User-adaptive Statistical Dialogue Management using OpenDial

Master Thesis

by

Nicolas Wagner

Reviewer:	Prof. Dr. Dr.-Ing. Wolfgang Minker
Co-Reviewer:	Prof. Dr.-Ing. Michael Weber
Supervisor:	M.Sc. Juliana Miehle

Institute of Communications Engineering
University of Ulm
01 December 2017

Version October 2018

Abstract

In Spoken Dialogue Systems, two techniques are currently used to create an optimal dialogue policy: hand-crafted rules and statistical procedures basing on machine learning. However, both types are not sufficient in complex areas where only limited training data is available. This thesis thus examines a hybrid approach to dialogue management that intents to combine the benefits of both rule-based and statistical methods. For this purpose, probabilistic rules are employed which depend on unknown parameters. Afterwards, these parameters are trained with supervised learning. Furthermore, the dialogue manager is designed to be adaptive to the user's cultural background and emotional condition as this is supposed to have a crucial influence on the conversational behaviour. The configuration is then investigated in the context of the KRISTINA domain. The conducted experiments reveal that it is possible to include emotional and cultural features in the dialogue management.

I certify that I have prepared this Master Thesis by my own without any inadmissible outside help.

Ulm, 01 December 2017

(Nicolas Wagner)

Contents

List of Figures

List of Tables

1. Introduction

> An interface is humane if it is responsive to human needs and considerate of human frailties.
>
> *Jef Raskin*

The above quotation from the well-known computer scientist Jef Raskin specifies requirements for a humane human-machine interface. In general, it is thus necessary to identify and understand both the needs and weaknesses of users. Although Raskin proposed his opinion on natural user interaction in 2001, present systems still do not meet these expectations yet. In contrast, most widely-used systems rely on input and output options such as screens and keyboards that feel strenuous to operate and are limited in their applicability. Some may even cause health problems as reported in [10, 32]. Hence, the transition to alternative channels seems inevitable.

As humans mainly communicate with each other through speech, these cognitive skills are already developed in childhood and trained since then. Due to this fact, it is a reasonable intention to use speech also for human-machine interaction. The ultimate design aim of a Spoken Dialogue System is to make it convenient for users to work with the system and reduce the occurrence of errors. Since speech is the most intuitive way of communication, this type of interface enables even non-experts to access complex technology. Moreover, a user is able to accomplish another task simultaneously while using a Spoken Dialogue System. The potential of this technology has been recognised particularly in recent years and research has been intensified accordingly. As a result, it is already possible presently to interact with diverse computer applications through natural spoken language. Nevertheless, public opinion on these systems is rather low so far.

According to [30], the performance of a dialogue system depends heavily on the behaviour and condition of the user. Since current approaches used in these systems are often not user-adaptive, this leads commonly to an insufficient user experience. Another reason that is responsible for the lack of acceptance is the static and inflexible dialogue flow created by predefined rules. As stated in [51], a significant problem here is not including real dialogue data to improve the dialogue management. As assumed in [17], the employment of statistical methods can help to overcome this limitation. However, these techniques require a large amount of dialogue data which are not available in most domains. The concept of combining the benefits of both rule-based and statistical approaches is therefore adopted by OpenDial.

It is already common knowledge that different languages require an adjusted speech recognition for a better user understanding. In order to ensure an improved user experience, the adaptation to the features of a language seems therefore also necessary for the dialogue manager. Since language is the most important part of a culture, these two concepts are regarded as synonymous in this thesis. In addition, emotions are a fundamental aspect of human behaviour and provide valuable non-verbal information. Due to this, the dialogue manager has to decide on the basis of emotions as well.

The task of this work is to use the toolkit OpenDial within the existing dialogue management framework OwlSpeak to handle dynamically created dialogue actions and enable user-adaptive dialogue management. For this purpose, a training algorithm is applied which considers the emotional condition and cultural characteristics of a user. The proposed methods are then integrated and evaluated in the KRISTINA project which aims to provide an intelligent agent for elderly care.

This thesis is organized as follows: Chapter 2 opens with the provision of background information on the key aspects of this work. Related work on dialogue management and assistive agents is examined in Chapter 3. In Chapter 4, the technical basics of the learning algorithm are described. Afterwards, the implementation of the training process is outlined in Chapter 5. The experimental results of the training algorithm are shown in Chapter 6. Lastly, Chapter 7 summarises the work in a conclusion.

2. Background

This chapter presents a brief overview of the most important background information. First, an explanation of the concept Spoken Dialogue System is given. Subsequently, in Section 2.2, the dialogue manager OwlSpeak is introduced and its architecture is briefly described. The software toolkit OpenDial will be examined in Section 2.3, also outlining its domain structure and learning techniques. Section 2.4 then explains Bayesian networks which are the basis for the employed probability model. Afterwards, in Section 2.5, relevant information on the KRISTINA project is provided.

2.1. Spoken Dialogue System

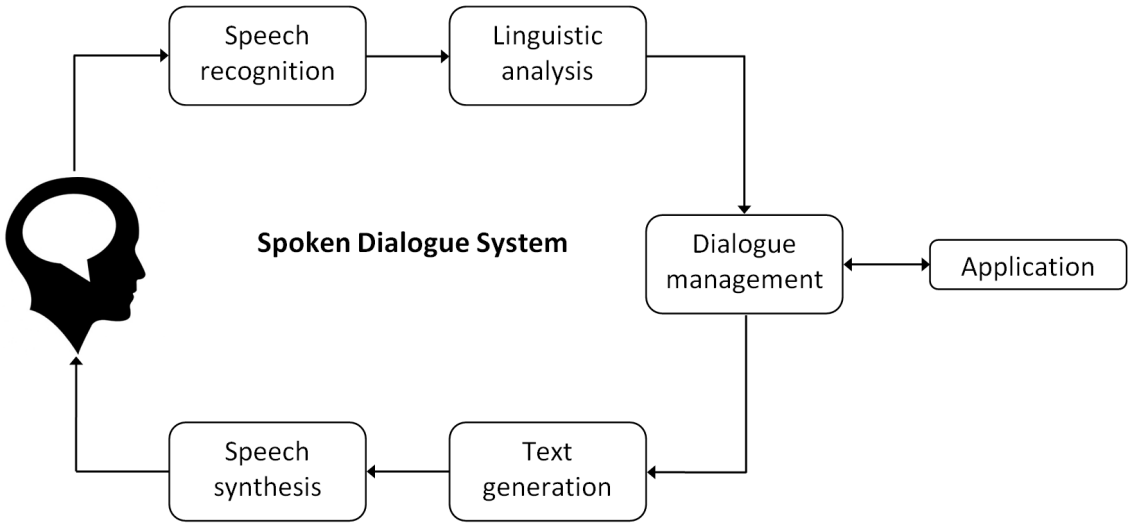

Figure 2.1.: Architecture of a Spoken Dialogue System

As soon as computer systems are involved, the question of the connection between human and machine arises. One particular kind of interface between those entities are Spoken Dialogue Systems (SDS). According to [25], there exist two main types of such systems: task-oriented and nontask-oriented. The first type accomplishes rather simple and predefined tasks using a dialogue, while the second one is not intended to cope with a fixed request and offer a straight solution but instead, engage a conversation. Nevertheless, a crossover between these categories is not ruled out but occasionally desired. Both types

of SDSs are based on the architecture shown in Figure 2.1 consisting of five distinct components. To begin with, the speech recognition receives the user's speech signal, which is then processed and converted into text. As output, the module provides a list of scored alternatives constructed as a word hypothesis graph; consequently, the output is forwarded to the linguistic analysis which evaluates syntactic and semantic characteristics, resulting to the transformation of the meaning of the user's utterance into a formal structure. Afterwards, the semantic representation is transmitted to the dialogue manager. Since this segment is responsible for the dialogue flow, it can be seen as the core of a SDS. The primary task of the dialogue management is the action selection for the next system turn. As a specific policy forms the basis for this decision, a well-considered design is required. Additionally, a dialogue history stores further knowledge about the previous and current states of the dialogue. The semantic representation of the selected action is then passed on to the text generation. There, grammatically correct structures are created depending on language properties. Concluding with the speech synthesis, the received text data is finally converted into a speech signal. This circular process ends thus at the starting point and illustrates the turn-based exchange between a human and a machine.
More detailed information is provided in [33]. This work focuses on the dialogue management which enables user adaptation. The underlying framework is addressed in the following section.

2.2. OwlSpeak

Originally, the dialogue manager was developed and presented in [22]. The Information State theory behind the implementation was described in [27]. This approach endeavours to represent relevant dialogue information in such a way that a flexible dialogue management becomes possible. OwlSpeak consists of the strictly separated model-view-presenter design outlined in [41]. As this splits data management, dialogue logic and dialogue interface, each part can be modified independently and the adding of extensions is facilitated. Figure 2.2 illustrates the general architecture of OwlSpeak which is implemented as a Java Servlet to allow communication with various speech devices. To enable the integration of the OpenDial toolkit and the KRISTINA project that will be introduced in Sections 2.3 and 2.5, some adjustment are necessary which are elucidated in the following subsections.

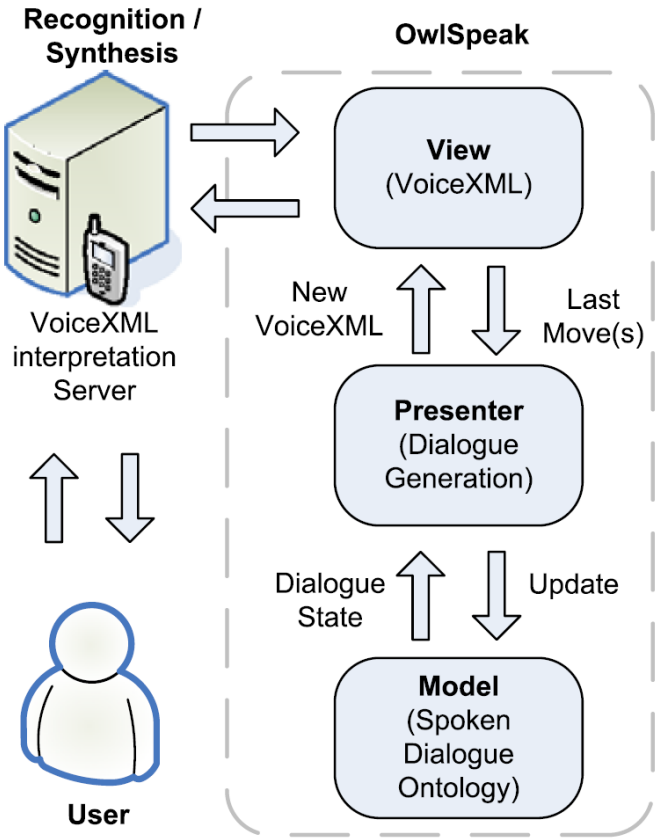

Figure 2.2.: The general architecture of OwlSpeak based on Heinroth et al. [22] following the model-view-presenter paradigm. Figure and its description are taken from [47].

Model

Since OwlSpeak is an ontology-based dialogue manager, this component is particularly relevant. The ontologies are formulated in the Web Ontology Language *OWL* which employs structuring in classes, relations between classes and individuals. For each domain, such a file is created. It is divided into a static *Speech* and a dynamic *State* part. While the *Speech* part contains concepts of the dialogue, the *State* part gets updated at each turn during the dialogue.

View

This layer represents the interface between the speech recognizer, the speech synthesis and OwlSpeak. The view is created by the presenter at each dialogue turn and afterwards sent to the synthesis. There, a for humans understandable utterance is created and emitted. In contrast, a received input from the user is transcribed into the formal structure and sent back to the presenter. Although it was originally intended to operate with VoiceXML documents, for this work, the substitution with KRISTINA documents is necessary. This type stores information on scenario and user-specific variables in order to include emotions and further dialogue history knowledge in the domain.

Presenter

The presenter is the logical core of OwlSpeak. Here, the system selects the next action according to the dialogue policy. This move is called *Agenda*, the entry point of a dialogue is defined in a *MasterAgenda*. For the KRISTINA project, an additional component in form of a knowledge integration module provides dynamic dialogue data that contributes to this decision. In this work, however, OpenDial is responsible for this layer. This toolkit is introduced in the next section.

2.3. OpenDial

OpenDial has been developed as a Java-based toolkit for SDS described in [28]. Similar to OwlSpeak, it bases on the Information State theory. However, both dialogue systems differ in their architecture since OpenDial is designed in the blackboard architecture. As stated in [13], this approach is intended to cope with complex scenarios where a difficult task is worked on cooperatively by multiple specialists. Although OpenDial was originally implemented as a pure dialogue manager, the flexible architecture shown in Figure 2.3 enables the integration of diverse modules. Encoded as a Bayesian network, the combination of modules like speech recognition, language understanding, language generation and speech synthesis is capable to form a complete dialogue system as needed. OpenDial goes for the approach to combine logical and statistical dialogue models in order to unite the advantages of both concepts. This is possible through the application of probabilistic rules which represent the domain model in a well structured manner. These rules allow system designers to bring in their knowledge of the problem structure which is then considered in the statistical model. Moreover, these rules contain unknown parameters. Their estimation relies on either supervised or reinforcement learning techniques. Thus, this hybrid concept permits experts to integrate domain-dependent constraints into a probabilistic context.

