systems 2019b].

4 IMPLEMENTATION

4.1 Training Data

A problem of most current chatbot implementations is their narrowed conversational skills [Hill et al. 2015]. These skills, primarily, the capability of providing the correct answer to a particular intent, depend inter alia on the size of the training data. In the case of museum chatbots, due to the different knowledge fields of each cultural institution, training data is not transferable from other museums and thus needs to be generated. Therefore, widening, refreshing and specializing the data basis is a mandatory part of implementing a chatbot [Boiano et al. 2003; Schaffer et al. 2018].

Training data in the context of this project includes possible visitor questions the chatbot should be capable of handling. This data is required in order to train the classifiers.
The ZKM provided a list of questions covering possible intentions a visitor could have when asking a chatbot questions. However, for each intent the variety of manners and formulations for asking questions was limited in the provided data set.
As an initial step to widen the different styles people ask for a certain information, we developed a questionnaire which was completed by 10 people. For each intent covered by the chatbot, the questionnaire asked the participant to think of possible different ways and formulations to ask for that particular intent.
To further enlarge the variety of questions, we manually extended the number of possible formulations for every intent to 150. Using data augmentation by integrating spelling mistakes through character swapping and deleting, we tripled the number of questions per intent. It turned out that this technique has to be applied carefully. Applying this approach to all words of a sentence led to a deterioration of the performance, whereas limiting it to filler words and excluding questions consisting of single words led to overall performance improvement.
Analyzing the visitor's dialogues with the chatbot provides more in-depth insights in the fields users are interested in and the manner in which they ask questions. In order to observe the real activities and conversations between visitors and the chatbot, we implemented a function to generate log files. It provides the opportunity to document the conversation with the chatbot and make weak performances visible. Thus, analyzing the log files offers the possibility to improve the chatbot's performance by incorporating new formulations in the training data and expanding the knowledge base by additional subjects users are particularly interested in [Boiano et al. 2003].
In order to expand the training data by new questions, they have to be labelled. To implement this, we made use of a semi-online learning approach to increase the information value of the log files. Therefore, the user has the possibility to enable a support mode. In case this mode is activated during the chatbot interaction, whenever a false classification occurs, the user is asked to label his question with the correct intent. These labels are added to the log file. Subsequently, we decided to review the log files manually in order to improve the quality by excluding wrong classifications before incorporating them in the data basis.

4.2 ZKM Dataset

The Content Management System of the ZKM builds the basis of the databases that provide information for the chatbot. These databases contain information about artists, works, events, publications and assets (audio & video). In this present work, only the SQL tables “zkm_export_person” and “zkm_export_exhibtion” are used. The usage of both tables is illustrated by Table 2. The database “zkm_export_person” contains all the information about the artists, whereas the other encompasses exhibitions at the ZKM.
To describe exhibitions, 51 variables are available. Inter alia, they include an ID, a title, a start and end date, a short description and links to artists that participated.
The database about artists contains 42 variables. The most important ones represent the artist's ID, name, year of birth and death, nationality and description of his life.

Before converting the SQL tables into a MindMeld readable format, we removed several regular expressions. For instance, the description variable often contained HTML-tags, such as ... or <td> ... </td>.
Furthermore, we removed references to the Content Management System that appeared in some observations of the SQL data table. A case in point is “Portrait - Joachim Tesch (Drupal_AID:1094)”. Here, the content inside the brackets was unnecessary.
After these adjustments, we transformed the SQL tables into a JSON file following a specific syntactically format required by MindMeld.

4.3 Additional Represented Knowledge

In addition to the knowledge bases created on the ZKM database, a knowledge base was created for the domain fun and the intent opening hours, respectively. The first consists of a collection of jokes, each of which is assigned an ID. The latter comprises a list of all weekdays as well as the opening time frame on a particular day. Besides, the knowledge base contains a list of the holidays on which the ZKM is not open for visitors. An ID was assigned to each of the weekdays and holidays.

