An Improved Approach of Intention Discovery with Machine Learning for POMDP-based Dialogue Management

Ruturaj Rajendrakumar Raval

University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
https://scholar.uwindsor.ca/etd/7731

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000 ext. 3208.
An Improved Approach of Intention Discovery with Machine Learning for POMDP-based Dialogue Management

By

RUTURAJ Rajendra Kumar RAVAL

A THESIS

Submitted to the Faculty of Graduate Studies
Through Computer Science
In Partial Fulfilment of the Requirements for
The Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada

2019

© 2019 RUTURAJ RAJENDRAKUMAR RAVAL
An Improved Approach of Intention Discovery with Machine Learning for POMDP-based Dialogue Management

By

RUTURAJ Rajendrakumar RAVAL

Approved by:

____________________________________
G. Bhandari
Odette School of Business

____________________________________
L. Rueda
School of Computer Science

____________________________________
X. Yuan, Advisor
School of Computer Science

April 25, 2019
Declaration of originality

I hereby certify that, I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

Some of the material in this thesis is inspired and taken into work as a part of the advanced work of their work. Their work can be found in the thesis 1 [1] and thesis 2 [2]; and thesis 3 [3] for setting a good benchmark, how to write a thesis and I have taken a good amount of inspiration.

I certify that, with the above qualification, this thesis, and the research to which it refers, is the product of my own work. I certify that, to the best of my knowledge, my thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the graduate studies office and that this thesis has not been submitted for a higher degree of any other University or Institution.
Abstract

An Embodied Conversational Agent (ECA) is an intelligent agent that works as the front end of software applications to interact with users through verbal/nonverbal expressions and to provide online assistance without the limits of time, location, and language. To help improve the experience of human-computer interaction, there is an increasing need to empower ECA with not only the realistic look of its human counterparts but also a higher level of intelligence. This thesis first highlights the main topics related to the construction of ECA, including different approaches of dialogue management, and then discusses existing techniques of trend analysis for its application in user classification. As a further refinement and enhancement to prior work on ECA, this thesis research proposes a cohesive framework to integrate emotion-based facial animation with improved intention discovery. In addition, a machine learning technique is introduced to support sentiment analysis for the adjustment of policy design in POMDP-based dialogue management. It is anticipated that the proposed research work is going to improve the accuracy of intention discovery while reducing the length of dialogues.

Keywords: Human-Computer Interaction, Q-Learning, POMDP, Sentiment Analysis, Reinforcement Learning, Machine Learning, Artificial Intelligence, 3D model, ECA, Decision-making Process, Interaction
Dedication

I dedicate my work to my brother Mr. Miraj, my Parents Mrs. Manisha and Mr. Rajendra and my beloved Family, Maternal Grandparents Mrs. Rekha and Mr. Subhashchandra; Maternal aunt-uncle Mrs. Nimisha and Mr. Mayur; Sisters Ms. Kalapini, Miss Naitri and Miss Krupali, for their continuous support and love. I am grateful for their motivation and faith in me which boosted my confidence to work hard and I thank them for believing in me and being there for me during my struggle; for feeding me great and delicious food; for serving me with the best facility possible even with the least available resources. I love you all from the bottom of my heart. I seek blessings of my ancestor Paternal Grandparents Late Mrs. Hemlata and Late Mr. Jashvantlal, I know we haven’t met but I firmly believe that you have blessed me and our family (my brother and my parents) a lot from heaven!

I thank almighty God, Lord Ganesha, Goddess Mahalaxmi, Goddess Saraswathy, Goddess Amba, Lord Sun, Lord Vishnu, Lord Shiva, and my GURU, for protecting me always and hearing me out all the time. Thank you for hearing my confessions and making me (the devil child of yours) a good person (I guess!) over the years. I thank your greatness in blessing me and my family.
Acknowledgement

I would like to grab the opportunity to thank my supervisor, Dr. Xiaobu Yuan for his constant encouragement and motivation and support since the beginning. I feel so grateful that he provided his immense support by contributing his suggestions and ideas during my research. His vast knowledge and insightful feedback have helped in a great way always.

I would like to acknowledge the constructive feedback from Dr. Luis Rueda and Dr. Gokul Bhandari, which helped me in modifying my work to make it better.

I take this platform to thank the Ex. Graduate Secretary of the University of Windsor, School of Computer Science, Ms. Karen Bourdeau for her constant support when I was stuck, and I needed a way out of it, and I thank her for standing by my side and motivating me and I thank her for believing in me.

I want to acknowledge my family for their mental; financial support throughout in pursuing my dream and my parents for allowing me to mortgage their house to get me the bank educational loan.

I want to extend my sincere thanks to my best friend Pinkal Patel for helping with her IELTS material so that I could prepare for the exam in less than 2 weeks. I want to thank Tristan Szucs, my fellow batchmate for helping me proofread this thesis document for me and helping me out with his valuable opinion.
Table of Contents

Declaration of originality................................................................. iii

Abstract........................................................................................................ iv

Dedication....................................................................................................... v

Acknowledgement ........................................................................................ vi

Table of Tables .......................................................................................... xii

Table of Figures ............................................................................................ xiii

Table of Equations ........................................................................................ xvii

Table of Nomenclatures ................................................................................ xix

Table of Appendices ...................................................................................... xx

Abbreviations/Symbols.................................................................................. xxi

Chapter 1: Introduction .................................................................................. 1
  1.1 Motivation .............................................................................................. 1
  1.2 Introduction to Human-Computer Interaction (HCI) ................................ 1
  1.3 Introduction to Embodied Conversational Agent (ECA) ........................... 4
  1.4 Problem statement ............................................................................... 6

Chapter 2: A Literature review ..................................................................... 8
  2.1 Embodied Conversational Agent (ECA) with Dialogue Management (DM) .... 9
  2.2 Dialogue management ......................................................................... 11
    2.2.1 Dialogue management using Genetic Algorithm ....................... 13
    2.2.2 Uncertainty estimates in deep reinforcement learning ............ 14
2.2.3 Multi-party dialogue with ECA ................................................................. 16

2.3 Markov Decision Process (MDP) ........................................................................ 16

2.3.1 Shortcomings of MDPs .................................................................................. 18

2.4 Partially Observable Markov Decision Process (POMDP).................................. 19

2.5 Dialogue Management (DM) in different domains ............................................. 21

2.5.1 Dialogue Management (DM) with POMDP ................................................... 21

2.5.2 Dialogue Management with Reinforcement Learning .................................... 21

2.5.3 Dialogue management (DM) using Long Short-Term Memory (LSTM)-based natural language generation ................................................................. 25

2.5.4 Dialogue Management (DM) approaches for belief tracking ...................... 25

2.5.5 Dialogue Management (DM) based on Ontology ......................................... 29

2.5.6 Other models in Dialogue Management (DM) models .................................. 30

Chapter 3: Detailed Partially Observable Markov Decision Process model ............... 31

3.1 Detailed POMDP in different approaches ......................................................... 31

3.1.1 Belief State (BS) .......................................................................................... 39

3.1.2 Policy ............................................................................................................ 43

3.1.3 Shortcomings of POMDP .............................................................................. 45

3.2 Previous works ................................................................................................... 46

3.2.1 Emotion detection from speech .................................................................... 46

3.2.2 The Contextual Control Model (COCOM) .................................................. 47

3.3 Belief state History and Trend Analysis ............................................................. 49

3.3.1 POMDP-based Dialogue management ......................................................... 49
Chapter 3: Previous approaches (Significant Problems)