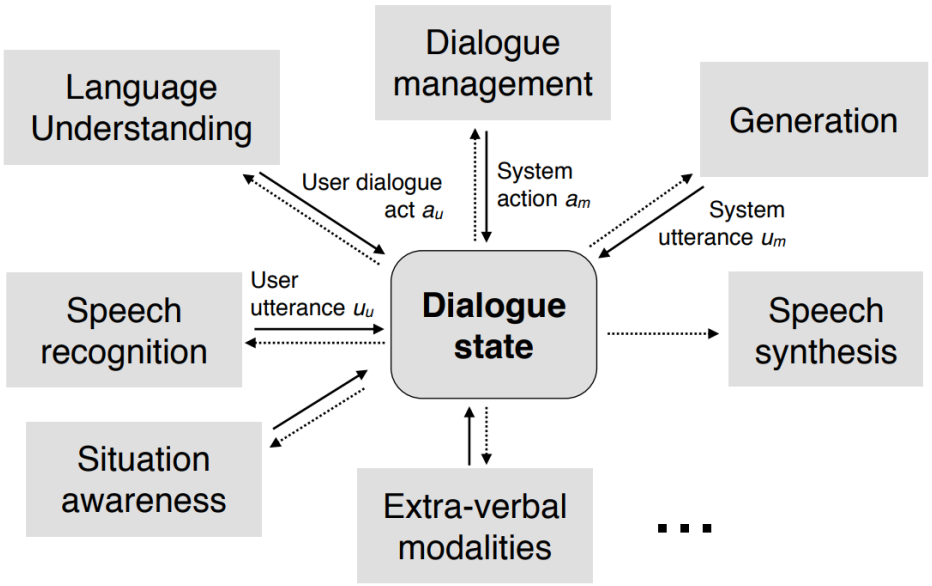

Figure 2.3.: Architecture of OpenDial. Figure is taken from [28]

The following subsections provide an overview of the most relevant aspects of the dialogue manager. A more detailed description is given in the doctoral dissertation of Pierre Lison [29], who is also the main developer of OpenDial.

2.3.1. Dialogue domain

It has already been shown that in OpenDial, the dialogue state represents the blackboard while the attached modules perform the processing. In order to work domain-independent, the characteristics have to be specified for each domain. This is realized by storing the complete knowledge about a domain in files encoded in Extensible Markup Language(XML) [12]. The illustration of the general structure can be seen in Figure 2.4. It will be described in the following subsections.

```
<domain>
        <initialstate>
                ...
        </initialstate>
        <parameter>
                ...
        </parameter>
        <model trigger="exampleTriggerVariable1">
                <rule id="exampleRule1">
                        ...
                </rule>
                ...
                <rule id="exampleRuleN">
                </rule>
        </model>
        ...
        <model trigger="exampleTriggerVariable2">
                <rule id="exampleRuleA">
                        ...
                </rule>
                ...
        </model>
</domain>
```

Figure 2.4.: General structure of a domain

Domain and Initial State

The document starts with the declaration of the domain. Since OpenDial does not support multi-domain dialogue management, the file consists of one domain. Next, the initial dialogue state is set as a list of state variables, each represented by a probability density function. However, due to architectural reasons, this is not used in this work and thus finds no further consideration.

Parameter

OpenDial bases on the assumption that an expert creates the domain rules using specific domain knowledge. These rules depend on certain values which are problematic to predetermine for most domains. This issue is overcome by employing unknown parameters. They are defined either as fixed numbers or probabilistic distributions and are afterwards estimated through Bayesian learning. In this toolkit the available types of prior distributions are categorical, uniform, Gaussian, and Dirichlet. An exemplary definition can be seen in Figure 2.5.

```
<parameter>
        <variable id="exampleGaussian">
                <distrib type="gaussian">
                        <mean>0</mean>
                        <variance>1</variance>
                </distrib>
        </variable>
</parameter>
```

Figure 2.5.: Definition of a standard normal distribution

Model

The logical part of each domain is defined in its models. With reference to the blackboard architecture, the dialogue state is the shared-memory for all components working on it. A model gets thus called as soon as the trigger condition is met, whereby it is possible that this condition depends on more than one variable. The models process an input based on the specification by several probabilistic rules. A distinction is made between probability and utility rules:

- **Probability rules** are formally described by $P(Output \mid Input)$ with both variables being subsets of state variables. This can be regarded as the probabilistic impact an input has on conditional outcomes.
- **Utility rules** are formally described by $U(Action \mid Input)$, with particular system actions depending on the state variable input. From the system's point of view, this expresses the extent to which a certain reaction is desired.

Given these fundamental differences, the decision on which type is used relies on the intended application. In practice, utility rules are preferred for action selection, whereas probability rules are deployed for language understanding and prediction. Both share the *condition-effect* pattern which is shown in Figure 2.6.

```
<rule id="exampleRule">
        <case>
                <condition>
                        <if var="input" value="I" />
                </condition>
                <effect (prob/util)="x">
                        <set var="output" value="O" />
                </effect>
        </case>
</rule>
```

Figure 2.6.: Example rule consisting of one condition and one effect

As can be seen, the *condition-effect* pattern equals the *if-then-else* constructs which are well-known in information technology. If the conditions are fulfilled, then the determined effects are executed. A rule is divided into one or more cases corresponding to a separate branch. Each of them contains arbitrarily nested conditions and mutually exclusive effects. Sub-conditions are aggregated through the application of conjunction and disjunction operators. The effects are either assigned with fixed values or refer to parameters and the following processing depends on the used type. In case of probability rules, the assignment is an actual probability, which has to comply with the subsequent fundamental axioms:

$$0 \leq P(effect_i) \leq 1 \; \forall \; i \; and \; \sum_{i=1}^{n} P(effect_i) = 1 \tag{2.1}$$

If the sum of the probability of all effects is not equal 1, an auxiliary void effect is added by default. In the next step, these probabilities are joined for each possible outcome and normalized. The result is thus stochastic.

For utility rules, their possible values are not limited to the interval [0,1] and the sum is not required to be 1. Moreover, the assignments are used to alter the utility of each system action. These utilities are afterwards converted into an empirically constructed table, where the most desirable action has the highest utility and gets selected.

2.3.2. Learning techniques

Although hand-crafted rules are a reliable approach for straightforward domains, it is not practicable to apply them in complex scenarios. In general difficult to implement, the resulting dialogue policy would be too rigid and not adaptive at all. In contrast, the basic idea of statistical dialogue management is to use interaction data for the automatic optimization of such a policy. However, one drawback is that a large number of training data is required. OpenDial combines both concepts, as only the parameters of pre-defined rules are learned. Two training methods are available: supervised learning which is trained on the basis of Wizard-of-Oz data and reinforcement learning which learns through trial

and error from iterated interactions. Both are grounded in Bayesian inference and are described in the following subsections.

Supervised approach

Supervised learning is based on the principle that a knowledgeable supervisor labels a set of training data. These examples then are used to train the system, enabling it to identify the correct action for each covered situation. First of all, a collection of interaction data has to be gathered. As stated in [26], humans behave differently while talking to a machine instead of a human counterpart. The Wizard-of-Oz avoids possible biases in training data since the user assumes an interaction with a system. Instead, a human remotely operates the dialogue flow. Assuming that the wizard aims to select the most appropriate system action, the responses are thereby interpreted as a 'gold standard'. The resulting dialogues are then used for training the rule parameters of the domain. The policy thus imitates the wizard's behaviour. According to [20], the use of this paradigm leads to a fast and reliable result, as only the wizard needs profound expert knowledge. For the subjects, just a minimal introduction is necessary. Nevertheless, it is claimed in [43], that the wizard is able to cope with situations in which machines have troubles, for example recognition problems or system lags. These deficiencies are therefore not covered by the policy and have to be considered by the wizard on his actions.

In addition to the conventional Wizard-of-Oz scenario between a user and a wizard, OpenDial offers the functionality to save and read pre-scripted interaction files in XML format.

Reinforcement learning

A weakness of supervised approaches is that the amount of labelled training data is severely limited. In addition, it consists exclusively of situations considered by the expert, which makes the policy inoperable in unfamiliar scenarios. To avoid these constraints, reinforcement learning aims to optimize the dialogue policy through trial and error. The main component of this approach is an agent interacting with its environment. Each action is associated with a certain influence and is rewarded with an immediate value. The reward, whether positive or negative, expresses how suitable the behaviour is. Thus, it can learn from its own experience in unknown scenarios, where according to [46], learning is most advantageous. OpenDial utilizes Bayesian reinforcement learning, which leads, as described in [49], to a faster learning process by defining a prior distribution. It can be trained with either model-based or model-free optimization strategies. For the model-based approach, updating the parameter distribution relies on observations and rewards that the system receives. The model-free approach, on the other hand, uses a temporal-difference learning method for the update of the parameters. Since it takes a lot of dialogue cycles to achieve a sufficient policy, the training with real user data is not possible in many domains. For this, OpenDial offers a user simulator to generate the necessary data.

Regardless of whether model-based or model-free approaches are employed, a reward model is required in both cases. As examined extensively in [34], it is difficult to develop

an appropriate reward function even in domains with rather low complexity. Facing the various requirements in the KRISTINA domain, it was not possible to create a reward model. Therefore, reinforcement learning is not used in this thesis. The challenges that are responsible for this decision are outlined in Section 2.5. However, the following section introduces Bayesian networks which are conceptually necessary for the learning algorithm in OpenDial.

2.4. Bayesian network

As described in Section 2.3, OpenDial's probabilistic model is encoded as a Bayesian network. This type of a directed graphical model represents a set of random variables and their conditional dependencies in form of nodes and directed edges. The direction of the arrow indicates the relationship between a *parent* node and a *child* node. This influence continues on a direct path, these nodes are then called *ancestors* and *descendants*. The structure as acyclic graph prevents loops and therefore no node can be its own *ancestor* or its own *descendant*. Formally, the joint distribution of all random variables $(X_1, X_2, \ldots, X_n)$ in the network is given as:

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid parents(X_i)) \tag{2.2}$$

Where $parents(X_i)$ is the set of parents of X_i. If a node does not have a *parent* node, its probability distribution is called unconditional. A more detailed description is provided in [8].

Figure 2.7 shows an example of a Bayesian network that consists of the random variables *Rain*, *Sprinkler* and *Grass wet*. Only *Rain* has an unconditional probability distribution since it is the only node without *parent* node. In contrast, *Sprinkler* and *Grass wet* have conditional probability distributions.

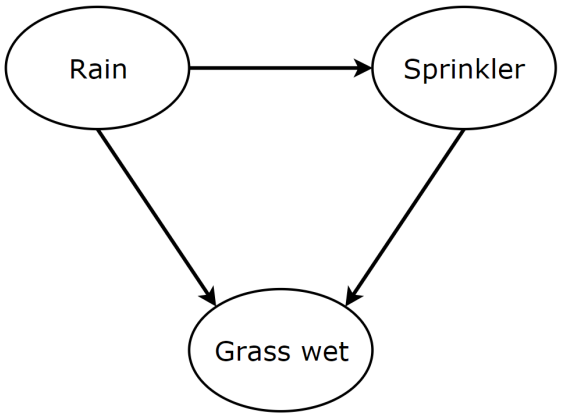

Figure 2.7.: Example of a Bayesian network consisting of three nodes and three edges

However, this model is not prepared for use in dynamic environments such as dialogue management. Therefore, an extension is necessary which expresses the variables as a function of time. For this, two presumptions are required: the Markov property and the stationarity of the process. The Markov assumption implies that the probability distribution of random variables in a current state depends only on the previous state. Let $\mathbf{X}_t$ be an arbitrary collection of random variables at time t. The Markov property can then be formulated as follows:

$$P(\mathbf{X}_t \mid \mathbf{X}_{t-1}, \ldots, \mathbf{X}_0) = P(\mathbf{X}_t \mid \mathbf{X}_{t-1}) \tag{2.3}$$

Furthermore, a stochastic process is called stationary if its probability distribution is time-invariant. Assuming that $n \in \mathbb{N}, \tau \in \mathbb{Z},$ and $(t_1, t_2, \ldots, t_n) \in \mathbb{Z}^n$, the strict stationarity is expressed as:

$$(\mathbf{X}_{t_1}, \mathbf{X}_{t_2}, \ldots, \mathbf{X}_{t_n}) = (\mathbf{X}_{t_1+\tau}, \mathbf{X}_{t_2+\tau}, \ldots, \mathbf{X}_{t_n+\tau}) \tag{2.4}$$

Given the inertial probability distribution $P(\mathbf{X}_0)$, both assumptions together define $P(\mathbf{X}_t \mid \mathbf{X}_{t-1})$ as a distribution which can be calculated for each time t depending only on its value at time *t-1*. The resulting model is called a *dynamic Bayesian network*.

In addition to temporal processes, OpenDial implements actions and utilities to extend the approach to decision-theoretical problems. The dialogue management is thus able to estimate and evaluate the utilities of system actions which are available in the dialogue state. This type of model is also referred to as a *dynamic decision network*.

Since the formal principles are now outlined, the following section introduces the context in which the dialogue management is supposed to operate.