4.4 Entity Mapping

To make the information about entities stored in the knowledge base accessible, an entity mapping JSON file creates a connection to the knowledge base. For each knowledge base, namely for artists, exhibition and opening hours, such a file exists in the JSON format. These files assign an ID to each entity, which can be used to query all the knowledge stored in the knowledge base.
For the knowledge base jokes, the creation of an entity mapping file was not necessary as the domain fun does not require the introduction of entities due to its simple structure.

Another useful feature of the entity mapping files is the so-called whitelist. This list contains synonyms of a defined entity, which helps to recognize the entity. In this work, the entity mapping file for artists contains the surnames. However, 2254 entries of the knowledge base contain the same surname. In these cases, no unique identification is possible, and therefore, their surnames were left out in the whitelist.

Adding synonyms to the whitelist of the exhibition entity mapping file was accompanied by the difficulty that the titles of the exhibitions do not follow a consistent structure. However, it turned out that various titles follow a particular form: “artist name: short title. longer description”. In these cases, the artist name followed by a colon was removed to avoid an entity of the type exhibition being misclassified by the classifier with an entity of the type artist. Because of its length, we excluded the longer description from the whitelist, while including the short title.
In the entity mapping files for opening hours, we added the adverbial version of each weekday to the whitelist.

4.5 Correction of Entity Misspellings

In order to create a suggestion system accounting for user misspellings, we calculated the similarity of the user's input with the other questions using the Levenshtein distance. Informally speaking, this distance equals the minimum number of character changes to transform one word into another [Levenshtein 1966]. The entry with the smallest distance is the one with the highest similarity and therefore chosen as a suggestion. The Levenshtein distance belongs to the broader topic of approximate string matching [Apostolico and Galil 1997].
However, this feature is only available for sentences that contain exhibitions as an entity. Entities of the entity type exhibition are most prone to spelling mistakes as they are long and inconsistent in their syntax. Therefore, as the scope of this work was limited, the suggestion system is only implemented for sentences containing this entity type.

4.6 Chatbot Interface

The interface enables the user to interact with the chatbot, and thus, it is vital to make the user experience as conveniently and intuitively.
In this work, we chose to build a web app. (see Figure 4). One advantage of this implementation is the independency from operating systems. In other words, the web application is compatible with any device. The framework for the web app is Flask, a Python-based web framework. In that way, it was easily possible to start the web app from a python script after having trained the different classifiers of the chatbot. The web app looks like a common chat messenger with the intention that the user can directly familiarise with the chat environment. The app mediates between the user and the chatbot. Every time the user enters a message, the request is sent to the chatbot and the answer is anticipated.

We used Java Script to describe the processing logic, which allows adjusting the interface based on the user action dynamically. To simulate a human-like behaviour of the chatbot, a typing GIF appears before the user gets the answer of the chatbot.
Furthermore, the interface integrates tables in the answers of the chatbot (see Figure 8 in Appendix A). For instance, if the user wants to know more about the current exhibitions at the ZKM, the chatbot looks up the available exhibitions in the database and sends them to the web app. Subsequently, the received data is transformed into a table via JavaScript, HTML and CSS and then integrated into the answer.

Table name	Field	Number of entries	Missing values [%]	Number of fields	Number of used fields
zkm_export_exhibition	id	677	0	51	6
	titel_de	677	0		
	start_tag	677	0		
	ende_tag	677	0		
	kurzbeschreibung_seo_de	476	30		
	person_freie_rolle	576	15		
zkm_export_person	id	20584	0	42	7
	vorname	18827	9		
	nachname_teamname	20584	0		
	born_year	4340	79		
	died_year	841	96		
	nationality_de	468	98		
	beschreibung_de	2310	89		

Table 2: Overview SQL data bases

A "home button" was implemented to enable the user to return to the start screen of the chatbot displaying the covered topics. Lastly, a toggle button enables the support mode to be activated (see Figure 4).