3.1 Overview

3.2 Trend analysis using Belief state

3.3 POMDP using Belief state History information

3.3.1 POMDP in a programming language

3.4 Wavelet Theory

3.4.1 Sharp Variation points

3.5 Fuzzy Logic

3.6 Shortcomings of previous approaches (Significant Problems)

3.7 The new approach – Thesis contribution

3.8 Thesis statement

3.9 Structure of the thesis report

Chapter 4: Proposed method

4.1 Overview

4.2 Sentiment Analysis

4.3 Reinforcement Learning

4.3.1 Q-Learning

4.3.2 Q-Learning Algorithm

4.4 POMDP with updated BS history to improve POLICIES

4.5 System Architecture

4.6 Discussion- Comparison and Contribution

4.7 Algorithm

4.8 Detailed Algorithm
Appendix D: Textual and Speech-based emotions discovery from a user’s interaction... 118

Appendix E: Android Studio setup ................................................................. 121

Appendix F: Domain specific ECA customization using Facegen Software with Brand Ambassador’s Alter Ego ........................................................................ 122

Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal........................................................................................................ 123

Appendix H: User-Agent chat conversation: Sentiment Analysis rewards .................. 126

Appendix I: Belief State History (BSH) database information for different users ....... 128

Appendix J: Q-Learning: Q-values Rewards data.................................................... 130

Appendix K: Policy improvement........................................................................... 131

Vita Auctoris........................................................................................................ 133
Table of Tables

Table 1 Confidence measure 1 ................................................................. 34
Table 2 Confidence measure 2 ................................................................. 34
Table 3 Confidence measure 3 ................................................................. 35
Table 4 Confidence measure 4 ................................................................. 35
Table 5 Algorithm for Q-Learning ............................................................ 66
Table 6 Different modules from the Thesis Architecture ............................ 67
Table 7 Discussion- Contribution and Comparison .................................... 70
Table 8 System Algorithm ........................................................................ 71
Table 9 Table of the detailed algorithm .................................................... 73
Table 10 Time Complexity of the algorithm .............................................. 74
Table 11 List of tools required to implement POMDP and emotion detection .................................................................................. 76
Table 12 Examples of Sentiment Analysis of different sentences ............... 86
Table 13 FAMM: Fuzzy Associative Memory Matrix table using COCOM ... 87
Table 14 Weights considered for FAMM table .......................................... 87
Table 15 Emotion decision taken based on COCOM table and weight table, to produce final emotion ................................................................. 88
Table 16 Final ECA emotion after crisp Fuzzy Logic System output .......... 88
Table 17 Results: Accuracy; Dialogue length; Neutral/Positive intention discovery ratio ................................................................. 92
Table 18 Final results discussion ................................................................ 92
Table 19 Chat conversation setup User-Agent .......................................... 125
Table 20 User-Agent interaction: Sentiment Analysis Rewards ................. 127
Table 21 BSH database data for different users ........................................... 129
Table 22 Rewards: Q-values data ............................................................. 130
Table 23 Policy table ................................................................................. 132
Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>The user interacts with the computer over this software interface using the given input and output (I/O) hardware.</td>
<td>3</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Modularized view of the interaction between the dialogue manager and the user in a dialogue management context.</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Dialogue Management (DM) as a Markov Decision Process (MDP).</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Dialogue Management (DM) as a Partially Observable Markov Decision Process (POMDP).</td>
<td>26</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Spoken Dialogue Systems (SDS).</td>
<td>26</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Dialogue Management (DM).</td>
<td>27</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Belief Tracking.</td>
<td>27</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Hidden Information State (HIS) dialogue states.</td>
<td>28</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Idealized HCI dialogue.</td>
<td>32</td>
</tr>
<tr>
<td>Figure 10</td>
<td>HCI decompose state variable model.</td>
<td>33</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Dialogue 1.</td>
<td>35</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Dialogue 2.</td>
<td>36</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Dialogue 3.</td>
<td>36</td>
</tr>
<tr>
<td>Figure 14</td>
<td>COCOM- Contextual Control Model.</td>
<td>48</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Flow chart for this POMDP-based DM.</td>
<td>50</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Architecture of POMDP-based on DM.</td>
<td>51</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Architecture diagram of the proposed framework using BSH information.</td>
<td>54</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Flow chart for every interaction between user and agent.</td>
<td>55</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Fuzzy Logic System.</td>
<td>58</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Fuzzy logic system based ECA expression based POMDP model.</td>
<td>59</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Reinforcement Learning flow diagram example.</td>
<td>63</td>
</tr>
</tbody>
</table>
Figure 22 Knowledge level selector architecture.............................................................67
Figure 23 Proposed POMDP-based Dialogue Management incorporating sentiment analysis and Reinforcement Learning architecture...............................................................69
Figure 24 Rewards chart: Chat conversation setup..........................................................79
Figure 25 3D model .........................................................................................................80
Figure 26 Ontology Chart ...............................................................................................82
Figure 27 Ontology chart - partial ...................................................................................83
Figure 28 Knowledge users: Sharp variation Points..........................................................85
Figure 29 Trend Analysis- Sharp Variation Points chart......................................................85
Figure 30 Sentiment Analysis Example data chart ............................................................86
Figure 31 Q-Values chart for all users .........................................................................89
Figure 32 Policy chart ....................................................................................................90
Figure 33 Accuracy calculation .......................................................................................91
Figure 34 A computer's Interface to a user ..................................................................110
Figure 35 A user's interface to a computer ..................................................................110
Figure 36 ECA smile .....................................................................................................115
Figure 37 ECA laugh ......................................................................................................115
Figure 38 ECA nose wrinkle ........................................................................................115
Figure 39 ECA lip close ................................................................................................115
Figure 40 ECA closed smile ........................................................................................115
Figure 41 ECA wide smile ...........................................................................................115
Figure 42 ECA small OH .............................................................................................115
Figure 43 ECA big OH ................................................................................................115
Figure 44 ECA smirk right ..........................................................................................115
Figure 45 ECA surprise ...............................................................................................115
Figure 46 ECA sneer right ................................................................. 115
Figure 47 ECA right smile ............................................................... 115
Figure 48 ECA funny smile ............................................................. 116
Figure 49 ECA sad .......................................................... 116
Figure 50 ECA puffed cheeks ................................................ 116
Figure 51 ECA mouth left .................................................. 116
Figure 52 ECA kiss .......................................................... 116
Figure 53 ECA jaw down .................................................. 116
Figure 54 ECA HMMM .................................................. 116
Figure 55 ECA frown .................................................. 116
Figure 56 ECA Fear ........................................................ 116
Figure 57 ECA disgust .......................................................... 116
Figure 58 ECA teeth barring ................................................ 116
Figure 59 ECA anger ........................................................ 116
Figure 60 ECA walleye .................................................. 117
Figure 61 ECA look up ................................................ 117
Figure 62 ECA look right ................................................ 117
Figure 63 ECA head up ................................................ 117
Figure 64 ECA head left ................................................ 117
Figure 65 ECA wink right eye ................................................ 117
Figure 66 ECA blink left the eye ................................................ 117
Figure 67 ECA suck .......................................................... 117
Figure 68 ECA mouth stretch ................................................ 117
Figure 69 ECA tongue out ................................................ 117
Figure 70 ECA depress .................................................. 117
Figure 71 ECA normal
........................................................................................................117
Figure 72 Splash/ welcome screen
........................................................................................................118
Figure 73 First half 3D model, 2nd half Dialogue Management manager
.............................................118
Figure 74 Edit Text to write a message, press blue button to send
..................................................118
Figure 75 DM module shows the interaction between user and agent- TEXT based
..........................................................118
Figure 76 Speech recognized interaction scenario by pressing the last button of recording audio
.........................................................................................................................................................119
Figure 77 2nd button in red, directly to recognize the emotion from the live recording audio
.........................................................................................................................................................119
Figure 78 SAD emotion detected from the user’s speech
.............................................................................................................119
Figure 79 ANGER emotion detected from the user’s speech
.............................................................................................................119
Figure 80 NEUTRAL emotion detected from the user’s speech
.............................................................................................................120
Figure 81 FEAR emotion detected from the user’s speech
.............................................................................................................120
Figure 82 Project structure 1
.......................................................................................................................................121
Figure 83 Project structure 2
.......................................................................................................................................121
Figure 84 Project structure 3
.......................................................................................................................................121
Table of Equations