2.5. KRISTINA

The KRISTINA project is part of the *Horizon 2020* research programme of the European Union [3]. It is structured in several work packages which are worked on cooperatively by partners all over Europe. The main task is the development of a socially skilled agent which supports migrants with linguistic and cultural difficulties and also serves as a contact point to receive information about elderly care. Some further tasks are:

- Dialogue management with respect to cultural aspects and social context
- Vocal communication analysis of scenarios including noise and emotions in different languages
- Gesture analysis aware of cultural and emotional characteristics
- Providing of medical and healthcare information by searching online

In this context, the Ulm University is responsible for the dialogue management. This module is the logical core of the KRISTINA agent as it controls the overall interaction. In general, a SDS serves as an interface between a user and a computer application. Systems currently in use aim to provide easy access to a service, but do not consider the

user's characteristics. This is sufficient in domains where a task can be accomplished independently of fundamental human components, for example the process of ordering a pizza. Although this assumption often applies to task-oriented agents, according to [11], nontask-oriented dialogue systems definitely require these influential conversational factors. In KRISTINA, both categories are intended: For instance, a user can ask the system to provide a weather report or engage a conversation about the course of the day. Furthermore, since the agent is designed to operate in highly sensitive domains such as medical care or the support of migrants, the dialogue manager has to adapt to two factors: emotional sensitivity and cultural awareness. The requirements are outlined in the following subsections.

2.5.1. Emotion

One key aspect of the effectiveness of a SDS is its acceptance by the user. According to [9], the user's acceptance increases if the system considers the emotional condition during an interaction. Since medical topics are mostly intimate, a trustful relationship is a precondition for a successful dialogue. In the KRISTINA project, emotions are defined on the *valence-arousal* scale which is categorized on several levels in two dimensions. In order to appear as natural as possible, the system is not only required to react to emotions, but also to express them itself. However, the dialogue management is not responsible for the actual representation but for the selection of an appropriate emotion. As isolated minorities of citizens are included in the target group, the decision of appropriate reactions is particularly challenging. Since emotions are difficult to express through strict rules, the hybrid concept of OpenDial offers a suitable approach for this model.

According to [6], the cultural background of people has influence on their perception of emotions. Hence, the next subsection addresses the cultural aspects.

2.5.2. Culture

For humans, the culture is of immense importance as it influences daily life and social behaviour. As stated in [7], among other things, the communication style varies between different cultures. It is therefore assumed that communication with migrants occasionally leads to misunderstandings due to cultural peculiarities. Since the KRISTINA agent provides sensitive information on medical topics, mistakes have to be avoided as possible. Thus, it is essential to adjust the system's behaviour to the culture of the user. As there are naturally a large number of cultures in the world, the KRISTINA project is limited to Arabic, German, Polish, Spanish, and Turkish. In this context, the dialogue management is able to influence the communication style by its rhetorical style, which implies a consistent flow of system moves based on cultural aspects. Moreover, the dialogue manager can be designed to vary the directness and verbosity of system responses. Though, this thesis only deals with the development of a policy for action selection.

These cultural and emotional aspects in combination with an individual dialogue flow therefore prevented the design of a reward model. However, the approach of supervised learning is still a promising alternative. The employed learning algorithm will be presented in Chapter 4. The following chapter examines related work.

3. Related Work

This chapter surveys related work on dialogue management and communicative agents. Currently, assistants such as Amazon Alexa, Apple Siri or Microsoft Cortana are probably the most famous representatives of SDSs. According to [2], these systems are designed as intelligent agents to support users in their daily lives. As stated in [14], the increasing power of hardware was responsible for the significant progress in speech recognition and speech synthesis. However, the dialogue management of these agents is still task-oriented. Moreover, mostly rule-based approaches are employed as explained in [1]. Since this method is only applicable in domains with low complexity, the motivation of current research is to develop statistical techniques.

In [48], the open-source toolkit *PyDial* for statistical SDSs is introduced which offers an end-to-end model for multi-domain conversations. Though, as described in [21], it requires an enormous amount of dialogue data to train the policy. As there exists no reward model in the KRISTINA domain, approaches with user simulators are excluded. To improve the efficiency of the learning process with real dialogue data, corpora of human-machine interactions are collected. An example of such a collection is reported in [18] where recordings of Wizard-of-Oz interactions were taken. Although this database contains many hours of dialogue data, its use is yet limited to a narrow field of domains. Furthermore, it is doubted in [44] that fixed corpora are a reasonable approach to train a dialogue policy. Accordingly, for nontask-oriented SDS, a corpus has to be unlimited in size in order to cover all possible dialogue states, which is not achievable in practice. Another major drawback common to all statistical approaches is that the training process does not guarantee an adequate outcome. Since the KRISTINA domain is designed for sensitive areas, a malformed policy is especially dramatic. Moreover, multi-domain dialogue management is not necessary here. Therefore, probabilistic rules minimize the aforementioned risks by structuring the fundamental characteristics of a single domain and then learn the posterior distribution of the parameters.

As stated in [23], adaptation to the user influences the interaction between a human and a machine. Since the dialogue manager is responsible for the dialogue flow, it is outlined in [39] how emotions can be extracted from the speech signal and integrated as a stochastic emotional dialogue model. Afterwards, the transition probability between dialogue and emotional states is trained with Wizard-of-Oz data. However, this approach does not include cultural aspects.

Currently, there is a lot of research on cultural adaptation of user interfaces; as described in [42], the aim is to enhance the usability by this. Depending on the culture of a user, the design and content of websites is adapted. This results in a more satisfying experience for a user and an increase in market share for the system provider. Since SDSs are a special form of interface, it is reasonable that these effects can be transferred to them.

According to [37], the *World Health Organization* predicts that elderly people in some regions will account for more than 30 percent of the population by 2050. Europe will be particularly affected by this development, since the ratio of caregivers to elderly people will decrease in the future. As stated in [19], independent living with home-based care is preferred by seniors to facility-based care since it retains their familiar surroundings. However, there are not enough ambulant nursing workers available for this purpose, even though this type of caregiving reduces costs to society compared to nursing homes. Projects such as KRISTINA aim to provide patients and their families with a contact point and thus reduce the need for human support. Several approaches are listed below.

The SWEET-HOME project presented in [40] offers a *smart home system* that was developed especially for elderly people. For this, omnidirectional microphones are employed which capture natural voice and everyday sounds such as clapping hands. As a result, users who are unfamiliar with computer systems do not have to endure a time-consuming instruction. This project targets seniors who are still autonomous and seek accompany during their daily life. Nevertheless, the lack of multi-modality limits the applicability of the system regarding ubiquitous computing.

Another approach to compensating for the deficit of nursing staff are *socially assistive robots*. A representative of this category is the nursebot *Pearl* which is described in [38]. This robot is capable of communicating with its environment through speech, graphical display, and motion. The target group is nursing staff as well as slightly impaired seniors. One feature is the plan manager that reminds elderly people about their activities. This project also implements a dialogue management which optimizes its policy with reinforcement learning. However, an adaptation to a user is not supported here.

A disadvantage for *smart home systems* is that seniors are obliged to enter different parts of the house to access the service. A drawback of *socially assistive robots* is that a robot is inefficient since it requires interaction with the home environment to control the system. Therefore, the KSERA project which is introduced in [24] endeavours to combine the benefits of both approaches. For this, the *smart home system* is represented via a *socially assistive robot* which in turn is controlled by sensor information collected by the smart home. Furthermore, the robot interacts with elderly people in order to support them in their daily life. However, this approach still lacks user adaptation.

Finally, an interesting opinion is presented in [35]. Accordingly, contemporary SDSs lead to a *habitability gap* due to the mismatch between the features offered by systems and the expectations of users. Thereby, the employment of natural voice evokes the impression of human-like competence among users. However, the current systems are limited in their beneficial use. Since this results in an unpleasant experience, it is proposed to make agents less humane, for example by using a robot voice, until they are indeed able to meet expectations.

4. Learning algorithm

This thesis focuses on the method of supervised learning since the requirements of the KRISTINA domain depicted in Section 2.5 did not allow a proper reward model which is necessary for reinforcement learning techniques. As the implementation of a training algorithm was part of this work, the formal basis of the applied Wizard-of-Oz learning is outlined in this chapter.

This chapter is ordered as follows. Section 4.1 outlines the correlation between parameters and prior distributions and explains the classification of node types. The architecture of the training process for the parameter estimation is described in Section 4.2

4.1. Node types in OpenDial

As already mentioned in Section 2.3, parameters of rules can be expressed either by fixed values or by prior distributions. However, fixed values are intended for scenarios in which probabilities are well-known. Prior distributions, on the other hand, allow an expression of uncertainty in the domain. An expert creating probabilistic rules is thus able to let statistical methods estimate the actual values on the basis of training data. OpenDial relies on Bayesian inference which requires the definition of a prior probability distribution. This complies with the intention of the toolkit since a system designer has usually knowledge of the problem structure and the domain itself, but has problems with the estimation of exact probabilities as noted in [36].

In OpenDial, every probabilistic rule, parameter, and state variable is instantiated as a node in a Bayesian network. The underlying conceptual model has already been introduced in 2.4. An example is given with Figure 4.1, where the visualized dialogue state consists of the input node a_u, the parameters θ_1, θ_2, θ_3, the utility nodes $rule_1$, $rule_2$, $rule_3$, and the decision node a'_m. For reasons of clarity, the prime indicates an output or decision variable. The connections between the nodes illustrate their dependencies according to their direction. While probability rules cause the creation of output nodes, utility rules instantiate new decision nodes which in turn serve as an input to the utility nodes. In the described figure, the input node represents a user action and the decision node represents a system action, the displayed dialogue state is hence the action selection. The rules can depend on more than one parameter and also do not necessarily require a state variable as input. In context of the action selection, however, it is assumed that the probability of input state variables is always equal to 1. This presupposes that speech recognition and linguistic analysis identify the intended user action without error.

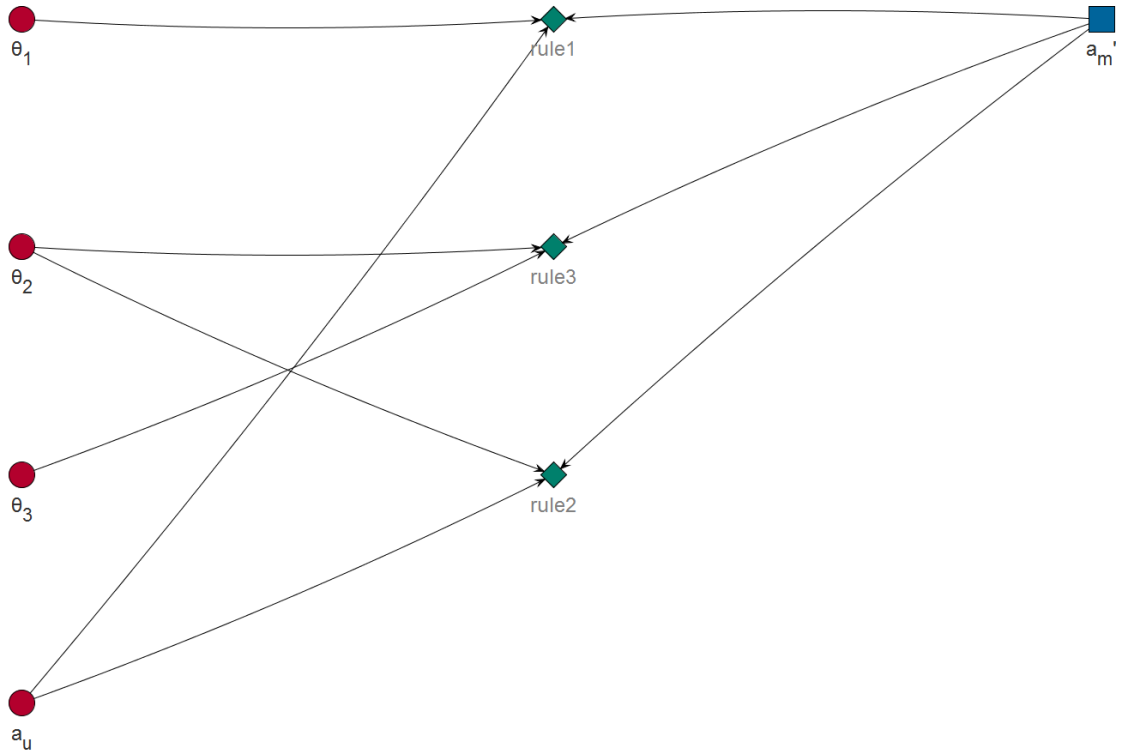

Figure 4.1.: Example of a Bayesian network representing the current dialogue state

The nodes of the decision network are thus divided into the following categories:

- **Chance nodes** represent classical random variables. The values of these nodes are conditional probability distributions which depend on the probability of the parent nodes. However, this dependency can not be displayed since the figure is limited to one dialogue state. By convention, chance nodes are visualized as circles.
- **Decision nodes** are used for variables administrated by the system. Decision nodes are linked to values that express an active choice of the system on certain actions. Therefore, these nodes are alternatively called action nodes and are represented by squares.
- **Utility nodes** correspond to the utilities of certain situations determined by the parent nodes. Since these can be both chance and decision nodes, the utility nodes express every combination of their values in the form of utility distributions. By convention, utility nodes are presented as diamonds.

Consequently, further processing depends on the category and is outlined in the next section.

4.2. Estimation of rule parameters

This section describes the estimation of rule parameters using the Wizard-of-Oz learning method. Since this approach aims to model a human-machine interaction, there are certain assumptions about the wizard's behaviour. In order to enable a meaningful parameter estimation, the wizard is supposed to be a rational agent who tends to select the most useful action in the current dialogue state. However, in a conversational situation, it is possible that there are several appropriate system reactions which are equally correct. This leads frequently to a different selection of wizard responses in similar dialogue states. As a result, the wizard actions are not interpreted as an absolute 'gold standard' that implies a unique, accurate output. Moreover, it is expected that the training data eventually contain errors and inconsistencies. Therefore, a level of confidence for the wizard decisions is introduced.