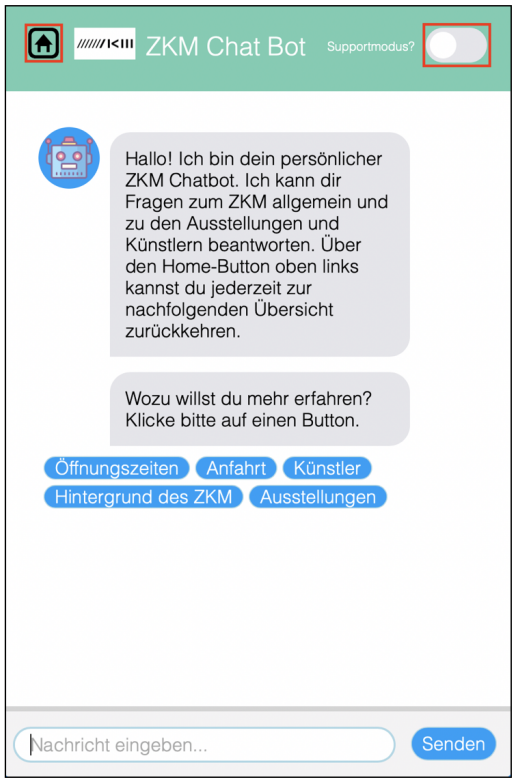

Figure 4: Interface of the chatbot with an implemented home button and toggle button for the support mode.

4.7 Natural Language Processing Component

Concerning the Natural Language Processing component, the implementation of the machine learning classification model was essential. For this purpose, we implemented the multistage classification model including domain classification, intent classification as well as entity recognition and resolver. In order to optimize the classification performance, two main aspects - a classifier pipeline and a semi-online learning approach - were considered. For each stage, several algorithms were trained and compared in a pipeline (see Figure 5).
In terms of preprocessing, n-grams with n=[1,2] and bag-of-words with bags in the interval [1,2] were implemented. For all three stages, the choice of classifiers was determined by the algorithms offered by MindMeld.

The domain and intent classification pipelines both comprise a Support Vector Machine (SVM), a Random Forest Classification Model and a Logistic Regression, respectively.
Besides the three considered models, MindMeld offers a Decision Tree classifier. As Random Forests are composed of several Decision Trees, each trained with random samples, they generally outperform Decision Trees given large data sets [Ali et al. 2012]. In order to reduce the time needed to train the models, we omitted the Decision Tree in the pipeline.

The pipeline for entity recognition consists of a Maximum Entropy Markov Model (MEMM) and a Conditional Random Field (CRF). In addition to the models mentioned, for entity recognition, MindMeld offers a Long-Short Term Memory Neural Network Model (LSTM). However, in the documentation, it was recommended only to incorporate the model if the amount of train data is 1000 or higher. As the maximum training data amount available during this study was 450 questions per intent, we excluded this model from the pipeline as well.

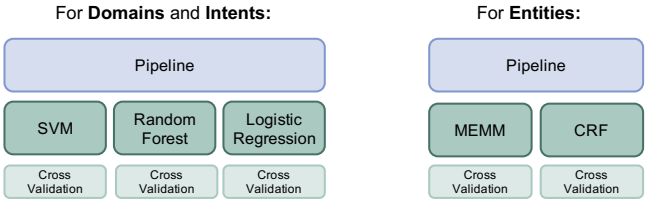

Figure 5: Training pipeline for the chatbot (own representation).

To choose the best classifier model for each stage as well as evaluate the overall performance of the chatbot, we chose the accuracy metric [Sokolova and Lapalme 2009]. For this choice, a balanced data set is advantageous. According to this metric and using 10-fold cross-validation, the best model was chosen for later use in the operating chatbot.

Moreover, a semi-online learning approach helps to augment the training data. The thereby generated training data has to be added manually to the training files of the chatbot classifiers. Since automatically labelling the new data is prone to errors, supervision of a human is required.
Besides, in this way, we are able to control which newly generated data is used for further training data improvements.

Clicking on a toggle button activates the support mode. In this mode, the chatbot asks the user to validate its output. Afterwards, the chatbot logs the results and differentiates between correctly and incorrectly answered queries.
If the answer of the chatbot does not fit to the user request, the user is asked to select the correct intent for the sent message. In this case, the user can click on the button with the right intent. This case is shown in Figure 6. Having selected the correct intent, the user message and the selected intent are added to the logfile. The information gain through user feedback can be used to improve the training data afterwards.