<table>
<thead>
<tr>
<th>Equation</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Decision maker in an MDP to find Policy</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>The policy estimation using DM in RL</td>
<td>22</td>
</tr>
<tr>
<td>3</td>
<td>Estimation of unique optimal policy</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>Calculate Uncertainty through explicit state variables</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>Distribution maintained overstate as a cumulative confidence measure</td>
<td>35</td>
</tr>
<tr>
<td>6</td>
<td>Transition function</td>
<td>37</td>
</tr>
<tr>
<td>7</td>
<td>Transition probabilities</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>Observation function</td>
<td>38</td>
</tr>
<tr>
<td>9</td>
<td>Observation probabilities</td>
<td>38</td>
</tr>
<tr>
<td>10</td>
<td>Expected immediate reward</td>
<td>38</td>
</tr>
<tr>
<td>11</td>
<td>Lower and upper bound reward function</td>
<td>38</td>
</tr>
<tr>
<td>12</td>
<td>Belief update</td>
<td>39</td>
</tr>
<tr>
<td>13</td>
<td>Normalizing constant evaluation</td>
<td>39</td>
</tr>
<tr>
<td>14</td>
<td>Bayes' theorem</td>
<td>40</td>
</tr>
<tr>
<td>15</td>
<td>Dialogue history tracing</td>
<td>40</td>
</tr>
<tr>
<td>16</td>
<td>Belief distribution</td>
<td>40</td>
</tr>
<tr>
<td>17</td>
<td>Belief distribution re-written assuming with the Markov Property and Bayes' rule</td>
<td>41</td>
</tr>
<tr>
<td>18</td>
<td>Belief state</td>
<td>42</td>
</tr>
<tr>
<td>19</td>
<td>Element count of S</td>
<td>42</td>
</tr>
<tr>
<td>20</td>
<td>Normalizing constant</td>
<td>42</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>S</td>
</tr>
<tr>
<td>22</td>
<td>Policy function</td>
<td>43</td>
</tr>
<tr>
<td>23</td>
<td>Policy function</td>
<td>44</td>
</tr>
</tbody>
</table>
Equation 24 Optimal policy ................................................................. 44
Equation 25 Maximum expected cumulative reward .................................. 45
Equation 26 Row vector ......................................................................... 45
Equation 27 Modified planning strategy ...................................................... 49
Equation 28 Deep Reinforcement Learning .................................................. 64
Equation 29 Deep Q-Learning (Deep Reinforcement Learning) ....................... 65
Equation 30 Defuzzification to generate crisp output .................................... 88
## Table of Nomenclatures

<table>
<thead>
<tr>
<th>Definition</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belief State (BS)</td>
<td>15</td>
</tr>
<tr>
<td>Belief State HISTORY (BSH)</td>
<td>49</td>
</tr>
<tr>
<td>Continuous Wavelet Transform (CWT)</td>
<td>56</td>
</tr>
<tr>
<td>Discrete Wavelet Transform (DWT)</td>
<td>56</td>
</tr>
<tr>
<td>Embodied Conversational Agent (ECA)</td>
<td>4</td>
</tr>
<tr>
<td>Fuzzy logic</td>
<td>57</td>
</tr>
<tr>
<td>Human-Computer Interaction (HCI)</td>
<td>1</td>
</tr>
<tr>
<td>Markov Decision Process (MDP)</td>
<td>16</td>
</tr>
<tr>
<td>Ontology</td>
<td>29</td>
</tr>
<tr>
<td>Partially Observable Markov Decision Process (POMDP)</td>
<td>19</td>
</tr>
<tr>
<td>Policy in POMDP</td>
<td>43</td>
</tr>
<tr>
<td>Reinforcement Learning (RL)</td>
<td>12</td>
</tr>
<tr>
<td>Sentiment Analysis</td>
<td>62</td>
</tr>
<tr>
<td>Trend Analysis</td>
<td>51</td>
</tr>
<tr>
<td>Wavelet Transform</td>
<td>56</td>
</tr>
</tbody>
</table>
Table of Appendices

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>HCI infrastructure model</td>
<td>110</td>
</tr>
<tr>
<td>B</td>
<td>ECAs as a reflex agent</td>
<td>111</td>
</tr>
<tr>
<td>C</td>
<td>Different Avatars</td>
<td>115</td>
</tr>
<tr>
<td>D</td>
<td>Textual and Speech-based emotions discovery from a user’s interaction</td>
<td>118</td>
</tr>
<tr>
<td>E</td>
<td>Android Studio setup</td>
<td>121</td>
</tr>
<tr>
<td>F</td>
<td>Domain specific ECA customization using Facegen Software with Brand</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td>Ambassador’s Alter Ego</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>User-Agent chat conversation Ontology-based customization: Online Book</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>Search portal</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>User-Agent chat conversation: Sentiment Analysis rewards</td>
<td>126</td>
</tr>
<tr>
<td>I</td>
<td>Belief State History (BSH) database information for different users</td>
<td>128</td>
</tr>
<tr>
<td>J</td>
<td>Q-Learning: Q-values Rewards data</td>
<td>130</td>
</tr>
<tr>
<td>K</td>
<td>Policy improvement</td>
<td>131</td>
</tr>
</tbody>
</table>
Abbreviations/Symbols

AI: Artificial Intelligence

BCI: Brain-Computer Interface

BMI: Brain-Machine Interface

BS: Belief State

BSH: Belief State History

CHI: Computer-Human Interaction

DM: Dialogue management

DNI: Direct-Neural Interface

DRL: Deep Reinforcement Learning

DWT: Discrete Wavelet Transformation

ECA: Embodied Conversational Agent

FSM: Finite State Machine

GP: Gaussian Process

GUI: Graphical User Interfaces

HCI: Human-Computer Interaction

HMI: Human-Machine Interaction

HSVII: Heuristic Search Value Iteration

LSTM: Long Short-Term Memory
MMI: Man-Machine Interaction / Mind-Machine Interface

NCI: Neural-Control Interface

NLG: Natural Language Generation

NLP: Natural Language Processing

NLU: Natural Language Understanding

NN: Neural Network(s)

POMDP: Partially Observable Markov Decision Process

RE: Requirements Elicitation

RL: Reinforcement Learning

SDS: Spoken Dialogue Systems

SLP: Structured Language Processing

SOTA: State-Of-The-Art

SPL: Software Product Line

VPA: Virtual Personal Assistants

VUI: Voice User Interfaces

WT: Wavelet Theory
Chapter 1: Introduction

1.1 Motivation

In the world of *Natural Stupidity*, we need an *Artificial Intelligence*, therefore we require a system to incorporate these advanced technologies in our day-to-day life, that means, people’s life should be less stressful to get things done with the help of available technological resources like HCI (Human-Computer Interaction) and an interaction with an ECA (Embodied Conversational Agent) and this thesis directs the work in the direction of more advanced work done using HCI and ECA as a tool to involve the POMDP and deep reinforcement learning to discover user’s intention and to improve the accuracy from the user’s input.

The idea and the motivation is to offer better understanding while a real user communicates with an agent that can be helpful to enhance the user operation with the existing applications like, Apple’s Siri, Microsoft’s Cortana, Google’s OK Google, Amazon’s Alexa, etc. in the field of AI, to detect the emotion from user’s intentions and to follow the set of rules offered by the POMDP (Partially Observable Markov Decision Process) to find the relatable policy based on the user data to check whether user’s intentions are being fulfilled or not!