In general, a Wizard-of-Oz interaction is defined as a sequence of state-action pairs:

$$\mathcal{D} = \{\langle \mathcal{B}_i, a_i \rangle : 1 \leq i \leq n\} \tag{4.1}$$

with $\mathcal{B}_i$ describing the dialogue state and a_i representing the performed wizard action at time i. The total number of recorded actions is expressed with the positive integer n. As already explained, the dialogue state is mapped as a Bayesian network and contains information of the dialogue history and contextual information. If the wizard does not specify a response, this results in a void action for the corresponding dialogue state. Nevertheless, the level of confidence can be expressed by the likelihood:

$$P_{\mathcal{B}_i}(a_i \ ; \ \boldsymbol{\theta}) \tag{4.2}$$

Here, the utility of action a_i in state $\mathcal{B}_i$ depends on the parameters $\boldsymbol{\theta}$. This allows a comparison between all possible actions. Since the parameters are not random variables in the classic sense, the conditional probability is indicated by a semicolon instead of a vertical bar. This convention is described in more detail in [4].

The learning process estimates the posterior distribution of the rule parameters $\boldsymbol{\theta}$ based on the Wizard-of-Oz training data set $\mathcal{D}$, which can be defined as:

$$P(\boldsymbol{\theta} \mid \mathcal{D}) \tag{4.3}$$

For the processing of the algorithm, this equation has to be iterated for every state-action pair of the data set $\mathcal{D}$ and updates the posterior distribution of the parameters $\boldsymbol{\theta}$ after each step. The implementation of these formal constructs is shown in the following subsections.

Likelihood distribution

The likelihood defined in Equation 4.2 is crucial for the efficiency of the learning algorithm, as it is a measure of how well the wizard action a_i fits to the dialogue state $\mathcal{B}_i$. Therefore, the utility of all possible actions a has to be determined:

$$U_{\mathcal{B}_i}(a \ ; \ \boldsymbol{\theta}) \tag{4.4}$$

Starting with the highest value, the utilities are then ranked in descending order. Since the selected wizard action is considered to be fairly rational, it is expected to be on top of this list. Given the parameters $\boldsymbol{\theta}$, the normalisation factor η, the probability p, and the determined position in the ranked list x, the likelihood can be described formally as:

$$P_{\mathcal{B}_i}(a\ ;\ \boldsymbol{\theta}) = \begin{cases} \eta p & \text{for } a_i \text{ is the action with highest utility} \\ \eta(1-p)p & \text{for } a_i \text{ is the action with second-highest utility} \\ \eta(1-p)^2 p & \text{for } a_i \text{ is the action with third-highest utility} \\ \dots & \\ & = \eta(1-p)^{x-1}p \end{cases} \tag{4.5}$$

The probability p therefore expresses the relevance of the wizard actions for the learning algorithm. In other words, a high probability p means a high confidence in the rationality of the actions. This affects the learning rate of the estimation process as well: while a low value slows down the convergence of a dialogue policy that imitates the wizard, it is more robust to errors and inconsistencies. Equation 4.5 characterises the special case of a negative binomial distribution, the geometric distribution. As outlined in [50], this distribution describes the probability of waiting for x events before the first success occurs, while the probability of success for each event is p. This complies with the assumption that the wizard selects the action with the highest utility with the probability p. Since the geometric distribution is monotonically decreasing, an action with low utility has thus always less probability than an action with high utility. An example of such distribution with p=*0.5* is shown in Figure 4.2.

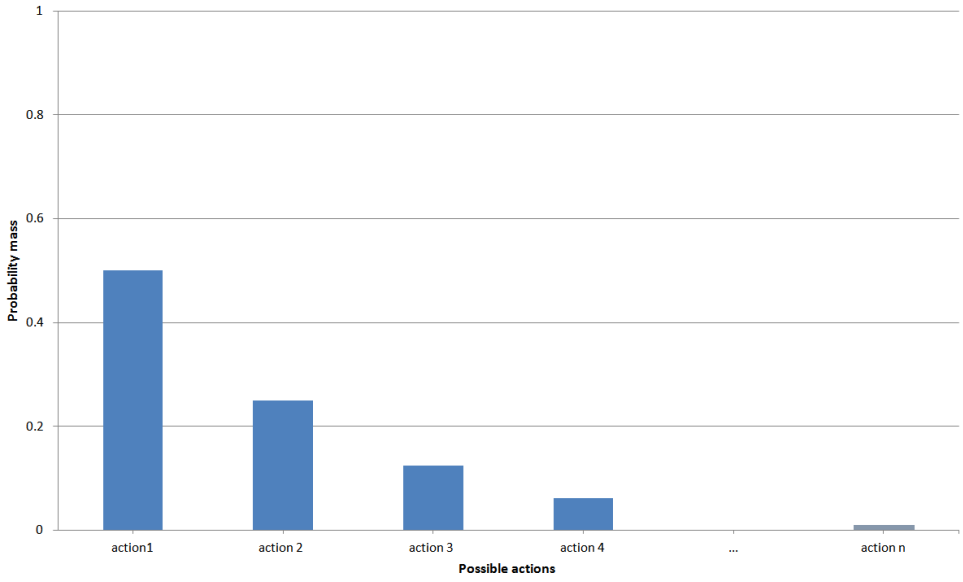

Figure 4.2.: Geometric distribution $P_{\mathcal{B}_i}(a_i\ ;\ \boldsymbol{\theta})$ for n actions with p=*0.5*

By definition, the geometric distribution is not finite. Therefore, the normalisation factor η restricts the distribution to a finite support. In the previous figure, the colour of the column of action n differs from the others, indicating that this value is not an actual but representational value. Accordingly, the utilities of the actions are ordered in this ranking:

$$U_{\mathcal{B}_i}(a_1\ ;\ \boldsymbol{\theta}) > U_{\mathcal{B}_i}(a_2\ ;\ \boldsymbol{\theta}) > U_{\mathcal{B}_i}(a_3\ ;\ \boldsymbol{\theta}) > U_{\mathcal{B}_i}(a_4\ ;\ \boldsymbol{\theta}) > \cdots > U_{\mathcal{B}_i}(a_n\ ;\ \boldsymbol{\theta}) \tag{4.6}$$

Consequently, if the wizard selects an action with poor utility, the probability of the likelihood is low. This influences the posterior parameter distribution which is described in the following.

Posterior parameter distribution

After the determination of the likelihood distribution and given the normalisation factor η, the posterior distribution over the parameters may be defined as:

$$P_{\mathcal{B}_i}(\boldsymbol{\theta} \mid a_i) = \eta P_{\mathcal{B}_i}(a_i\ ;\ \boldsymbol{\theta}) P(\boldsymbol{\theta}) \tag{4.7}$$

The formula is derived from the Bayes' rule. However, the probabilistic model thus contains both continuous and discrete random variables. This leads to a nontrivial inference problem. OpenDial offers two approaches to solve this issue: discretion of the range of parameter values into distinct buckets, hence converting continuous variables in discrete variables, and application of sampling techniques to approximate the inference process. Although both methods satisfy in theory, according to [29], the sampling techniques are favoured in practice as they operate more efficient. Therefore, this thesis focuses on this approach.

The basic task of a sampling algorithm is the estimation of the posterior distribution by acquiring a large number of samples. A rather simple though reliable method is called *likelihood weighting* that is described in more detail in [16]. This algorithm samples the random variables consecutively in topological order. Thereby, each sample is weighted according to its particular evidence in relation to all other variables. If utility variables are included in the model, an accumulated sum of the utility of all samples is generated additionally. In OpenDial, the term *sample* does not refer to the process of sampling an actual distribution to collect values. Instead, the calculation bases on pseudo-random number generators which depend on the prior distribution of the parameter.

Since probability rules and utility rules have a different architecture, they require separate methods. However, the algorithms presented in this section aim to clarify the applied estimation technique which is the basis of the training process. By this, the architectural differences are not regarded although the actual implementation depends on the type of the rule. Moreover, the following algorithms are taken from [29]. As the first step in the procedure, the pseudocode in Algorithm 1 shows the just mentioned sampling process with utility.

Algorithm 1 GET-SAMPLE ($\mathcal{B}$, $\mathbf{e}$)

Input: Bayesian/decision network $\mathcal{B}$ and evidence $\mathbf{e}$
Output: Full sample drawn from $\mathcal{B}$ together with a weight and utility

Let $X_1, \ldots, X_n$ be a topological ordering for the variables in $\mathcal{B}$
Initialise sample $\mathbf{x} \leftarrow \langle \mathbf{e} \rangle$
Initialise weight $w \leftarrow 1$ and utility $u \leftarrow 0$
for $i = 1, \ldots, n$ **do**
    if X_i is a chance variable and is included in the evidence **then**
        $w \leftarrow w \times P(X_i = \mathbf{e}(X_i) \mid \mathbf{x})$
    else if X_i is a chance or decision variable **then**
        $x_i \leftarrow$ sample value drawn from $P(X_i \mid \mathbf{x})$
        $\mathbf{x} \leftarrow \mathbf{x} \cup \langle x_i \rangle$
    else if X_i is a utility variable **then**
        $u \leftarrow u + X_i(\mathbf{x})$
    end if
end for
return $\mathbf{x}, w, u$

Here, $\mathbf{e}(X_i)$ expresses the value of the variable X_i in the assignment $\mathbf{e}$. However, this method is only an intermediate step. As shown in Algorithm 2, the weighting process accumulates a large number of samples. The exact amount is specified by the system designer and is a trade-off. Although many samples increase the significance of the estimation, they also delay the computing time. To ensure system execution, the sampling process is thus automatically terminated if it takes too long. The samples are collected in a vector for each query variable in $\mathbf{Q}$ and afterwards normalised. In general, the query variables calculate a marginal distribution if there is no assignment provided in the evidence.

Algorithm 2 LIKELIHOOD-WEIGHTING ($\mathcal{B}, \mathbf{Q}, \mathbf{e}, N$)

Input: Bayesian/decision network $\mathcal{B}$
Input: Set of query variables $\mathbf{Q}$ and evidence $\mathbf{e}$
Input: Number N of samples to draw
Output: Approximate posterior distribution $P(\mathbf{Q}|\mathbf{e})$

Let $\mathbf{W}$ be vectors of weighted counts for each possible value of $\mathbf{Q}$, initialised with zeros
for $i = 1 \rightarrow N$ **do**
    $\mathbf{x}, w \leftarrow$ GET-SAMPLE($\mathcal{B}, \mathbf{e}$)
    $\mathbf{W}[\mathbf{x}(\mathbf{Q})] \leftarrow \mathbf{W}[\mathbf{x}(\mathbf{Q})] + w$
end for
Normalise the weighted counts in $\mathbf{W}$
return $\mathbf{W}$

Once the sampling process is finished for each parameter, the posterior distribution $P_{\mathcal{B}_i}(\boldsymbol{\theta} \mid a_i)$ has to be identified. However, the likelihood weighting does not ensure that the posterior remains in the same probability distribution type as the prior. Furthermore, it is not possible to reconstruct some distribution families such as the Dirichlet by a finite amount of samples. Due to this, OpenDial applies a non-parametric strategy to represent the posterior distribution in the form of *kernel density estimation*. Given a set of samples $x_1, x_2, \ldots, x_n$ of a continuous variable X, the *kernel density estimator* may be defined as:

$$P(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{4.8}$$

where K is the kernel that is a non-negative function which integrates to 1 and the so-called *bandwidth* $h > 0$ is a smoothing parameter. Although there exist several possible kernel functions, OpenDial uses only Gaussian kernels. Hence, the resulting distribution consists of n normal distributions which have their means at x_i and are smoothed corresponding to the bandwidth. Further material on this topic is provided in [15].

Thus, a representation of the posterior distribution is obtained. Subsequently, the actual learning algorithm is described.

Learning algorithm

With the principles of the previous sections, the learning algorithm for estimating model parameters based on Wizard-of-Oz data is shown in Algorithm 3. The procedure iterates for each state-action pair $\langle \mathcal{B}_i, a_i \rangle$. If a trigger condition is satisfied, a model instantiates the current dialogue state. Thereby, the prior distribution of the parameters is called. After the likelihood of the wizard action is calculated, the posterior parameter distribution is sampled via likelihood-weighting. The outcome is then expressed as a *Kernel density estimator*. This procedure is performed as long as training data is available.

Algorithm 3 WIZARD-OF-OZ LEARNING $(\mathcal{M}, \boldsymbol{\theta}, \mathcal{D}, N)$

Input: Rule-structured models $\mathcal{M}$ for the domain
Input: Model parameters $\boldsymbol{\theta}$ with prior distribution $P(\boldsymbol{\theta})$
Input: Wizard-of-Oz data set $\mathcal{D} = \{\langle \mathcal{B}_i, a_i \rangle : 1 \leq i \leq n\}$
Input: Number N of samples to draw for each learning example
Output: Posterior distribution $P(\boldsymbol{\theta} \mid \mathcal{D})$ for the parameters

1: **for all** $\langle \mathcal{B}_i, a_i \rangle \in \mathcal{D}$ **do**
2: Set $\mathcal{B}_i \leftarrow \mathcal{B}_i \cup \boldsymbol{\theta}$
3: Trigger models $\mathcal{M}$ on $\mathcal{B}_i$
4: Draw N samples $\mathbf{x}_1, \ldots, \mathbf{x}_N$ from posterior $P_{\mathcal{B}_i}(\boldsymbol{\theta} \mid a_i) = \eta P_{\mathcal{B}_i}(a_i \ ; \ \boldsymbol{\theta}) P(\boldsymbol{\theta})$
5: **for all** parameter variable $\theta \in \boldsymbol{\theta}$ **do**
6: Set $P(\theta) \leftarrow$ Kernel density estimator$(\mathbf{x}_1(\theta), \ldots, \mathbf{x}_N(\theta))$
7: **end for**
8: **end for**
9: **return** $P(\boldsymbol{\theta})$

The principles of the estimation process have thus been shown. The next chapter will outline the implementation of the presented algorithm.