The support mode is an efficient solution to improve the chatbot. While using the support mode, new training data can be generated easily. For each defined intent, two files are created: one for the already correctly modelled questions and another for the incorrectly answered questions.

4.8 Dialogue Manager

The Dialogue Manager provided by MindMeld enables the chatbot to not only respond to an incoming request but also to have a realistic dialogue with the user corresponding to the human language. It uses pattern-based rules to derive the dialogue state of a request sent to the chatbot. This means that the chatbot can remember the context of a dialogue throughout multiple dialogue states. As a result, it can return a natural language response matching the context to the user [Cisco Systems 2019d].

Prior to implementing a dialogue manager, possible dialogues within the existing use case have to be conceptualized. Here, we decided

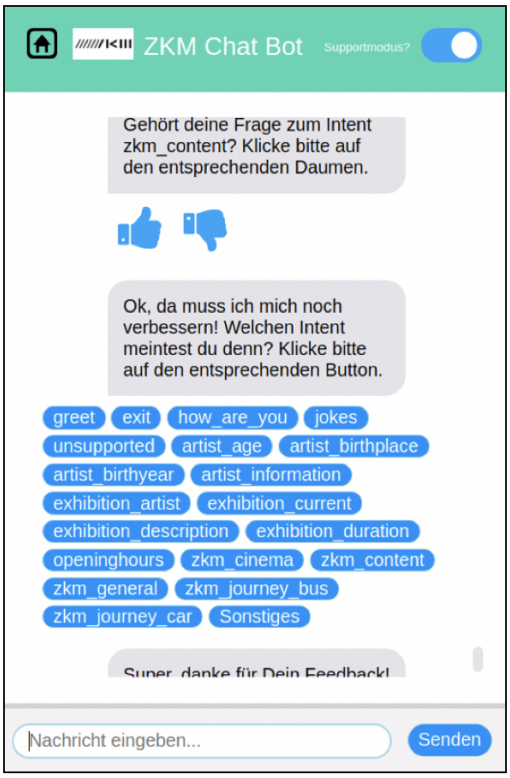

Figure 6: Interface of the chatbot with support mode enabled asking the user to label the correct intent.

to focus on using the dialogue manager for context memory as well as user guidance.
Within the area of context memory, a dialogue functionality for the topic artists as well as opening hours was realized. Once the user asks the chatbot a question regarding an artist, it can answer further questions about this particular artist without explicitly mentioning his/her name again. The chatbot can continue this dialogue flow until a question not related to the current artist is raised.
Analogously, we implemented this functionality for opening hours. Once the chatbot has realized that a user requests information about the ZKM's opening hours, it will ask for which day a user wants to know exactly the opening hours. Now, the user can only answer with the particular day without setting it to the context of opening hours again. To enable this functionality, the chatbot stays in this dialogue state until an entity of the type time is received. In contrast to the dialogue flow implementation of artists, the chatbot will exit the opening hour state once the answer for a particular day was provided and does not remain in this state until another topic is raised. The reason for this is that during testing, users tended to ask several questions in a row about an artist. However, they only requested the opening hours of a specific day.

User guidance aims to prevent a user from asking questions to the chatbot that are outside its area of competence. We realized this by introducing the topics covered by the chatbot at the start

screen of the conversation combined with the following question and respective buttons for topic selection: “Wozu willst du mehr erfahren? Klicke bitte auf einen Button”. A user can return to this initial state by clicking on a home button located in the upper left corner (see Figure: 4).

4.9 Deployment and Hosting

To deploy the chatbot independently of the operation system, we decided to use Docker. Docker is open-source software that isolates applications using containers [Docker Inc. 2019a]. Such a container contains all the packages necessary for the application.
Additionally, Continuous Integration/Continuous Deployment is also a key factor. The administrator of the online tool benefits from faster and more efficient code changes in the system [Docker Inc. 2019b].
In this case, two Docker container communicate with each other using a predefined interface. This interface is defined in the YAML file. The first Docker service contains MindMeld with all its dependencies, and the second one represents the web application. While MindMeld's Docker container already exists on Docker Hub, the Docker container of the web application needs to be built. All necessary steps of bringing it up are defined in the so-called Dockerfile.