1.2 Introduction to Human-Computer Interaction (HCI)

**Definition- Human-Computer Interaction (HCI):** Human-Computer Interaction (HCI) is a field of research within the design and use of computer technology, focused on the interfaces between people (users) and computers. Researchers in the field of HCI both observe the ways in which humans interact with computers and design technologies that let humans interact with computers in novel ways. As a field of research, human-computer interaction is situated at the intersection of [4],
The recent advances in cognitive psychology and related sciences lead us to the conclusion that knowledge of human cognitive behaviour is sufficiently advanced to enable its applications in computer science and other practical domains. The goal is to help to create an applied information-processing psychology, as with all applied science, this can be only done by working within some specific domain of application. This domain is called Human-Computer Interface for the Interaction. The entire world is in the midst of transforming itself to use the power of computers throughout its entire fabric—wherever information is used and that transformation depends critically on the quality of human-computer interaction [5]. Most of the times computation becomes ubiquitous and our environments are enriched with new possibilities for communication. The field of HCI confronts difficult challenges of supporting complex tasks, such as mediating networked interaction, and managing and exploiting the ever-increasing availability of digital information. So, the research requires a more theoretical foundation that not only addresses the critical issue but also focuses on a human-centred approach [6].

The user interacts directly with hardware for the human input and output such as displays, e.g. through a graphical user interface. Software and hardware must be matched so that the processing of the user input is fast enough, the latency of the computer output should not disruptive to the workflow [7]. Voice User Interfaces (VUI) are used for speech recognition and synthesizing systems, and the emerging multi-modal and Graphical User Interfaces (GUI) allow humans to engage with embodied character agents in a way that cannot be achieved with other interface paradigms. The growth in the human-computer interaction field has been in the quality of interaction, and in different research branching in its history.
Instead of designing regular interfaces, the different research branches have had a different focus on the concepts of multimodality rather than unimodality, intelligent adaptive interfaces rather than command/action-based ones, and finally active rather than passive interfaces. For instance, more recently, sensors like video cameras and eye trackers can be used to feed the physiological information of humans back to computer systems [8].

![Diagram of human-computer interaction](image)

_Figure 1 The user interacts with the computer over this software interface using the given input and output (I/O) hardware_

Such information can be used by computers to dynamically adapt the content of interfaces. Thus, computers could develop responsiveness to cognitive load and human emotion [9]. Kindly refer Appendix A: HCI infrastructure model, for detailed discovery description in the field of HCI.
1.3 Introduction to Embodied Conversational Agent (ECA)

**Definition - Embodied Conversational Agent (ECA):** The ECAs in artificial intelligence, also sometimes referred to as an interface agent [10], is an intelligent agent that interacts with the environment through a physical body within that environment. Agents that are represented graphically with a body, for example, a human or a cartoon animal, are also called embodied agents, although they have only virtual, not physical, embodiment.

A branch of AI focuses on empowering such agents to interact autonomously with human beings and the environment. Mobile robots are one example of physically embodied agents; Ananova and Microsoft Agent are examples of graphically embodied agents. Embodied conversational agents have embodied agents (usually with a graphical front-end as opposed to a robotic body) that are capable of engaging in conversation with one another and with humans employing the same verbal and nonverbal means that humans do (such as gesture, facial expression, and so forth) [11]. There are different terms for ECAs like, intelligent agent, Avatar, chatbot, etc. An Intelligent Agent (IA) is an autonomous entity which observes through sensors and acts upon an environment using actuators (i.e. it is an agent) and directs its activity towards achieving goals (i.e. it is "rational", as defined in economics). Intelligent agents may also learn or use knowledge to achieve their goals. They may be very simple or very complex. A reflex machine, such as a thermostat, is considered an example of an intelligent agent [12]. One application of ECAs is automated online assistants, where they function to perceive the needs of customers in order to perform individualized customer service. Such an agent may basically consist of a dialogue system, an avatar, as well as an expert system to provide specific expertise to the user. They can also be used to optimize the coordination of human groups online [13]. In computing, an avatar is the graphical representation of the user or the user's alter ego or character. An icon or figure representing a person in a video game, Internet forum, etc. It may take either a three-
dimensional form, as in games or virtual worlds or a two-dimensional form as an icon in Internet forums and other online communities [14]. For the mean of planning the biological intelligence and to handle the challenges, there is a need to apply robot programming, assembly planning, virtual reality, neural networks, for the incorporation of the intelligence [15].

ECAs can be used as virtual embodiments of embodied agents, which are driven more or less by artificial intelligence rather than real people. Automated online assistants are examples of avatars used in this way. Such avatars are used by organizations as a part of automated customer services in order to interact with consumers and users of services. This can avail for enterprises to reduce their operating and training cost. A major underlying technology to such systems is natural language processing. Some of these avatars are commonly known as "bots". Famous examples include IKEA's Anna, an avatar designed to guide users around the IKEA website. Such avatars can also be powered by a digital conversation which provides a little more structure than those using NLP, offering the user options and clearly defined paths to an outcome. This kind of avatar is known as a Structured Language Processing or SLP Avatar. Both types of avatar provide a cost-effective and efficient way of engaging with consumers [14]. Kindly refer, Appendix B: ECAs as a , where ECAs are grouped into five classes based on their degree of perceived intelligence and capabilities [16].

Different kinds of ECA avatar are shown in Appendix C: Different Avatars, with different expressions. There can be 100+ possibilities of different gestures and mixed emotions, but only major ones are shown here. There could be thousands of possible combinations available so that multiple emotions/gestures can be combined to create one single 3d model.

An ECA chatbot (also known as a talkbots, chatterbot, Bot, IM bot, interactive agent, or Artificial Conversational Entity) is a computer program or an artificial intelligence which
conducts a conversation via auditory or textual methods. Such programs are often designed to convincingly simulate how a human would behave as a conversational partner, thereby passing the Turing test. Chatbots are typically used in dialogue systems for various practical purposes including customer service or information acquisition. Some chatterbots use sophisticated natural language processing systems, but many simpler systems scan for keywords within the input, then pull a reply with the most matching keywords, or the most similar wording pattern, from a database [17].

There are certain emotions which were considered which can be detected easily without any manipulation from the user input, so the basic emotions which are considered are joy, sadness, anger, disgust, shame, guilt, fear and above all, neutral emotion is the default [18] [19] [20] [21]. When we focus on emotion recognition from user input (without POMDP specification) the bucket full of work can be found to detect an emotion from user’s intention during the communication and that can be found in different areas such as text-based emotion analysis [22], use of AI to detect emotion [23], automation of detecting emotions from text [24].

1.4 Problem statement

POMDP-based dialogue managers have been unfitting for deployment because POMDP requires several thousands of dialogues for training, but recent studies show that it would be possible to train a POMDP-based dialogue system on just a few hundred dialogues corresponding to the interactions with users. The old approach of POMDP and Gaussian Processes SARSA (State-Action-Reward-State’-Action’) is outperformed by the deep RL (Reinforcement Learning) algorithms in many ways over the years.

Now, the problem is to enhance the usability of the user to discover the almost exact intention from the user input, with the old approach of POMDP (the work in this area is being carried
out for more than 2 decades) [old approach, in the sense of not-so-advanced approach], the accuracy and scalability cannot be achieved on a huge scale.

Shortcomings of the POMDP-model are outperformed and at the same time all the advantages are retained of iteration of Belief State History (BSH) to find the trend using Discrete Wavelet Transformation (DWT) to improve the intention discovery.