5. Implementation

This chapter addresses the main part of the work for this thesis which is the enabling of user-adaptive dialogue management. For this, the toolkit OpenDial is used which provides probabilistic rules that have been described in Section 2.3. Since this approach relies on the estimation of the posterior distribution of the parameters, the training algorithm introduced in Chapter 4 is applied. The context of the dialogue management is the KRISTINA project that has been presented in Section 2.5.

The integration of OpenDial into OwlSpeak is depicted in Section 5.1. Section 5.2 explains the design of a domain in OpenDial regarding emotional and cultural aspects. Afterwards, the implementation of the learning algorithm is described in Section 5.3.

5.1. Integration of OpenDial into OwlSpeak

As previously described in Chapter 2, both OpenDial and OwlSpeak offer independent approaches for the dialogue management. In order to enable complementary work on this task, the responsibilities have to be split up. Since OwlSpeak was originally the exclusive dialogue manager for the KRISTINA domain, it was already optimized to process data provided by attached modules such as the knowledge integration. To avoid disturbing the workflow, this basic configuration has been retained subsequently. Though, the rule-based and the purely statistical methods in OwlSpeak are not sufficient to achieve a proper dialogue policy in the KRISTINA domain as there is neither the possibility to create adequate rules nor is there enough training data available. As a result, OpenDial is employed for the action selection for the next system move using hybrid probabilistic rules.

In the beginning of this work, as presented in [31], OpenDial was integrated in OwlSpeak as an Apache Maven project. The major advantage of this build automation tool for Java is its dependency management within a software project that ensures a rapid and trouble-free deployment. A detailed description of Maven is given in [45]. However, using this method causes the OpenDial to perform like a black box making it impossible to edit the program code. As several modifications were necessary in the course of this thesis, the approach was discarded. Instead, OpenDial's development code was integrated as a standalone Java project. This enabled the thorough examination of all functionalities and allowed their adjustment.

As stated in [28], OpenDial is designed to be operated by a *graphical user interface* (GUI) which facilitates the user to select a dialogue domain and monitor its behaviour in three different views. However, the large number of rules and parameters in the KRISTINA domain caused a memory problem during the training process. This led to severe delays or complete crashes of the GUI. Due to this, the entire dialogue manager was inoperable.

Although OpenDial also accepts instructions via command line, these are required to be specified before execution. This is suitable for static settings such as the path of the domain file. Since the employed learning algorithm consists of an arbitrary amount of training cycles, the approach is not applicable here. Therefore, a different type of control was necessary which utilizes the connection of the dialogue managers. So far, a client that connects to the OwlSpeak Servlet usually transmits data of the linguistic analysis, emotional characteristics, and further meta information. In order to control the embedded OpenDial, additional data is attached to the inquiry and processed afterwards. This allows the system designer to administrate the system without manipulating the Servlet. The following three control variables are sufficient to run the training algorithm. Although they express Boolean values, the data type *int* is preferred due to the subsequent processing.

- **isTraining** indicates whether a training cycle should be started. If it is set to 1, the dialogue manager switches in the Wizard-of-Oz learning mode and thus estimates the posterior distribution of the parameters. Requires the specification of a path to a pre-scripted interaction file. Once the procedure has finished, the dialogue manager stays in the learning mode and waits for new training data. For this, either the same file is used again or a different path is specified.
- **toExport** signals that the current parameter distribution has to be exported if it is set to 1. However, it can only be saved in the types categorical, uniform, Gaussian, and Dirichlet. Since the posterior distribution is expressed as a *Kernel density estimator* with Gaussian kernel function, the dialogue manager transforms the posterior distribution automatically to a Gaussian distribution. Requires the specification of a path to a folder where it should be saved.
- **terminateWizard** indicates if the training cycle should be ended. Since the dialogue manager is not able to perform the action selection in the learning mode, this variable terminates the procedure if it is set to 1. After that, OpenDial needs two dummy *userMoves* as input to function properly. This is due to architectural reasons.

Given this way of controlling the dialogue manager, the integration of OpenDial into OwlSpeak is complete. The following section outlines the design of the domain.

5.2. Domain design

This section describes the work on the domain file that represents the logical core of the dialogue manager. Here, the dialogue policy is defined which determines the action selection of the system depending on the action of the user. In OwlSpeak, the atomic dialogue step in an interaction is expressed as a *userMove* and a *systemMove*. However, the training algorithm of OpenDial requires a renaming in *a_u* and *a_m*. For better readability, the notation of OpenDial is only used in the code segments in this thesis.

The action selection model in the domain file is therefore triggered by the variable *userMove*. Further processing in the probabilistic rules depends on the dialogue action in this variable. In KRISTINA, there are a total of 68 dialogue actions. A complete list thereof is provided in the appendix. Nonetheless, not all of those dialogue actions can be performed by the user. In addition, some of them are grouped for linguistic reasons. For example, the dialogue actions *SimpleGreet, PersonalGreet, MorningGreet, EveningGreet, and AfternoonGreet* are summarized in the linguistic analysis as the dialogue action *Greet*. To find a basis for determining which pairs of dialogue actions occur during a conversation, an annotation database provides guidance. This corpus is explained in the following subsection.

5.2.1. Annotation database

Part of the work of the use-case partners in the KRISTINA consortium has been to record exemplary conversations between two humans in the KRISTINA domain. There exist transcripts in the languages Arabic, German, Polish, Spanish, and Turkish. Thereby, the aim was to identify the communicative requirements in the fields of elderly care and support of migrants. The recordings have been annotated by a group of student assistants of the Ulm University. The database is implemented in MySQL which is documented in [5]. The basic structure of the database for a dialogue in German is shown in Figure 5.1

DialogueID	DialogueActionNR	Participant	SpeakerID	Utterance	DialogueAction
de096	1	User	spk06f	hallo Kristina .	PersonalGreet
de096	2	System	spk04f	hallo Frau Schmidt .	PersonalGreet
de096	3	System	spk04f	wie geht's Ihnen heute ?	AskMood
de096	4	User	spk06f	danke ,	SimpleThank
de096	5	User	spk06f	es geht mir eigentlich gut . mir ist ein bisschen langweilig .	Declare
de096	6	User	spk06f	und ich kann ja jetzt nicht mehr Zeitung lesen .	Declare
de096	7	User	spk06f	könntest du mir vielleicht aus der Zeitung was vorlesen ?	RequestNewspaper
de096	8	System	spk04f	gerne ,	Accept
de096	9	System	spk04f	möchten Sie eine bestimmte Zeitung ?	RequestMissingInformation

Figure 5.1.: Basic structure of the database representing an excerpt from a dialogue

Here, the figure is limited to the most relevant columns. First, the *DialogueID* is a unique identifier which is assigned to each dialogue in the corpus. Next, the *DialogueActionNR* starts at 1 and increases incrementally for every dialogue action. Both variables together define the primary key of the database. *Participant* then specifies whether the user or

the system interacts in that dialogue action. This is followed by the *SpeakerID* which refers to another table that provides information about the speaker's characteristics such as the cultural background. The *Utterance* column contains the spoken content of a dialogue action. Finally, the intention behind the participants' utterance is stored in a *DialogueAction*.

The above excerpt from a conversation reveals several challenges. One issue is that a dialogue turn occasionally consists of multiple dialogue actions. Although this is completely normal in human interaction, it is difficult to model in a domain file. In this work, the system is therefore restricted to a maximum of two subsequent dialogue actions in order to keep the complexity of the policy manageable. In contrast, the dialogue action of the user is provided by the linguistic analysis.

Another problem is related to the emotional aspects of the dialogue. The student assistants attempted to assign a rating of how positive and surprising the actions were. This two variables are a substitution for the ratings of valence and arousal which are used for video data. However, the results were insufficient since the textual representation of a interaction is not suitable for extracting emotional information. This is particularly obvious in the line *DialogueActionNR*=6, where the utterance 'I am no longer able to read the newspaper.' could be a statement with either neutral or sad emotions. Due to this ambiguity, the database columns that refer to emotions were not usable in this work.

The KRISTINA project aims to employ multiple languages, but at the time of this thesis, only German and Polish dialogues were already annotated in the database. Therefore, the adaptation of the dialogue manager is limited to those two cultures. The next subsection presents how this data are used to create the domain rules.

5.2.2. Probabilistic rules

The fundamental decision on the design of the domain has been whether to employ probability rules or utility rules. After several tests, the use of probability rules was discarded since the implemented learning algorithm of OpenDial is not tailored to probabilities. Thereby, the values of the posterior distribution in form of a *Kernel density estimator* can not be limited to the interval [0,1] and its integral not to 1. If this condition is violated, no probability is assigned to a parameter which corrupts the action selection model. In addition, the likelihood procedure generates a ranking of actions according to their utility. If no utilities are specified, this leads to a conceptual mistake in the learning process since no likelihood can be calculated.

The utility rules are derived from the database whereby each dialogue action is covered by a rule whose effects correspond to the annotated data. Therefore, a SQL query extracts culture-independent all possible *systemMoves* in response to a *userMove*. Since it is possible to react with one ore two consecutive dialogue actions, two different queries are required which are provided exemplary for the 'Greet' rule in the appendix. Nevertheless, the requests are equivalent for the remaining rules. As already mentioned, a maximum of two consecutive *systemMoves* are selected. This approach of design ensures that unreasonable pairs of dialogue actions such as '*userMove:* Greet, *systemMove:* SayGoodbye' are excluded. However, not all plausible responses are represented in the database, for example is the

systemMove 'EveningGreet' never selected in the interactions. The system designer could obviously add effects that are not included in the database; but since in this case training data would be missing, the training algorithm would ignore these dialogue actions, which makes them unnecessary.

In the KRISTINA domain, specific dialogue knowledge is managed by a knowledge integration module. If information is available or necessary at some point during a conversation, this module provides the new variable *kiVariable*. If this variable is not empty, the dialogue manager is not allowed to respond with generic but preselected dialogue actions. Therefore, special rules are created which only apply once the *kiVariable* holds a value.

Another aspect in the design of the domain rules was the prior probability distribution of the parameters. In this work, it was assumed that each effect of a rule starts with the same distribution to express the uncertainty of the system designer. Thereby, experiments have been conducted for three families of probability distributions: Dirichlet, Gaussian, and Uniform. The results of these experiments are shown in Chapter 6.

To cover the adaptation to emotions, a separate rule was created. In the KRISTINA domain, as already mentioned in Section 2.5, an emotion is expressed as values in the valence-arousal scale. In OwlSpeak, these values are processed and a new variable *emotion* is triggered if a certain threshold is reached. This variable indicates that an emotional action is required, which is afterwards selected by OpenDial. However, the database is not sufficient for collecting training data since the textual representation of utterances prevents the extraction of emotional data. Due to this, the parameters in this rule are associated with fixed values. Although this limits the applicability drastically, the precondition allows a restriction to special cases which can be modelled with strict rules. At the beginning of this work, such a rule was already available and was subsequently used as a guideline.

Since only the action selection is outsourced to OpenDial, further processing remains part of the tasks of OwlSpeak. The *presenter* layer of OwlSpeak works with *Agendas*. Therefore, OpenDial is required to add the prefix 'agenda_' to each selected *systemMove*. If two consecutive dialogue actions are selected, their combination is expressed with a + symbol.