As Docker cannot handle multiple requests well, we used NGINX as a web server. NGINX is open-source software for web serving with a focus on performance and stability [NGINX Inc. 2019].

The server we hosted the chatbot on is part of bwCloud. bwCloud is a free infrastructure as a service (IaaS) of the federal state Baden-Württemberg for tertiary education [bwCloud 2019]. In particular, we used an Ubuntu server.

5 EVALUATION

5.1 Testing the Chatbot with ZKM Visitors

To evaluate our chatbot, two evaluation rounds were carried out. The first one involved asking people onsite at the ZKM to use our chatbot. Based on these results of the first testing round, we conducted adjustments, which we consequently verified in the second evaluation round.

As part of the first iteration, 50 questions were asked, out of which 43 were about topics our chatbot covered. 32 of these 43 questions were classified correctly, yielding an accuracy rate of 74%.
Learnings from the first iteration were not only about the answers of the chatbot itself, but also about the user interface. Especially, the position of the home button seemed to be inconvenient for the users.
Furthermore, the user interface was not interactive enough. For instance, when the chatbot replied with a table containing the artists of an exhibition, there was a strong desire for the user to click upon the artist to obtain further information.
In terms of age, there was also a tendency of older users to use only keywords as questions, while younger people showed a greater enthusiasm to formulate whole sentences. Secondary functions, such as telling the user a joke, sparked great joy since it made the chatbot more human-like.

To implement the learnings in the chatbot, several measures were taken. Inter alia, the home button was shifted (see Figure 9 in Appendix A), new questions were added as training data and the data was augmented artificially by character swapping and deletion. However, the last step was only done with semantically unimportant words, such as articles and prepositions.

In the second round testing, 150 questions were asked. This time, the accuracy rate increased to 77% with 116 questions correctly classified. Thus, the undertaken measures improved the overall performance. The accuracy of both iterations rounds are summarized below (see Figure 7).

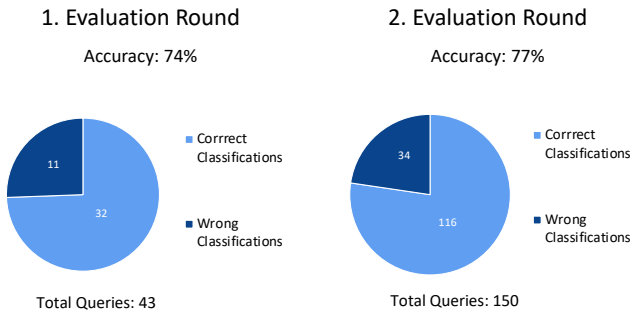

Figure 7: Performance of the chatbot after evaluation round 1 and 2 (own representation).

As illustrated in Figure 7, the adjustments after the first testing round lead to an improvement of the accuracy by 3%. Hereby, the extension of the amount of training data from 150 to 450 questions per intent was decisive for the performance improvement.

Comparing chatbots with each other is relatively tricky because each of them has a distinct area of expertise and depends on the query language itself. For instance, children will use a different language when communicating with a chatbot than adults. As this chatbot is built and tested for and with adults, the results will most likely be inferior when testing with children. Hence, the smallest comparison base is if the output of the chatbot satisfies the query of the user.
In 2014, [Coniam 2014] followed precisely this assumption and compared several chatbots. The author introduced two evaluation criteria, namely the grammatical and meaning fit. On average, there was a high accuracy of 87.9% in terms of grammar and a significantly lower one of 65.3% in meaning. Overall, when looking for a meaning and grammar fit, only 59.4% of all 163 queries were answered correctly.
Comparing the chatbot of the ZKM with these results might suggest a superior performance, however, as indicated previously, comparing chatbot results is difficult as the results strongly depend on the test users, their fields of interest and their way of asking a question. Moreover, the named study comparing several chatbots was published five years ago. Since then technology and especially chatbot frameworks have developed significantly.