How to make the agent more intelligent in terms of generating natural dialogues? To solve this problem the focus should be on the goal-driven dialogue conversation, sentiment learning and way to improve the policy, etc.

There is a need to implement constant learning for the agent to reduce the dialogue length using machine learning approach- Reinforcement Learning (RL), which encourages a belief or a pattern of behaviour in terms of the rewards. Using sentiment analysis and RL techniques can improve the intention discovery of the user’s aim of reaching the goal of using DM and context-driven communication.
Chapter 2: A Literature review

This chapter focuses on the discussion about the existing work that has been built by researchers over the years. The next step is to dig in different approaches starting from the dialogue management to MDP and POMDP methods. This chapter follows the structure showing how different entities relate to each other starting from conversation agent with Dialogue Management (DM) with Markov Decision Process (MDP) to POMDP with belief state history with deep learning to improve the Policy.

For HCI, making Artificial Intelligence (AI) more human [25] is necessary, that means to conduct HCI between ECA and a user to make the conversation real enough, AI is required to shape the intelligent ECA. The parameters to include for this progress, involves environmental influence, agent parameters, generality, flexibility, learning speed, effect of cheap talk, loyalty, honesty, speech profile, etc. and then turing test (some work can be based on the grounds of speech act and dialogue act, turn-taking, grounding, etc. [26]) can be conducted over the agents to test the intelligence of the ECA against user-to-user for more advanced ECA-to-user conversation.

In the process of automation, two aspects are important to be considered and those are, appearance and intelligence, that how effective the ECA can be developed for the ease of access for the replica of human face-to-face interaction in terms of HCI to make it more reliable and more real. That can be decided by the animated speech, lip movement and facial expressions, eye, head and body movements to the gestures, emotion expression, etc. and last but not the least the automation process involves NLU and NLP to take an optimized decision from the ECA’s end for the user that can be useful in detecting the intention of the user which analyze the understanding of what users aim or plan or actually intended about the requirement and to find out, agent needs to be intelligent enough and that is called as an
**intention discovery.** With the help of intention which is discovered by the agent, what user meant, then interaction begins in a true sense as agent will be exchanging the words based on the discovery of user’s aim and that focus will be carried forward when a user and an agent interacts and the number of conversations it takes for user and agent to reach the desired result of a user, is known as a **dialogue length** and to improve the policy, there is an importance of improving the optimization level of dialogue that how minimal or the shortest length it took for an agent to reach the goal and that dialogue length can be optimized by anticipating the user knowledge level. The user knowledge level can be categorized in different categories (Different knowledge level selectors to decide the policy based on the understanding of user), such as expert, professional, amateur, and novice, that how advanced the user is or how basic knowledge a user must carry forward the interaction and that will decide the policy structure of interaction with the help of Dialogue Management (DM).

2.1 Embodied Conversational Agent (ECA) with Dialogue Management (DM)

ECAs have the same ability as per the human in terms of the representation properties as face-to-face conversation which includes, an ability to recognize and respond to the input, ability to generate an output, use of conversational functions are utilized under HCI branch. So the motivation works in terms of the interaction, the included attributes are, intuitiveness, redundancy and modality switching, the social nature of the interaction, etc. [27] The input to the DM is the human utterance, usually converted to some system-specific semantic representation by the Natural language understanding (NLU) component. The DM usually maintains some state variables, such as the dialogue history, the latest unanswered question, etc., depending on the system [28]. That is how we can combine the great features of ECA with Dialogue Manager to iterate the functionalities of a communicative artificial intelligent agent to satisfy the aim of this system.
The communicative behaviours can be considered for the experimentation purpose as to how ECA will interact with a real USER, so for that three kinds of approaches are considered in the field of DM using ECA. And they are Content-only feedback (CONT), Content + Emotional feedback (EMO), Content + Envelope feedback (ENV) [29].

The conversational functions are realized physically as conversational behaviours [30] that can be found while working with the dialogue manager that is explained in detail in the following sections. The related work can be found in REA [27] and Greta [31] (an interactive ECA platform), etc. with socio-emotional and communicative behaviours. The ECAC provides the representation of conversational characters through the convergence of animated interface agents and human-computer dialogue systems [32]. Therefore to offer more readily interact with natural language through personified agent using DM. The different knowledge domains can be explored using the DM to reach the real user with different semantics [33].

The decision-making approach in DM world, how to reach a goal and how the decisions are made based on the data we have, there have been plenty of research work has been carried
out in terms of Breadth First Search (BFS), Depth First Search (DFS) and A* algorithms in the deterministic environment where actions are predictable. These approaches work when we have a finite set of data, but it can’t work under uncertainty to-predict-the-future-goals environment. Various approaches are proposed since the year 1960 to understand and model the decision-making task under uncertainty like, Markov Decision Process (MDP), etc. and ECA cannot help in discovering the intention we have taken Partially Observable Markov Decision Process (POMDP) into an action which is an extension of an MDP.

2.2 Dialogue management

Dialogue management is a system consisting of a dialogue manager, which is a core of a Spoken Dialogue System (SDS) with its main features like, tracking dialogue states and maintaining a dialogue policy which decides how the system reacts on given dialogue state. Recently, different approaches in automatic dialogue management policy optimization are there among the Reinforcement Learning (RL) and the POMDP has been the most famous one [34]. Statistical approaches to dialogue modelling allow automatic optimization of the SDS. An SDS is typically designed according to the structured-ontology [35]. Instead of letting a human expert write a complex set of decision rules, it is more common to use reinforcement learning. The dialogue is represented as a Markov Decision Process (MDP) - a process where, in each state, the DM has to select an action, based on the state and the possible rewards from each action. In this setting, the dialogue author should only define the reward function, for example: in tutorial dialogues, the reward is the increase in the student grade; in information seeking dialogues, the reward is positive if the human receives the information, but there is also a negative reward for each dialogue step. RL techniques are then used to learn policy, for an example, what kind of observation and feedback should we use in each state? etc. This policy is later used by the DM in real dialogues.
The different models existing in the DM domain are, Finite State Machine (FSM), frame-based or slot-filling, example-based, POMDP, etc. [36] as described below with their own advantages and disadvantages. Some other types of DM are described here [37] as a switch statement, finite state machine, goal-based, belief-based, etc.

### FSM
- **Advantages**
  - Clear structure
  - Easy to develop to effectively control the dialogue process
- **Disadvantages**
  - Not suitable for a complex dialogue task

### Frame-based
- **Advantages**
  - Can handle more complex inputs
- **Disadvantages**
  - Dialogues turns out to be unnatural

### POMDP
- **Advantages**
  - Very popular in theoretical studies
  - Proven to be a good model to deal with uncertain problems in speech recognition and language understanding
  - Can be applied to more fields by factoring its state space
- **Disadvantages**
  - Cannot handle multitopic tasks and the tagging and training corpus are expensive and time consuming

**Definition - Reinforcement Learning (RL):** To learn an optimum policy conducted by an agent by maximizing its cumulative reward. One of the advantages of RL-based dialogue management (DM) is the robustness. RL algorithms are mostly-data demanding, which leaves dialogue system developers in worry as there are usually few or even no data available at the early stage of development. Several methods have been proposed to mitigate this
A user simulator is often built using wizard-of-oz dialogue data, and then the simulator is used to train an RL-based DM. In recent studies, it has been shown that by incorporating domain knowledge into the design of kernel functions [34].

2.2.1 Dialogue management using Genetic Algorithm

A Genetic Algorithm is implemented in optimizing DM policies which are comprehensible to human and easy to verify and modify. The underlying data could be intuitive. This algorithm performs on the human-readable domain language to sketch the basic structure of the DM policy. The template used for the genetic algorithm and dialogue policy can be described as a general optimization framework.