As an example, the implementation of the 'Greet' rule is depicted in Figure 5.2. The basic structure is the same for the remaining dialogue actions.

```
<rule id="greet">
        <case>
                <condition>
                        <if var="a_u" value="Greet"/>
                        <if var="kiVariable" value=""/>
                </condition>
                        <effect util="theta_Greet[0]">
                                <set var="a_m" value="agenda_PersonalGreet
                                                      +agenda_AskMood"/>
                        </effect>
                        <effect util="theta_Greet[1]">
                                <set var="a_m" value="agenda_PersonalGreet
                                                      +agenda_AskTask"/>
                        </effect>
                        <effect util="theta_Greet[2]">
                                <set var="a_m" value="agenda_PersonalGreet
                                                      +agenda_ShareJoy"/>
                        </effect>
                        <effect util="theta_Greet[3]">
                                <set var="a_m" value="agenda_AskMood"/>
                        </effect>
                        <effect util="theta_Greet[4]">
                                <set var="a_m" value="agenda_AskPlans"/>
                        </effect>
                        <effect util="theta_Greet[5]">
                                <set var="a_m" value="agenda_PersonalGreet"/>
                        </effect>
                        <effect util="theta_Greet[6]">
                                <set var="a_m" value="agenda_SimpleGreet"/>
                        </effect>
                        <effect util="theta_Greet[7]">
                                <set var="a_m" value="agenda_MorningGreet"/>
                        </effect>
        </case>
</rule>
```

Figure 5.2.: Implementation of the ‘Greet’ rule

5.3. Wizard-of-Oz learning

This section clarifies how the supervised learning technique is put into practice. The basic principle of the Wizard-of-Oz approach has already been introduced in Section 2.3. For this, OpenDial offers the functionality to connect two remote computers to each other. This simulates a scenario where a user does not know that the dialogue system is controlled by another human. However, this procedure did not work in the test cases. In addition to the selected action of the wizard, a response was provided by OpenDial itself. Although the instructions in the official documentation have been followed strictly, this conduct couldn't be suppressed. Nevertheless, an alternative option to estimate the posterior distribution of parameters is given with pre-scripted interaction files which are transcripts of Wizard-of-Oz dialogues. An example of such data is shown in Figure 5.3.

```
<interaction>
        <userTurn>
                <variable id="a_u">
                        <value>Greet</value>
                </variable>
        </userTurn>
        <systemTurn>
                <variable id="a_m">
                        <value>agenda_PersonalGreet</value>
                </variable>
        </systemTurn>
</interaction>
```

Figure 5.3.: Example of a transcript of a Wizard-of-Oz interaction

It is observable, that the dialogue is divided into a user turn and a system turn. Thereby, the value of the user turn has influence on the dialogue state. According to the utility rules in the model, a utility table is created that consists of all available system actions. Afterwards, the dialogue manager can calculate the likelihood of the wizard action with regard to the utility table. The example above demonstrates a fundamental problem: the *userMove* 'Greet' allows several, equally correct responses. For example, in this situation both dialogue actions 'PersonalGreet' and 'SimpleGreet' are valid. Subsequently, the expert who creates the probabilistic rules usually assigns the same prior distribution to the parameters of these *systemMoves*. Since the values of the prior distributions are obtained by sampling, each parameter is assigned with a different value and corresponding utility. The resulting list of equal *systemMoves* bases therefore on the random numbers of the sampling process. As the wizard may select an action that is at the end of this ranking, this causes a low probability of the likelihood which leads to a poor posterior distribution of the rule parameter, although it is a suitable response. Hence, a minimum amount of training data is necessary to overcome this probabilistic bias. The number depends of

course on the domain and its structure, but the 'Greet' rule in the previous example requires approximately 50 interactions for a meaningful parameter estimation. Another issue is the geometric factor which is of great importance as it indicates how relevant the wizard action is for the algorithm. A high value for the probability p causes thereby a fast convergence of the policy towards the behaviour of the wizard. However, this favours *systemMoves* that have positions at the end of the pre-scripted interaction file while the amount of their occurrence is neglected in the calculation. This contradicts the intention that the *systemMoves* with the highest occurrence in the Wizard-of-Oz interactions are also selected most frequently by the dialogue policy. Due to this, the probability p is set to a low value in this thesis, usually between 0.1 and 0.25. Its exact influence will be examined in the experiments in Chapter 6.

Although the domain file is the same for all cultures, the training files are not. As already shown in Figure 5.1, the database contains information about the sequence in which the dialogue actions are performed. However, in this work, it is assumed that the order and the selection of the next action depend on the culture. Since the interactions are between two humans, the participant in the role of the system is similar to the wizard in a Wizard-of-Oz scenario. As a result, two sets of training data were created, one in German and one in Polish. The experiments in Chapter 6 show the effects of the cultural adaptation.

6. Experimental results

This chapter presents the results of the experiments that have been conducted. After the implementation of the domain file and the development of trainings sets for the two cultures German and Polish, the next step was to identify the ideal settings for the training environment. This has been achieved by varying the prior distribution of the parameters and the geometric factor of the likelihood of the wizard action. Since this thesis focused on the dialogue management, the other modules of a SDS have not been examined. Due to this, the experiments rely on exemplary data provided by the linguistic analysis and do not employ actual speech recognition; correspondingly, the provided output of the dialogue manager is not transmitted to the speech synthesis. Moreover, the experiments were initially limited to a subdomain in order to allow a better observance. This restriction does not affect the applicability to the entire domain, although a complete evaluation is only possible for the available exemplary data.

In Section 6.1, it is presented how the most suitable prior distribution was identified. Afterwards, in Section 6.2, the influence of the geometric factor to the trained policy is described. The culture-dependent differences are then presented in 6.3. Finally, the extension of the subdomain to the entire domain is outlined in 6.4.

6.1. Prior distribution

The available types of prior distributions in OpenDial are categorical, uniform, Gaussian, and Dirichlet. Since the probability density of a dialogue action can not be expressed through discrete values, the categorical distribution is no appropriate representation and was therefore excluded from the experiments. However, the remaining three types have been tested intensively. The result was that the family of the prior distribution had a marginal influence on the training algorithm in the KRISTINA domain. This is due to the architecture of the likelihood estimation procedure which collects samples at each dialogue state and afterwards updates the posterior distribution. Consequently, the prior distribution which is defined in the domain file only applies for the first training interaction and is then substituted by a *Kernel density estimator*. Experiments have shown that after approximately 25 learning cycles, the original distribution can no longer be identified.

In contrast, the mean value of the distribution is an important decision. Since the action selection of the dialogue manager bases on the utilities of the actions, OpenDial ensures that the action with the highest utility is chosen. As this unit expresses how desirable an action is from the perspective of the system, a negative value indicates that an action is unfavourable. For this reason, the action selection has a lower limit for the utility at 0. If all possible actions have a negative associated utility, an automatically generated *void*

action is responded by the dialogue manager. Therefore, the mean value of the parameter distribution has to be selected larger than 0. For an uniform distribution and a normal distribution, this condition is easily met. If the type is a Dirichlet, the mean value cannot be defined since it is derived from the values of its characteristic parameters. However, if these parameters are set to a sufficient size, it is also guaranteed that one or more actions have a positive utility. In addition to the mean values, the experiments showed that it is necessary to have a small distance between minimum and maximum for an uniform distribution and to have a low variance for a Gaussian distribution in order to obtain more variability in the results.

In the following experiments, the training process is based on the German data set which includes a total of 77 interactions for the subdomain. For comparison, the Polish data set consists only of 12 interactions. Although the type of the prior distribution can be chosen arbitrarily in compliance with the above-mentioned conditions, the Dirichlet distribution is most suitable to represent the correlation between the effects of a rule. This type is therefore used in the actual implementation. However, since the multivariate character of this distribution type does not allow a clear graphical representation, the next figures illustrate the influence of the learning algorithm on a Gaussian distribution. Figure 6.1 visualizes the prior distribution before the training with a mean value of 5 and a variance of 1. In the rules, each effect depends on a parameter associated with the same distribution. The result of the learning procedure for the dialogue action 'PersonalGreet' that appears frequently in the training file is shown in Figure 6.2. In contrast, Figure 6.3 displays the outcome for the dialogue action 'AskTask' with few occurrence.

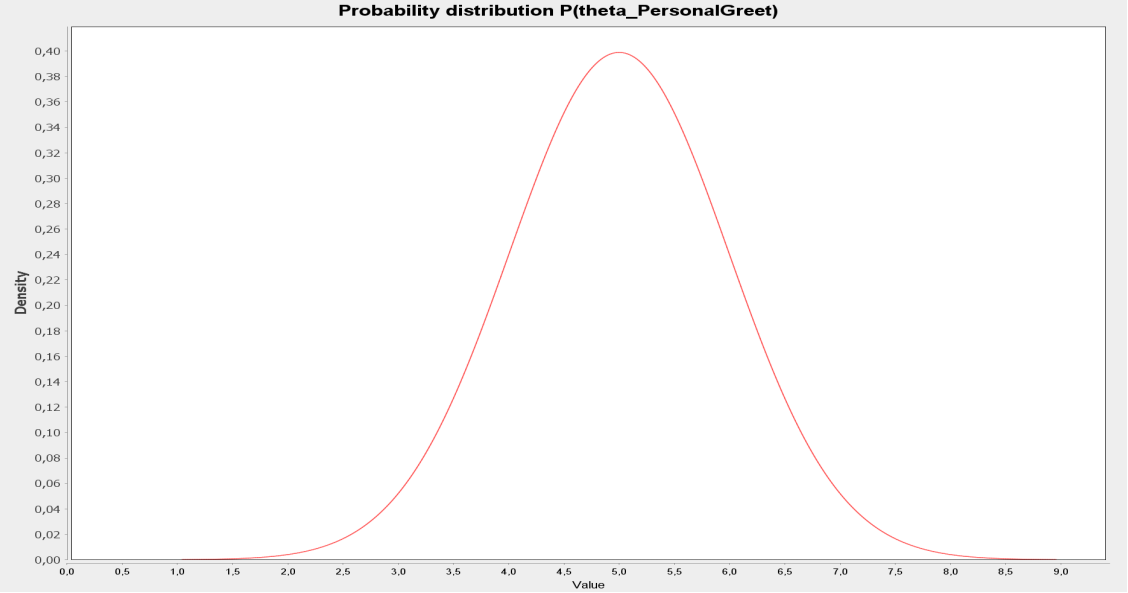

Figure 6.1.: Gaussian distribution with $\mu = 5$ and $\sigma^2 = 1$ before the learning algorithm for all possible dialogue actions

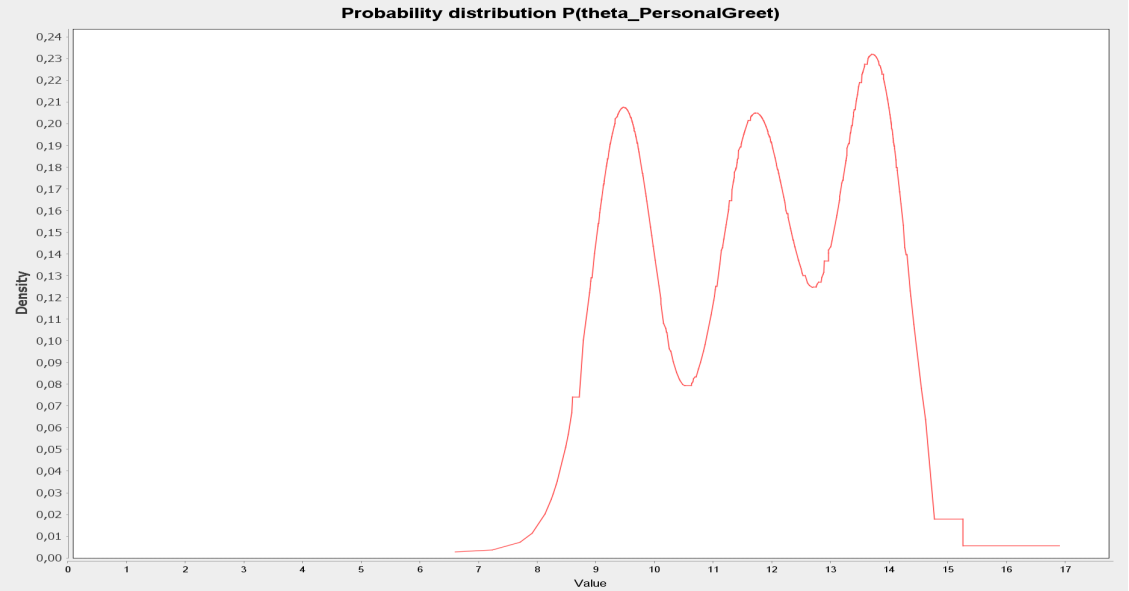

Figure 6.2.: *Kernel density estimator* after the learning algorithm for the dialogue action 'PersonalGreet' with high occurrence

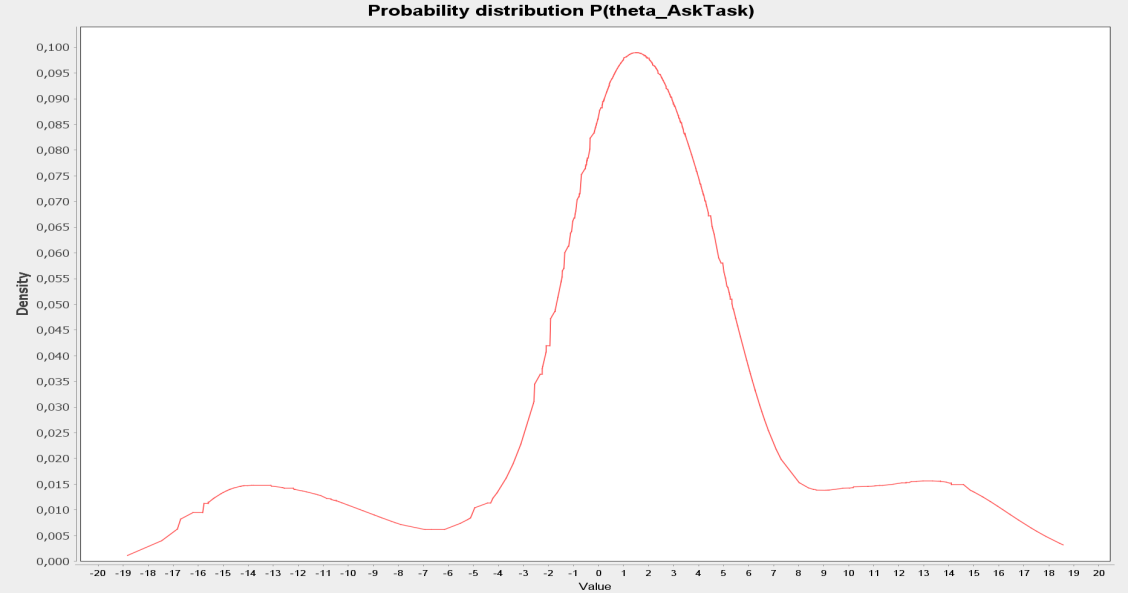

Figure 6.3.: *Kernel density estimator* after the learning algorithm for the dialogue action 'AskTask' with low occurrence

It can be observed that the mean values of the distributions are shifted by the learning algorithm according to the number of occurrences of the dialogue actions. A frequent selection leads to a higher mean value of the parameter density which thus increases the value of the corresponding utility. The variance of the distribution is adjusted in a similar way; but in contrast to the mean value, the variance has only a minor influence on the action selection. The figures use intentionally different axis scales in order to enable a central depiction of the density functions.