5.2 Discussion and Limitations

As described in section 3.3, MindMeld offers a four-step classification process. However, these four steps could be insufficient and too inflexible for larger projects as they limit the manageable complexity.

Moreover, in the chatbot implementation of the current work, the Dialogue Manager can only stay in a dialogue flow until a certain entity is supplied. For instance, when asking for the opening hours, the user is asked to choose a day of the week. Once it is chosen, the dialogue flow is exited, and then the user cannot ask for another weekday unless he/she repeats his first question. Figure 10 in Appendix A illustrates this case.

Furthermore, MindMeld is relatively new. As a consequence, the only source for problem-solving is the documentation. However, the documentation could not cover all the problems.

In terms of establishing a human-like conversation, the answers generated by the chatbot need to sound as natural as possible. MindMeld, however, uses a rule-based Dialogue Manager, which limits the flexibility of the chatbot's answers. A non-rule-based Dialogue Management approach might be more applicable in this context.

Nevertheless, MindMeld is only partly to blame since the supplied databases from the ZKM in many cases consists of language that would not be used in informal conversations. Mainly, this is the case for the variables that describe the exhibition or the artist's life. For instance, the sentence "Ein Leitprojekt des Wissenschaftsjahres im ZKM." would not be used in an informal conversation. Another shortcoming is the high percentage of missing data entries (see Table 2). These two shortcomings could not be resolved as part of this work and can be used as starting points for further improvements.

In terms of speed, some queries need long processing times, especially when the query involves computation of the manually implemented Levenshtein distance. The computational time to calculate the similarity of two strings of length n is:

$$O(n^{2-\epsilon}) \tag{1}$$

ϵ denotes any value greater than zero. Consequently, it is a slow but accuracy increasing feature because it corrects misspellings of entities [Backurs and Indyk 2015].

Furthermore, the user interface still has potential for improvement. As already pointed out, users want to be able to receive information without typing a complete question quickly. Therefore, they would prefer the implementation of more typing effort reducing features like buttons.

Finally, we recommend improving the interface for mobile devices by adapting the size of the chat to the mobile phone screens.

6 CONCLUSION AND OUTLOOK

In the scope of this project, we implemented a chatbot providing the ZKM visitor with information about the topics exhibitions, artists, opening hours, ZKM background, the journey to the museum, greeting, farewell and jokes.

The chatbot guides the user through these covered topics by providing an all-time accessible topic overview. Thereby, for each topic, a more detailed description of the chatbot's capabilities is accessible by clicking the respective button in the overview.

Furthermore, the bot can handle context-dependent aspects within dialogues flows related to the topics opening hours and artists.

In order to optimize the performance of the classification model, we implemented a classification pipeline for the domain, intent and entity level respectively, within which we selected the best classification model for the deployed chatbot using accuracy and 10-fold cross-validation.

As the amount of train data is a crucial issue, we implemented a semi-online-learning approach to continuously expand labelled training data by incorporation of the chatbot user.

As shown in section 5, the extension of the training data from 150 to 450 already resulted in an accuracy improvement of 3%.

A further significant increase in the number of training data, therefore, offers clear potential for further performance improvements.

Future work on the chatbot could include to further enhance its answering quality by examining the syntax of the answers given by the chatbot. Implementing a more realistic, human-like wording could increase the acceptance by the users and should be considered in future development.

Besides, extending the text input with an additional voice input could contribute to an increase in the acceptance of the chatbot by the user as well. A third party Automatic Speech Recognition System can be used to convert the input speech into text which can subsequently be processed by MindMeld. Finally, a third party Text to Speech System is required to synthesize the audio response in order to return a spoken answer to the user [Cisco Systems 2019e].

Furthermore, conversation quality is highly dependent on the quality of the knowledge base. Currently, the ZKM database contains many empty fields. Consequently, in this case, the chatbot provides the user with the answer of not having any question-related information available. Thus, improving data quality offers further room for improvements.

Additionally, the chatbot could be used site-specifically within exhibitions of the ZKM. If a visitor stands directly in front of an exhibition object, his or her location could be used to directly provide information about the exhibition object through interacting with the chatbot.