The genetic algorithm can be divided into three parts as shown below.

Genetic algorithm using Elitism

- In genetic algorithm, two kinds of genetic operators are used, i.e. mutation and crossover. During crossover, two parents are selected, then random parts of the two parents are exchanged, giving birth to a new child. The mutation operator checks each component of a chromosome sequentially, either leaving it intact or perturbing it randomly.

Policy optimization with a user simulator (Episodic fitted Q-iteration)

On-corpus Q-points regression

Note that, rule-based and Genetic Algorithm-based RL are instantiations of the same policy template and the simulation offers that this algorithm works best in DM policy learning.
2.2.2 Uncertainty estimates in deep reinforcement learning

The step-by-step procedures in the achievement of the uncertainty estimation, there are few algorithms that need to have a look on!

Uncertainty estimates the policy and allows the system to generalize across different noise levels and mitigate errors incurred by the user input, therefore, resulting in a more robust dialogue manager [35].

- Value-based deep RL [38]
  - The value approximation and policy gradients are two major methodologies in RL frameworks. As the name suggests the approximations are required for large or continuous belief and action spaces so that generalization can be stored over the necessary state space. So, some of the RL techniques are used, such as,
    - Generalization over the belief space is efficient and the need for summary spaces will be eliminated, which consumes less space than GP-SARSA.
    - Memory requirements are limited, that could be determined in an advance
    - Deep architecture with several hidden layers can be efficiently used for complex tasks and environments.

Some of the other work of RL in a dialogue system is impressive the field of rating the dialogue success with Neural Networks (NN) [39]. The deep RL based end-to-end framework for both dialogue state tracking and dialogue policy that addresses the issues existing in the older approaches [40].
The model described in a paper (Deep Variational reinforcement learning for POMDPs) [41], known as DVRL (Deep Variational Reinforcement Learning) introduces an inductive bias that allows an agent to learn a generative model of the environment and perform inference in the model to effectively aggregate the available information. This method solves POMDPs for, given only a stream of observations, without knowledge of the latent space or the transition and observation functions operating in that space.

**Definition- Belief State (BS):** The SDS enables HCI via speech so that the DM has two aims to maintain the dialogue state based on the current spoken language understanding input and the conversation history, choose a response according to its dialogue policy. To offer the robustness and to track the distribution of all the dialogue states at every dialogue turn, called as a belief state. Then the system response is based on the belief state rather than an inaccurate estimate of most likely dialogue state [42]. The state-of-art statistical methods for policy learning are based on RL which makes it possible for a system to learn from the interaction. In GP-RL the kernel function defines the correlations of the objective function given different belief states, which can significantly speed up the policy optimization.

The symbolic dialogue act form of NLU will be passed on to the DM. the classic DM is a charge of both state tracking and policy learning. The state tracker will keep tracking the evolving slot value pairs from both agent and user and based on the conversation history, a query may be formed to interact with an external database to retrieve the available result. In every turn of the dialogue, the input is, the state tracker is updated based on the retrieved results from the database and the latest user dialogue action which provides an output as a dialogue state. The dialogue state often includes the latest user dialogue action, conditioned on the dialogue state, the dialogue policy is generated with the help of next available agent action [43].
2.2.3 Multi-party dialogue with ECA

In the context of Multi-party dialogue with virtual humans (ECA), the paper (Towards a generic framework for multi-party dialogue with virtual humans) [44], a genetic framework is used to aid in the development of multi-modal, multi-party dialogue. It contains the mechanisms inspired by social practice theory for both action selection and timing which includes the handling of interruption. The expectations are usually utilized to paint a rough sketch of what a socially-acceptable interaction should look like. The agent’s aim is to follow the set of candidate actions under exceptional circumstances as a result.

The paper (Deep decentralized multi-task multi-agent reinforcement learning under partial observability) [45] addresses the problem of multi-task multi-agent RL under partial observability, using decentralized single-task learning approach that is robust to concurrent interactions of teammates. It also presents an approach for distilling single-task policies into a unified policy that performs well across multiple related tasks without any explicit provision of task identity.

The paper (Learning to communicate with deep multi-agent RL) [46] proposed two approaches: Reinforced Inter Agent Learning (RIAL); and Differentiable Inter Agent Learning (DIAL). The RIAL approach uses deep Q-Learning whereas, DIAL exploits the fact that, during learning agents can backpropagate error derivatives through communication channels. One of the first attempt at learning communication and language with deep learning approaches were introduced, which offers novel environments and successful techniques for learning communication protocols.

2.3 Markov Decision Process (MDP)

**Definition- Markov Decision Process (MDP):** Markov Decision Process (MDP) is an output of the continuous cast of Dialogue management (DM) which is composed of a finite set of
actions, a continuous multivariate belief state space and a reward function [47]. A Markov
decision processes (MDPs) is a discrete time stochastic control process. It provides a
mathematical framework for modelling decision making in situations where outcomes are
partly random and partly under the control of a decision maker. MDPs are useful for studying
optimization problems solved via dynamic programming and reinforcement learning [48].

At each time step, the process is in some state $s$, and the decision maker may choose any
action $a$ that is available in state $s$. The process responds at the next time step by randomly
moving into a new state $s'$, and giving the decision maker a corresponding reward $R_a(s, s')$.
The probability that the process moves into its new state $s'$ is influenced by the chosen action.
Specifically, it is given by the state transition function $P_a(s, s')$. Thus, the next state $s'$
depends on the current state $s$ and the decision maker's action $a$. But given $s$ and $a$, it is
conditionally independent of all previous states and actions; in other words, the state
transitions of an MDP satisfies the Markov property. Markov decision processes are an
extension of Markov chains; the difference is the addition of actions (allowing choice) and
rewards (giving motivation). Conversely, if only one action exists for each state (e.g. "wait")
and all rewards are the same (e.g. "zero"), a Markov decision process reduces to a Markov
chain. A Markov decision process is a 5-tuple $(S, A, P_a, R_a, \gamma)$, where

- $S$ is a finite set of states,
- $A$ is a finite set of actions ($A_s$ is the finite set of actions available from state $s$),
- $P_a(s, a, s') = P_{s_{t+1}} = s' | s_t = s, a_t = a$ is the probability that action $a$ in state $s$ at
time $t$ will lead to state $s'$ at time $t+1$,
- $R_a(s, a, s')$ is the immediate reward (or expected immediate reward) received after
  transitioning from state $s$ to state $s'$, due to action $a$. 

17
• $\gamma \in [0, 1]$ is the discount factor, which represents the difference in importance of future rewards and present rewards.

2.3.1 Shortcomings of MDPs

The core problem of MDPs is to find a "policy" for the decision maker: a function $\pi$ that specifies the action $\pi(s)$ that the decision maker will choose when in state $s$. Once a Markov decision process is combined with a policy in this way, this fixes the action for each state and the resulting combination behaves like a Markov chain. The goal is to choose a policy $\pi$ that will maximize some cumulative function of the random rewards, typically the expected discounted sum over a potentially infinite horizon:

$$\sum_{t=0}^{\infty} \gamma^t R_{at}(s_t, s_{t+1}), \text{ where } at = \pi(s_t)$$

*Equation 1 Decision maker in an MDP to find Policy*

where $\gamma$ is the discount factor and satisfies $0 \leq \gamma < 1$. (For example, $\gamma = 1/(1 + r)$ when the discount rate is $r$.) $\gamma$ is typically close to 1. Because of the Markov property, the optimal policy for this problem can indeed be written as a function of $s$ only, as assumed above. MDP is fully observable in stochastic environment actions are random.