The next experiment has been to export the parameters of a trained policy and integrate them into the domain file as a prior distribution. In this way, the estimated parameters would be retained if the dialogue manager was restarted. Since it is not possible to export a *Kernel density estimator* in OpenDial, this type is converted into a Gaussian distribution. However, the major drawback is that probabilistic characteristics are lost during this process. The converted distribution is rarely comparable to the original and leads to a completely different behaviour of the dialogue management. For this reason, the approach of exporting the parameters was discarded.

The prior distribution provides only part of the information necessary for the estimation process, the geometric factor also contributes to the procedure and is therefore examined in the following section.

6.2. Geometric factor

The geometric factor expresses how relevant the wizard's actions are for the training algorithm and has huge influence on the learned dialogue policy as can be seen in Tables 6.1,6.2, and 6.3. As in the previous section, the German training data set is used since this allows concentrating on the algorithm and not on cultural differences. For the tests, the dialogue policy was trained with three different values of the geometric factor. Afterwards, the occurrence of dialogue actions was evaluated for a total of 1000 trials. In order to obtain a more reliable result, each test was performed three times. In the first table, the geometric factor is valued high with a probability p=0.7. Hence, although the dialogue action 'SimpleGreet' occurs only once out of a total of 77 interactions, it is granted absolute priority since it has the last position in the training file. This ratio is totally inconsistent with the database and therefore not desired. The second table shows the results for p=0.5. Here, the ratio of the outcomes does not correspond to the training data as well. The position in the training file still has influence. Nevertheless, at least a slight variation of the dialogue actions is observable even though the occurrence of dialogue actions can be extreme. The complete lack of variability consequently invalidates this configuration. Finally, the geometric factor p=0.2 in the third table achieves a policy which strictly selects relevant dialogue actions of the database. As typical for statistical methods, the estimation of the parameters is different for each training run and the number of possible choices varies. The probability of selecting a dialogue action with few occurrences converges to 0, and the position in the training file is no longer considered. A still existing weakness is that a result is occasionally not variable, this drawback could not be compensated by a further reduction of the geometric factor.

Dialogue action	Run1	Run2	Run3
SimpleGreet	1000	1000	1000
PersonalGreet	0	0	0
PersonalGreet+AskMood	0	0	0
PersonalGreet+AskTask	0	0	0

Table 6.1.: Action selection of the dialogue manager for a total of 1000 trials after the learning algorithm with geometric factor p=0.7

Dialogue action	Run1	Run2	Run3
SimpleGreet	43	0	326
PersonalGreet	872	1000	674
PersonalGreet+AskMood	0	0	0
PersonalGreet+AskTask	85	0	0

Table 6.2.: Action selection of the dialogue manager for a total of 1000 trials after the learning algorithm with geometric factor p=0.5

Dialogue action	Run1	Run2	Run3
SimpleGreet	0	0	0
PersonalGreet	752	627	852
PersonalGreet+AskMood	86	120	0
PersonalGreet+AskTask	162	253	148

Table 6.3.: Action selection of the dialogue manager for a total of 1000 trials after the learning algorithm with geometric factor p=0.2

As a consequence, it was assumed that the amount of training data is not sufficient to ensure variability. Accordingly, the learning algorithm was repeated several times with the same training set. The more repetitions of the learning algorithm, the fewer actions were possible until only the dialogue action with the most occurrences in the database was selected. Since this is neither an imitation of the wizard behaviour, the reuse of training data was discarded.

After determining the geometric factor and the characteristics of the prior distribution, the most suitable setting for the learning algorithm could finally be specified. On this basis, the differences between the cultures German and Polish are presented in the next section.

6.3. Influence of culture

The adaptation of the dialogue manager to the culture of a user was achieved by learning the policy for different cultures separately. So far, the experiments used training data of German dialogues. Although the available training data set in Polish was rather small, the results revealed large differences between both cultures. The in Table 6.4 shown experiment correlates to Table 6.3. Thereby, the Polish culture prefers single dialogue actions as response. Additionally, the action selection is limited to two possibilities: 'SimpleGreet' and 'PersonalGreet'. In contrast to the German dialogues, 'SimpleGreet' is still selected if the training algorithm employs a low geometric factor. Furthermore, the ratio of dialogue actions varies. Both cultures have in common that 'PersonalGreet' is the most popular dialogue action.

Dialogue action	Run1	Run2	Run3
SimpleGreet	59	23	70
PersonalGreet	941	977	930
PersonalGreet+AskMood	0	0	0
PersonalGreet+AskTask	0	0	0

Table 6.4.: Action selection of the dialogue manager for a total of 1000 trials after the learning algorithm with geometric factor p=0.2, trained with Polish interactions

By modelling the culture-dependent behaviour in the policy, the dialogue manager acts finally user-adaptive. However, one reason for the differences may be the small Polish training corpus. The following section outlines the extension of the tests to the entire domain.

6.4. Entire domain

After all experiments in the subdomain were completed, the entire domain was implemented. For this, the structure of the 'Greet' rule and its training file has been retained. In addition, the rules for emotions and information from the knowledge integration module were included. As the learning algorithm processes the estimation of the parameters independently for each rule and the conditions of the rules are obeyed; hence, the dialogue management is able to operate in the entire domain.

Since every conversation in the KRISTINA domain has its own course without the necessity of a task, the dialogue manager depends absolutely on data from the linguistic analysis and the knowledge integration. However, there were only example data sets available for this thesis. Due to this, a quantitative evaluation such as for the subdomain is not possible. Instead, the mean values of the parameter distributions are compared for both cultures. A list of the three most important parameters for the most relevant rules of the domain is provided in Table 6.5. The prior distribution was a Gaussian distribution with a mean value of 5 and a variance of 1. The geometric factor was set to p=0.2.

Rule name	Parameter	German	Polish
'Accept'	$\theta_{Acknowledge+SimpleThank}$	4.78	4.99
	$\theta_{Declare}$	5.42	5.06
	$\theta_{ShareJoy}$	5.20	3.42
'Acknowledge'	$\theta_{Acknowledge}$	5.63	5.10
	$\theta_{Acknowledge+PersonalThank}$	3.12	5.01
	$\theta_{AskTask}$	4.78	3.35
'AskMood'	$\theta_{Accept+SimpleThank}$	4.95	3.25
	$\theta_{Acknowledge+SimpleThank}$	4.93	3.62
	$\theta_{SimpleThank}$	5.26	3.45
'Declare'	$\theta_{Acknowledge}$	5.23	4.95
	$\theta_{Acknowledge+RequestMissingInformation}$	5.14	4.73
	$\theta_{RequestMissingInformation}$	4.86	5.56
'Greet'	$\theta_{SimpleGreet}$	3.63	4.97
	$\theta_{PersonalGreet}$	5.31	5.35
	$\theta_{PersonalGreet+AskTask}$	5.07	3.21
'Thank'	$\theta_{AnswerThank}$	5.13	5.07
	$\theta_{AnswerThank+SimpleSayGoodbye}$	4.59	4.96
	$\theta_{PersonalAnswerThank}$	5.22	3.87
'Reject'	$\theta_{Acknowledge}$	4.99	5.06
	$\theta_{Declare}$	3.92	4.76
	$\theta_{RequestAdditionalInformation}$	5.48	4.86
'Request'	θ_{Accept}	4.97	4.90
	$\theta_{Accept+Declare}$	5.93	4.38
	$\theta_{Declare}$	5.76	8.14
'SayGoodbye'	$\theta_{MeetAgainSayGoodbye}$	4.23	3.56
	$\theta_{PersonalSayGoodbye}$	5.45	3.68
	$\theta_{SimpleSayGoodbye}$	6.21	3.51

Table 6.5.: Mean values of the rule parameters for the cultures German and Polish after the learning algorithm

In this table, the depicted values are the averages of three training runs. To allow a statistical comparison, both cultures were trained with the same amount of dialogues. In contrast to Section 6.3, also the Polish interactions which begin without a greeting could be used. This resulted in a total number of 16 Polish dialogues; consequently, the same number of German dialogues were selected. It was ensured that the dialogues had a similar topic of conversation. Since a shift of the mean value of parameter distributions also means a shift of the corresponding utility, this table reveals which dialogue actions are preferred by a culture in a specific dialogue state. The most significant differences and similarities are discussed in the following.

For the 'Accept' rule, it is likely that both cultures will respond with the same dialogue action 'Declare'. However, the dialogue action 'ShareJoy' is never used in Polish interactions and is therefore rated poorly, but it is quite common in German conversations. Although the dialogue action 'Acknowledge' has the highest priority for Polish and German in the 'Acknowledge' rule, the second-highest differs. It can be observed here that consecutive dialogue actions are also selected in Polish. Furthermore, the values for the actions are more extreme in German. A special case is the 'AskMood' rule since there are no Polish training data for this. As a result, all available dialogue actions have a small utility for this culture. In the 'Declare' rule, the contrast between cultures can be recognized clearly. In addition to the mere difference of the most relevant dialogue action, the Polish dialogue manager has only one the favourite 'RequestMissingInformation' while the German dialogue manager selects between two possibilities: 'Acknowledge' and 'Acknowledge+RequestMissingInformation'. Since the 'Greet' rule has already been described in detail in the previous section, it is not explained here. An example for a conceptual mistake is observable in the 'Thank' rule. Since the database consists various interactions, the dialogue actions 'AnswerThank+SimpleSayGoodbye' can occasionally be selected by both cultures. However, this is only valid at the end of a conversation. This underlines the need for a dialogue history which enables the correct mapping of the dialogue state. A system designer has to be aware of this limitation and therefore has to verify the training data. In the 'Reject' rule, the most relevant dialogue action is different for both cultures. Nevertheless, the Polish mean values are quite close to each other which implies a high variability. As can be seen in the 'Request' rule, the Polish culture has a clear preference 'Declare' for the action selection. Such a high mean value excludes the occurrence of further dialogue actions. In contrast, two consecutive dialogue actions are favoured by the German culture: 'Accept+Declare' and 'Declare'. Finally, the 'SayGoodbye' rule reveals that none of the Polish conversations contained a farewell. For the German culture, the 'SimpleSayGoodbye' has the highest priority, unlike to the preference of a personal greeting.

After the completion of this test, the experiments of this thesis were finished.

7. Conclusion

In this thesis, the dialogue management within the KRISTINA domain has been examined. Since neither rule-based nor statistical approaches are applicable in this context, probabilistic rules have been employed which were subsequently trained taking into account the user's cultural background.

First, the toolkit OpenDial has been integrated into the already existing dialogue manager OwlSpeak. However, this caused a malfunction of the GUI of OpenDial as soon as a training procedure was started. Therefore, a new method for controlling the toolkit has been developed that attaches data to an inquiry to the OwlSpeak Servlet.

The most important task of the dialogue manager is probably the selection of the next action. In OpenDial, a model in the domain file determines the policy. The basic structure of this model was obtained from a database of annotated interactions in the KRISTINA domain. In this process, it was ensured that the result is independent of the culture. Moreover, the probabilistic rules have been designed to support up to two consecutive system actions.In addition to the action selection model, a model for emotional actions was integrated into the domain. However, the database did not provide a proper basis for this because of the unreliability of the ratings. As a result, a learning algorithm could not be applied due to the lack of training data. The probabilistic rules of this model therefore had to be associated with fixed values instead of unknown parameters. Nevertheless, since this model requires a precondition, the mapping to fixed values is limited to particular cases. During the design of the models, the two types of rules were investigated as well. Since the number of occurrences of the dialogue actions in the database corresponds to a probability, it was initially intended to employ probability rules. However, the learning algorithm is not capable of processing probabilities as its likelihood calculation bases on utilities. Therefore, the rules in the action selection model were implemented as utility rules.

Subsequently, the learning algorithm for the estimation of the posterior probability distribution of the parameters has been implemented. The method of reinforcement learning was discarded as no reward model could be determined due to the requirements of the KRISTINA domain. Hence, the technique of Wizard-of-Oz learning was applied. As a connection between two remote computers did not work correctly in this thesis, the training procedure was performed using pre-scripted interaction files. Comparable to the domain model, the annotation database provides the basis for these files. However, since the cultural background has a significant influence on the conversational behaviour, the training sets were created separately for the cultures. In this work, two languages have been covered: German and Polish.

After the implementation of the learning algorithm, it was tested which effects are caused by varying the prior distribution of the parameters and the geometric factor of the

learning algorithm. Therefore, a subdomain was observed to allow a better monitoring. For the prior distribution, the influence of the family of probability distribution was negligible. That is plausible since the type only affects the first learning step and is then substituted by a *Kernel density estimator*. However, the experiments showed that the mean of the distributions can be chosen arbitrarily if it is greater than 0. For the geometric factor, the tests indicate the best value between 0.1 and 0.25. For this, the ratio of occurrence of the dialogue actions in the interaction files is of interest. The more frequent one dialogue action occurs, the less the geometric factor is to choose. Otherwise, always the same dialogue action is selected by the dialogue manager. Next, experiments on the influence of the culture on the dialogue policy have been conducted. It was again limited to a subdomain in order to allow a quantitative comparison. Thereby, the differences between German and Polish were clearly visible in the results. Afterwards, the experiments have been repeated for the entire domain. Since a quantitative comparison is not meaningful here, only the functional efficiency has been ensured.