Lastly, the deployed version of the chatbot runs in its isolated environment and has to be integrated into the ZKM's IT infrastructure.

Taking these ideas for further development into consideration, the current version of the chatbot, which will be handed over to the ZKM, offers a suitable starting point for the realization of a chatbot that can be launched at the ZKM.

REFERENCES

Jehad Ali, Rehanullah Khan, Nasir Ahmad, and Imran Maqsood. 2012. Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)* 9, 5 (2012), 272.

Alberto Apostolico and Z. Galil. 1997. *Pattern matching algorithms.* Oxford University Press, New York and Oxford.

Arturs Backurs and Piotr Indyk. 2015. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing.* ACM, 51–58.

Stefania Boiano, Ann Borda, Guiliano Gaia, Stefania Rossi, and Pietro Cuomo. 2018. Chatbots and New Audience Opportunities for Museums and Heritage Organisations. In *Proceedings of the Conference on Electronic Visualisation and the Arts (EVA '18).* BCS Learning & Development Ltd., Swindon, UK, 164–171. https://doi.org/10.14236/ewic/EVA2018.33

Stefania Boiano, Giuliano Gaia, and Morgana Caldarini. 2003. Make Your Museum Talk: Natural Language Interfaces for Cultural Institutions. (2003).
Luciana Bordoni, Francesco Mele, and Antonio Sorgente. 2016. *Artificial intelligence for cultural heritage.* Cambridge Scholars Publishing.
Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (01 Oct 2001), 5–32. https://doi.org/10.1023/A:1010933404324
bwCloud. 2019. bwCloud IaaS fÄijr Forschung und Lehre in Baden-WÃijrttemberg. (2019). Retrieved 2019-07-28 from https://www.bw-cloud.org/en/bwCloud_old
Cisco Systems. 2019a. Conversational AI platform for deep-domain voice interfaces and chatbots. (2019). Retrieved 2019-07-26 from https://www.mindmeld.com
Cisco Systems. 2019b. Conversational AI platform for deep-domain voice interfaces and chatbots. (2019). Retrieved 2019-07-26 from https://www.mindmeld.com/docs/userguide/architecture.html
Cisco Systems. 2019c. Conversational AI platform for deep-domain voice interfaces and chatbots. (2019). Retrieved 2019-07-26 from https://www.mindmeld.com/docs/quickstart/03_define_the_hierarchy.html
Cisco Systems. 2019d. Conversational AI platform for deep-domain voice interfaces and chatbots. (2019). Retrieved 2019-07-26 from https://www.mindmeld.com/docs/userguide/dm.html
Cisco Systems. 2019e. Dealing with Voice Inputs. (2019). Retrieved 2019-08-29 from https://www.mindmeld.com/docs/userguide/voice.html
David Coniam. 2014. The linguistic accuracy of chatbots: usability from an ESL perspective. *Text & Talk* 34, 5 (2014), 545–567.
Docker Inc. 2019a. What is a Container. (2019). Retrieved 2019-07-28 from https://www.docker.com/resources/what-container
Docker Inc. 2019b. What is a Container. (2019). Retrieved 2019-07-28 from https://www.docker.com/solutions/cicd
Giuliano Gaia, Stefania Boiano, and Ann Borda. 2019. Engaging Museum Visitors with AI: The Case of Chatbots. In *Museums and Digital Culture.* Springer, 309–329.
Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning.* MIT press.
Jennifer Hill, W Randolph Ford, and Ingrid G Farreras. 2015. Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations. *Computers in Human Behavior* 49 (2015), 245–250.
Chih-Wei Hsu and Chih-Jen Lin. 2002. A Comparison of Methods for Multiclass Support Vector Machines. *Trans. Neur. Netw.* 13, 2 (March 2002), 415–425. https://doi.org/10.1109/72.991427
Stefan Kopp, Lars Gesellensetter, Nicole C Krämer, and Ipke Wachsmuth. 2005. A conversational agent as museum guide–design and evaluation of a real-world application. In *International Workshop on Intelligent Virtual Agents.* Springer, 329–343.
S. B. Kotsiantis. 2007. Supervised Machine Learning: A Review of Classification Techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies.* IOS Press, Amsterdam, The Netherlands, The Netherlands, 3–24. http://dl.acm.org/citation.cfm?id=1566770.1566773
John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).
Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. *CoRR* abs/1603.01360 (2016). arXiv:1603.01360 http://arxiv.org/abs/1603.01360
Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, Vol. 10. 707–710.
Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 591–598. http://dl.acm.org/citation.cfm?id=645529.658277
Jay Mollica. 2017. Send Me SFMOMA. (2017). Retrieved 2019-07-26 from https://www.sfmoma.org/read/send-me-sfmoma/
NGINX Inc. 2019. What is NGINX? (2019). Retrieved 2019-07-28 from https://www.nginx.com/resources/glossary/nginx/
Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.
S. R. Safavian and D. Landgrebe. 1991. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics* 21, 3 (May 1991), 660–674. https://doi.org/10.1109/21.97458
Stefan Schaffer, Oliver Gustke, Julia Oldemeier, and Norbert Reithinger. 2018. Towards chatbots in the museum. (2018). http://ceur-ws.org/Vol-2176/paper5.pdf.
Marina Sokolova and Guy Lapalme. 2009. A systematic analysis of performance measures for classification tasks. *Information processing & management* 45, 4 (2009), 427–437.
Charles Sutton, Andrew McCallum, et al. 2012. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning* 4, 4 (2012), 267–373.
Micah Walter. 2016. Object Phone: The continued evolution of a little chatbot. (2016). Retrieved 2019-07-26 from https://labs.cooperhewitt.org/2016/object-phone-the-continued-evolution-of-a-little-chatbot/
ZKM Center for Art and Media. 2019. The ZKM. (2019). Retrieved 2019-08-25 from https://zkm.de/en/the-zkm