Therefore, the solution could be termed as a Partially Observable Markov Decision Process (POMDP) that would solve the optimization problem of MDP in the stochastic environment. Suppose we know the state transition function $P$ and the reward function $R$, and we wish to calculate the policy that maximizes the expected discounted reward. The standard family of algorithms to calculate this optimal policy requires storage for two arrays indexed by state: value $V$, which contains real values, and policy $\pi$ which contains actions. At the end of the algorithm, $\pi$ will contain the solution and $V(s)$ will contain the discounted sum of the rewards to be earned (on average) by following that solution from state $s$. 
2.4 Partially Observable Markov Decision Process (POMDP)

**Definition- Partially Observable Markov Decision Process (POMDP):** The solution above in [Error! Reference source not found.], assumes that the state \( s \) is known when action is to be taken; otherwise \( \pi(s) \) cannot be calculated. When this assumption is not true, the problem is called a partially observable Markov decision process or POMDP. A partially observable Markov decision process (POMDP) is a generalization of a Markov decision process (MDP). A POMDP models an agent decision process in which it is assumed that the system dynamics are determined by an MDP, but the agent cannot directly observe the underlying state. Instead, it must maintain a probability distribution over the set of possible states, based on a set of observations and observation probabilities, and the underlying MDP. The POMDP framework is general enough to model a variety of real-world sequential decision processes [49]. The POMDP helps to build the discrete-time relationship between an agent and its environment. Formally, a POMDP is a 7-tuple \((S, A, T, R, \Omega, O, \gamma)\), where

- \( S \) is a set of states: The input which is divided into a finite set of possible states
- \( A \) is a set of actions: A finite set of possible ACTIONS available and actions are information-driven or goal-driven
- \( T \) is a set of conditional transition probabilities between states: It captures the probabilistic relationship between the states and the actions executed to change the state of the world
- \( R \) is the reward function: It gives the relative measure of desirability to be in a state
- \( \Omega \) is a set of observations: It captures the probabilistic relationship between the state and observations
- \( O \), is a set of conditional observation probabilities: a finite set of observations of the state
- $\gamma$, is the discount factor: The discount factor decides how much immediate rewards are favoured over future rewards.

The POMDP interaction between user and agent can be explained algorithmically in a flow chart as shown in Figure 2 [50]. In a POMDP we add a set of observations to the model. So instead of directly observing the current state, the state gives us an observation which provides a hint about what state it is in. The observations can be probabilistic; so we need to also specify an observation function. This observation function simply tells us the probability of each observation for each state in the model. We can also have the observation likelihood depend on the action if we like [51].

![Figure 2 Modularized view of the interaction between the dialogue manager and the user in a dialogue management context](image)

The POMDP model is further explained in Chapter 3: Detailed followed by its details on the model, belief state, policy.
2.5 Dialogue Management (DM) in different domains

This section is explained further this section and in the next section, where the theory of DM is explained. The following subsections explain different applications in different models to achieve the interaction possible between the user and an agent in the DM domain module.

2.5.1 Dialogue Management (DM) with POMDP

In a dialogue management context, the agent is the dialogue manager, the system. Part of the POMDP environment represents the user's state and the user's action. Depending on the design for a dialogue application, the rest of the POMDP environment might be used to represent other modules such as speech recognition and emotion recognition. Because the user’s state cannot be directly observed, the agent uses a state estimator (SE) to compute its internal belief (called belief state) about the user’s current state and an Action Selector (AS) where the policy \( \pi \) is implemented to select actions. The SE takes as its input the previous belief state, the most recent system action and the most recent observation and returns an updated belief state. The AS takes as its input the agent's current belief state and returns an action that will be sent to the user [50].

The paper (Gaussian Processes for POMDP-based dialogue manager optimization) [52] suggests the approach of optimizing the POMDP-based DM by Gaussian Processes. As discussed, SDS (Spoken Dialogue System) enables HCI where the primary input is speech. This paper is discussed in the next chapter in detail after the introduction of policy and belief state for better understanding.

2.5.2 Dialogue Management with Reinforcement Learning

As the dialogue assistants are rapidly becoming an obvious daily aid, to avoid the significant effort needed to hand-craft the required dialogue flow, the DM module can be cast as a continuous MDP and rained through RL. Several RL models are there which are
implemented recently over some years. The paper (A benchmarking environment for reinforcement learning based task-oriented dialogue management) [53] proposes the set of challenging simulated environments for DM model development and evaluation. The DM is cast as a continuous MDP composed of a continuous multivariate BS space $B$, a finite set of actions $A$ and a reward function $R(b_t, a_t)$. The BS $b$ is a probability distribution over all the possible discrete states. At a given time $t$, the agent (policy) observes the BS $b_t \in B$ and executes an action $a_t \in A$. The agent then receives a reward $r_t \in R$ drawn from $R(b_t, a_t)$. The policy $\pi$ is defined as a function $\pi: B \times A \rightarrow [0, 1]$ that with probability $\pi(b, a)$ takes an action an in a state $b$. For any policy $\pi$ and $b \in B$, the value function $V^\pi$ corresponding to $\pi$ is defined as:

$$V^\pi(b) = \mathbb{E}\{r_t + \gamma r_{t+1} + \ldots \mid b_t = b, \pi\}$$

*Equation 2 The policy estimation using DM in RL*

Where, $\gamma$, $0 \leq \gamma \leq 1$, is a discount factor and $r_t$ is a one-step reward. The objective of reinforcement learning is to find an optimal policy $\pi^*$, i.e. a policy that maximizes the value function in each belief state. Equivalently, we can estimate the unique optimal value function $V^*$ which corresponds to an optimal policy. In both cases, the goal is to find an optimal policy $\pi^*$ that maximizes the discounted total return,

$$R = \sum_{t=0}^{T-1} \gamma^t r_t(b_t, a_t)$$

*Equation 3 Estimation of unique optimal policy*
over dialogue with \( T \) turns, where \( r_t(b_t, a_t) \) is the reward when taking action \( a_t \) in dialogue state \( b_t \) at turn \( t \) and \( \gamma \) is the discount factor.

Shortcomings: A set of extensive simulated DM environments along with a comparison of several RL algorithm using an open-domain toolkit is available. A large amount of improvement is still necessary for the data-driven model to match the performance of handcrafted policies which are used in larger domains.

In another paper (Strategic dialogue management via deep reinforcement learning) [54] based on Deep Reinforcement Learning (DRL) for DM where the work is carried out by training the intelligent agents with strategic conversational skills in a situated dialogue setting. The previous approach suggests that behaviour for a strategic agent can be trained using deep learning with large action sets can be prohibitively expensive due to computation time. Therefore, this paper suggests a solution to the limitation of learning from constrained action sets rather than static action sets.

The latest work (Feudal reinforcement learning for dialogue management in large domains) [55] in the RL field proposes a promising approach to the solve dialogue policy optimization. The problem is, traditional RL algorithms fail miserably when it comes to large domain dimensionality. It proposes a novel approach on DM architecture based on Feudal RL which decomposes the decision into two steps such as, first a master policy selects a subset of primitive actions, then a primitive action is chosen from the specifically selected subset. This model is achieved based on Deep-Q networks, which outperforms the previous state-of-the-art (SOTA) in dialogue domains and environments without any need of additional reward signal.

The paper (Sub-domain modelling for dialogue management with hierarchical reinforcement learning) [56] focuses on solving the problem where policy learning for dialogue systems is
very challenging. Standard RL methods do not provide an efficient framework for modelling these kinds of dialogues. So, the researchers propose to focus on the under-explored problem of multi-domain dialogue management. Here, the hierarchical RL uses the options framework. This architecture learns faster and arrives at a better policy than the existing flat ones do. Then displaying the pretrained policies can be adapted to more complex systems by adding an additional set of new actions. Hierarchical RL in DM paradigm be a control problem which estimates a distribution over possible user requests- belief states and chooses what to say to the user, that means which actions to take to maximize positive user feedback-the reward.