In summary, this thesis has successfully enabled the dialogue management to adapt to the emotions and the culture of a user in the KRISTINA domain. However, in the course of this thesis, many ideas and concepts had to be abandoned due to restrictions within the OpenDial toolkit. The superficial and complex documentation required a lot of time for the familiarization with this toolkit. In addition, some functionalities did not work the way we expected what required time-consuming modifications.

The major drawback of the supervised approach is the limitation of the available training data. Since even the German data set with 77 interactions is not sufficient for a consistent result, the Polish data set with only 12 interactions lacks the statistical reliability completely. For future research, the concept of reinforcement learning for the posterior distribution of the parameters is proposed. This method is much more versatile than the Wizard-of-Oz technique, but could not be further investigated within this work. Since the toolkit OpenDial provides model-based and model-free algorithms, different reward models are conceivable. Furthermore, training data for the emotional model should be acquired. The rules associated with fixed values are not suitable for imitating natural behaviour. Another promising research topic is the use of verbosity and directness of dialogue actions in the dialogue management. This is also intended to cause an adaptation to the culture of a user. Finally, the experiments should be conducted with real users to reveal new information on the proper behaviour of the system.

A. Appendix

Complete list of dialogue actions in the KRISTINA domain:

1. Statement
2. Declare
3. RequestMissingInformation
4. AdditionalInformationRequest
5. RequestClarification
6. ExplicitlyConfirmRecognisedInput
7. ImplicitlyConfirmRecognisedInput
8. RequestRepeat
9. RequestRephrase
10. StateMissingComprehension
11. Affirm
12. Accept
13. Reject
14. Acknowledge
15. Advise
16. Obligate
17. Order
18. ShowWebpage
19. ShowVideo
20. ReadNewspaper
21. Greet

22. PersonalGreet
23. SimpleGreet
24. MorningGreet
25. AfternoonGreet
26. EveningGreet
27. CalmDown
28. CheerUp
29. Console
30. ShareJoy
31. AdressEmotion
32. RequestReasonForEmotion
33. SayGoodbye
34. MeetAgainSayGoodbye
35. PersonalSayGoodbye
36. SimpleSayGoodbye
37. MorningSayGoodbye
38. AfternoonSayGoodbye
39. EveningSayGoodbye
40. WeekendSayGoodbye
41. Thank
42. PersonalThank
43. SimpleThank
44. AnswerThank
45. PersonalAnswerThank
46. Request
47. ShowWeather
48. Canned

49. SimpleApologise
50. PersonalApologise
51. AskMood
52. AskTask
53. AskPlans
54. AskTaskFollowUp
55. SimpleMotivate
56. PersonalMotivate
57. Empty
58. Incomprehensible
59. Unknown
60. RequestFurtherInformation
61. BooleanRequest
62. IRResponse
63. RequestFeedback
64. UnknownStatement
65. UnknownRequest
66. AdditionalInformation
67. RequestSpecifyingInformation
68. ProactiveResponse

```
SELECT b.DialogueAction AS FirstSystemMove,
       a.DialogueAction AS SecondSystemMove,
       COUNT(b.DialogueAction) AS NumberOfOccurrences
FROM Annotation AS a
JOIN Annotation AS b
ON a.DialogueID = b.DialogueID
AND a.DialogueActionNR = b.DialogueActionNR + 1
JOIN Annotation AS c
ON a.DialogueID = c.DialogueID
AND a.DialogueActionNR =c.DialogueActionNR + 2
WHERE a.Participant = 'System' AND b.Participant = 'System'
AND c.Participant = 'User' AND
((((c.DialogueAction = 'PersonalGreet'
OR c.DialogueAction = 'SimpleGreet')
OR c.DialogueAction = 'EveningGreet')
OR c.DialogueAction = 'MorningGreet')
OR c.DialogueAction = 'AfternoonGreet')
GROUP BY b.DialogueAction,a.DialogueAction
```

Figure A.1.: SQL-Query of the 'Greet' rule for two consecutive dialogue actions

```
SELECT b.DialogueAction AS SystemMove,
       COUNT(b.DialogueAction) AS NumberOfOccurrences
FROM Annotation AS a
JOIN Annotation AS b
ON a.DialogueID = b.DialogueID
AND a.DialogueActionNR = b.DialogueActionNR + 1
JOIN Annotation AS c
ON a.DialogueID = c.DialogueID
AND a.DialogueActionNR =c.DialogueActionNR + 2
WHERE a.Participant = 'User' AND b.Participant = 'System'
AND c.Participant = 'User' AND
((((c.DialogueAction = 'PersonalGreet'
OR c.DialogueAction = 'SimpleGreet')
OR c.DialogueAction = 'EveningGreet')
OR c.DialogueAction = 'MorningGreet')
OR c.DialogueAction = 'AfternoonGreet')
GROUP BY b.DialogueAction
```

Figure A.2.: SQL-Query of the 'Greet' rule for one dialogue action

Bibliography

[1] Amazon Alexa. `https://alexa.amazon.com`. Retrieved: 25.10.2017.

[2] Apple Siri. `https://www.apple.com/de/ios/siri/`. Retrieved: 25.10.2017.

[3] KRISTINA project. `http://kristina-project.eu`. Retrieved: 05.09.2017.

[4] Meaning of probability notations $p(z; d, w)$ and $p(z|d, w)$. Cross Validated, `https://stats.stackexchange.com/q/10234`. Version: 26.04.2014 , Retrieved: 16.11.2017.

[5] Mysql 5.5 reference manual. Oracle Corporation, `https://dev.mysql.com/doc/refman/5.5/en/`. Revision: 54876, Retrieved: 20.11.2017.

[6] Elisabeth André. Preparing emotional agents for intercultural communication. *The Oxford Handbook of Affective Computing*, page 309, 2015.

[7] Soon Ang and Linn Van Dyne. *Handbook of cultural intelligence*. Routledge, 2015.

[8] Irad Ben-Gal, F Ruggeri, F Faltin, and R Kenett. Bayesian networks, encyclopedia of statistics in quality and reliability, 2007.

[9] Gregor Bertrand, Florian Nothdurft, Wolfgang Minker, Harald Traue, and Steffen Walter. Adapting dialogue to user emotion-a wizard-of-oz study for adaptation strategies. In *Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop*, pages 285–294. Springer, 2011.

[10] Clayton Blehm, Seema Vishnu, Ashbala Khattak, Shrabanee Mitra, and Richard W. Yee. Computer vision syndrome: A review. *Survey of Ophthalmology*, 50(3):253 – 262, 2005.

[11] Kevin K. Bowden, Shereen Oraby, Amita Misra, JiaQi Wu, and Stephanie M. Lukin. Data-driven dialogue systems for social agents. *CoRR*, abs/1709.03190, 2017.

[12] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and Franois Yergeau. Extensible markup language (xml) 1.0 (fifth edition). *W3 Consortium*, 2008.

[13] Daniel D. Corkill. Blackboard systems. *AI Expert*, 6:40–47, 1991.

[14] Ramon Lopez Cozar Delgado and Masahiro Araki. *Spoken, multilingual and multimodal dialogue systems: development and assessment*. John Wiley & Sons, 2007.

[15] Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

[16] Robert Fung and Kuo-Chu Chang. Weighing and integrating evidence for stochastic simulation in bayesian networks. *arXiv preprint arXiv:1304.1504*, 2013.

[17] Milica Gašić, Steve Young, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[18] Milan Gnjatović and Dietmar Rösner. The nimitek corpus of affected behavior in human-machine interaction. In *Proceedings of the Second International Workshop on Corpora for Research on Emotion and Affect (satellite of LREC'08)*, pages 5–8, Marrakech, Morocco, 2008. European Language Resources Association (ELRA), ISBN: 2-9517408-4-0.

[19] Michael Gordon. Community care for the elderly: is it really better? *CMAJ: Canadian Medical Association Journal*, 148(3):393, 1993.

[20] Paul Green and Lisa Wei-Haas. The rapid development of user interfaces: Experience with the wizard of oz method. *Proceedings of the Human Factors Society Annual Meeting*, 29(5):470–474, 1985.

[21] David Griol, Zoraida Callejas, Ramón López-Cózar, and Giuseppe Riccardi. A domain-independent statistical methodology for dialog management in spoken dialog systems. *Computer Speech & Language*, 28(3):743–768, 2014.

[22] T. Heinroth, D. Denich, and A. Schmitt. Owlspeak - adaptive spoken dialogue within intelligent environments. In *2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, pages 666–671, March 2010.

[23] Tobias Heinroth and Wolfgang Minker. *Introducing spoken dialogue systems into Intelligent Environments*. Springer Science & Business Media, 2012.

[24] David O Johnson, Raymond H Cuijpers, James F Juola, Elena Torta, Mikhail Simonov, Antonella Frisiello, Marco Bazzani, Wenjie Yan, Cornelius Weber, Stefan Wermter, et al. Socially assistive robots: a comprehensive approach to extending independent living. *International journal of social robotics*, 6(2):195–211, 2014.

[25] Kristiina Jokinen and Michael F. McTear. *Spoken Dialogue Systems*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2009.

[26] Arne Jönsson and Nils Dahlbäck. *Talking to a computer is not like talking to your best friend*. Universitetet i Linköping/Tekniska Högskolan i Linköping. Institutionen för Datavetenskap, 1988.

[27] Staffan Larsson and David R. Traum. Information state and dialogue management in the trindi dialogue move engine toolkit. *Natural Language Engineering*, 6(3-4):323340, 2000.

[28] P. Lison and C. Kennington. Opendial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Demonstrations)*, pages 67–72, Berlin, Germany, 2016. Association for Computational Linguistics.

[29] Pierre Lison. *Structured Probabilistic Modelling for Dialogue Management.* dissertation, University of Oslo, 2013.

[30] Diane J. Litman and Shimei Pan. Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction*, 12, 2002.

[31] Patrick Lutz. Integrating the OpenDial Toolkit into OwlSpeak. unpublished.

[32] Norashikin Mahmud, Siti Fatimah Bahari, and Nurul Farha Zainudin. Psychosocial and ergonomics risk factors related to neck, shoulder and back complaints among malaysia office workers. *International Journal of Social Science and Humanity*, 4(4):260, 2014.

[33] Michael F. McTear. Spoken dialogue technology: Enabling the conversational user interface. *ACM Comput. Surv.*, 34(1):90–169, March 2002.

[34] Juliana Miehle. *IQ-adaptive statistical dialogue management using Gaussian processes.* 2017.

[35] Roger K Moore. Is spoken language all-or-nothing? implications for future speech-based human-machine interaction. In *Dialogues with Social Robots*, pages 281–291. Springer, 2017.

[36] Anthony O'Hagan, Caitlin E Buck, Alireza Daneshkhah, J Richard Eiser, Paul H Garthwaite, David J Jenkinson, Jeremy E Oakley, and Tim Rakow. *Uncertain judgements: eliciting experts' probabilities.* John Wiley & Sons, 2006.

[37] World Health Organization. *World Report on Ageing and Health.* Nonserial Publication. World Health Organization, 2015.

[38] Joelle Pineau, Michael Montemerlo, Martha Pollack, Nicholas Roy, and Sebastian Thrun. Towards robotic assistants in nursing homes: Challenges and results. *Robotics and autonomous systems*, 42(3):271–281, 2003.

[39] Johannes Pittermann, Angela Pittermann, and Wolfgang Minker. Emotion recognition and adaptation in spoken dialogue systems. *International Journal of Speech Technology*, 13(1):49–60, 2010.

[40] François Portet, Michel Vacher, Caroline Golanski, Camille Roux, and Brigitte Meillon. Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects. *Personal and Ubiquitous Computing*, 17(1):127–144, Jan 2013.

[41] Mike Potel. Mvp: Model-view-presenter - the taligent programming model for c++ and java. 1996.

[42] Katharina Reinecke and Abraham Bernstein. Knowing what a user likes: A design science approach to interfaces that automatically adapt to culture. *Mis Quarterly*, 37(2), 2013.

[43] Laurel D Riek. Wizard of oz studies in hri: a systematic review and new reporting guidelines. *Journal of Human-Robot Interaction*, 1(1), 2012.

[44] Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The Knowledge Engineering Review*, 21(2):97126, 2006.

[45] John Ferguson Smart et al. An introduction to maven 2. *JavaWorld Magazine. Available at: http://www. javaworld. com/javaworld/jw-12-2005/jw-1205-maven. html*, 2005.

[46] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[47] Stefan Ultes and Wolfgang Minker. Managing adaptive spoken dialogue for intelligent environments. In *Journal of Ambient Intelligence and Smart Environments*, volume 9, pages 523–539, 2014.

[48] Stefan Ultes, Lina M. Rojas Barahona, Pei-Hao Su, David Vandyke, Dongho Kim, Iñigo Casanueva, Paweł Budzianowski, Nikola Mrkšić, Tsung-Hsien Wen, Milica Gasic, and Steve Young. PyDial: A Multi-domain Statistical Dialogue System Toolkit. In *Proceedings of ACL 2017, System Demonstrations*, pages 73–78, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[49] Nikos Vlassis, Mohammad Ghavamzadeh, Shie Mannor, and Pascal Poupart. Bayesian reinforcement learning. *Reinforcement Learning: State-of-the-Art*, 12:359–386, 2012.

[50] Christian Walck. Hand-book on statistical distributions for experimentalists. Technical report, 1996.

[51] Jason Williams, Antoine Raux, and Matthew Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33, 2016.