A APPENDIX

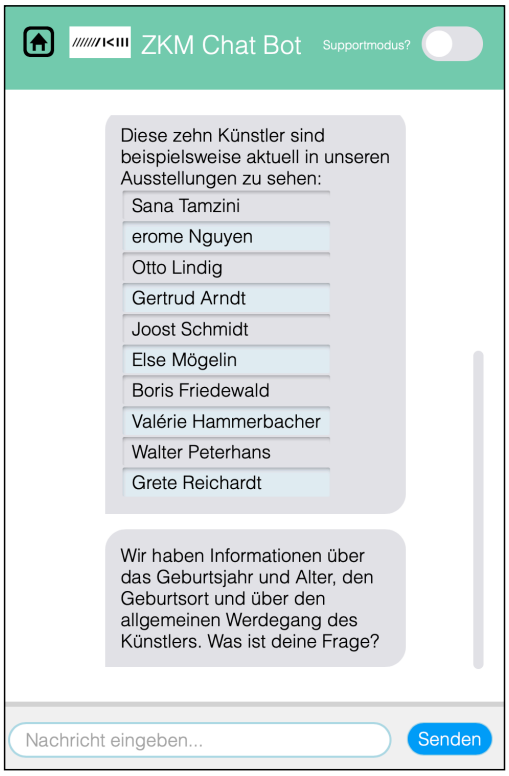

Figure 8: Chatbot's answers displayed in table format.

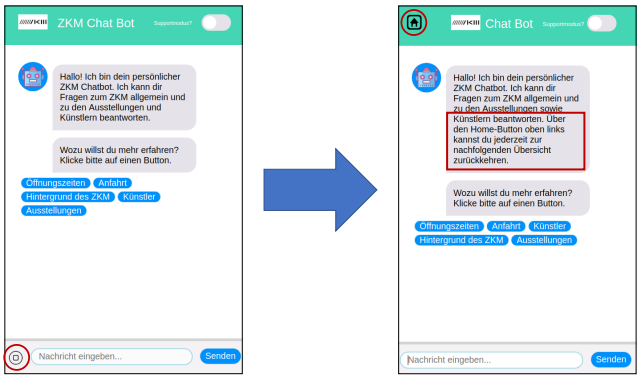

Figure 9: Development of the home button location in the chatbot interface.

Figure 10: Shortcomings in the dialog flow opening hours.