In the domain of DM domain adaption using Gaussian Process (GP) using RL is explained in the paper (Dialogue manager domain adaption using Gaussian process reinforcement learning) [57] that the data-driven machine learning methods have been applied to dialogue modelling and the results achieved for limited-domain applications are comparable to or outperform traditional approaches. The methods based on Gaussian Processes are particularly effective as they enable good models to be estimated from limited training data which additionally helps in the explicit estimation of the uncertainty which is particularly useful for RL.

The paper (Distributed dialogue policies for multi-domain statistical dialogue management) [58] focuses on the hierarchical distributed dialogue architecture in which policies are organized in a class hierarchy aligned to an underlying knowledge graph. The Gaussian process-based RL is used to represent within the framework, generic policies can be constructed which provides acceptable user performance.

The paper (Deep learning for dialogue Systems) [59] focuses on the motivation of the work on conversation-based intelligent agents in which the core underlying system is task-oriented
dialogue systems. The different aspects are considered as a part of the survey like language understanding; dialogue management; natural language generation; end-to-end learning for dialogue system; dialogue breadth; dialogue depth; etc.

2.5.3 Dialogue management (DM) using Long Short-Term Memory (LSTM)-based natural language generation

Natural Language Generation (NLG) is a critical component of SDS and the LSTM model [60] learns from the unaligned data by jointly optimizing sentence planning and surface realization using a simple cross entropy training criterion. Various algorithms used for NLP are mainly dependent on the RNN (Recurrent Neural Network) [61]. This model is based on Recurrent Neural Network (RNN) architecture. Deep Neural Networks (DNN) enable increased discrimination by learning multiple layers of features and represents the state-of-the-art (SOTA) for many applications for speech recognition and NLP. The LSTM-based natural language generator can be easily extended to be deep in both space and time by stacking multiple LSTM cells on top of the original structure.

2.5.4 Dialogue Management (DM) approaches for belief tracking

This work done by [62] explains the simple models for SDS architecture, DM, DM as MDP, DM as POMDP, belief tracking, Hidden Information State (HIS) model, etc. in the simplest way possible. All these models are summarized below.

![Diagram of Dialogue Management as a Markov Decision Process (MDP)](image)

*Figure 3 Dialogue Management (DM) as a Markov Decision Process (MDP)*
Figure 4 Dialogue Management (DM) as a Partially Observable Markov Decision Process (POMDP)

Data
- Noisy observations of dialogue states
- Reward - a measure of dialogue quality

Model
- Partially Observable Markov Decision Process

Prediction
- Distribution over possible dialogue states - Belief States
- Optimal system actions

Figure 5 Spoken Dialogue Systems (SDS)

Waveform

Speech Recognition → Semantic Decoding → Dialogue Management → Ontology

Speech synthesis → Natural Language Generation

Distribution over text hypotheses
Distribution over dialogue acts
For the belief state tracking, the requirement is a summation over all possible states at every dialogue turn which is intractable. The requirements are explained below.

**Dialogue history**
- Markov property
  - The system needs to keep track of what happened so far in dialogue, called markov property

**Task-oriented dialogue**
- User goal
  - The system needs to know what the user wants, known user goal

**Robustness to errors**
- User act
  - The system needs to know what the user says, known user act

---

**Figure 6 Dialogue Management (DM)**

**Figure 7 Belief Tracking**
The approach shown has the following advantages over the research in the domain of dialogue management,

- Properties of belief tracking for dialogue management include Markov assumption, being able to model the user goal and being robust to speech recognition errors
- Generative models for belief tracking are based on partially observable Markov decision processes
- Hidden Information State (HIS) model decomposes the dialogue state into the user goal, the user action and the dialogue history. Transitions are hand-crafted, and the goals are grouped together to allow tractable belief tracking
- Bayesian Update of Dialogue State (BUDS) model further factorizes the state which allows tractable belief tracking but also the learning of the shapers of distributions via Expectation propagation
2.5.5 Dialogue Management (DM) based on Ontology

**Definition- Ontology:** Ontology is the philosophical study of being. More broadly, it studies concepts that directly relate to being, in particular, becoming, existence, reality, as well as the basic categories of being and their relations. Traditionally listed as a part of the major branch of philosophy known as metaphysics, ontology often deals with questions concerning what entities exist or may be said to exist and how such entities may be grouped, related within a hierarchy, and subdivided according to similarities and differences [63].

For Virtual Personal Assistants (VPA), this approach (An ontology-based dialogue management system for virtual personal assistants) [64] in the DM domain features the potential solution to core DM problems. The paper offers solutions to the problems by running the VPA example domain. This domain is believed to be the first commercially available, fully implemented DM system that employs the ontologies, reasoning and ontology-based rules for, domain model representation and reasoning, dialogue representation and state-tracking, response generation, etc. This model is a declarative, knowledge-based system which can be customized based on ontologies and rules-based as this model is domain-independent like generic, but with dialogue-specific upper-level ontology and DM rules. The Ontology-based rules can be represented as below as dynamics of the dialogue. Therefore, the parameters listed below facilitate the ontology.

- Domain model representation
- Dialogue ontology
- Domain-specific reasoning
- Reasoning to compute VPA’s responses
- Reasoning to handle polysemy and ambiguity of natural language and dialogue
2.5.6 Other models in Dialogue Management (DM) models

The approach is given in this paper (TFSM-based dialogue management model framework for effective dialogue systems) [36] describes, the aim to provide a service of information inquiry and effective interaction, is proposed by constructing Two Finite State Machines (TFSM) to model the user and the system respectively and they simulate the dialogue process as an information exchange between the two states machines. Based on the Finite State Machine (FSM) this DM model works by viewing the system as a finite automaton and does not consider the context of the user’s state, therefore, the transfer of user’s intentions into dialogue can also be modelled then the dialogue process between the user and system is simulated in domain-independent framework based on information inquiry and effective interaction.
Chapter 3: Detailed Partially Observable Markov Decision Process model

3.1 Detailed POMDP in different approaches

As explained the 2.4, the POMDP outperforms other approaches in the DM domain module to build the HCI with ECAs to fulfil our need.

Example- In a POMDP environment, the lily pond (the transition in MDP as a frog in a pond jumping from lily pad to lily pad [65]) is covered by the mist, therefore the frog is no longer certain about the pad it is currently on. Before jumping the frog can observe information about its current location. This intuitive view is very appropriate to model the effective dialogue management system [66].

Idealized human-computer dialogue for an example can be explained as shown below [67] [68]. As shown in the image in Figure 9,

- Current state = dialogue modelling
- System action selection = dialogue management
- Dialogue state is unobserved, where
  - $G$ defines User’s goal
  - $A$ defines User’s real action
  - $S$ defines Conversation state
- Inference via Observation (defined by $O$).

May contain errors (for an example, Austin is an error state)
The decompose state variable into 3 different models as shown below.

Figure 9 Idealized HCI dialogue

The decompose state variable into 3 different models as shown below.
How can we reduce the length of the dialogue?

- For that purpose, we need to maintain the distribution over POMDP states. Let us take an example to understand this phenomenon.

- Typical approaches encode uncertainty through explicit state variables. Example,

```
System: How many pizzas?
User: Three, please.
System: You want two pizza?
User: No.
System: How many pizzas?
User: I want three.
System: You want three pizza?
User: Yes.
```

- Now, think of the possibility of maintaining a distribution over states, as a list of all belief states and updating the distribution at each timestamp.

- By this approach, belief state can be believed as a ‘cumulative’ confidence measure of overall user goals. Example,

```
System: How many pizzas?
User: Three, please.
System: Was that two or three?
```
User: Three

System: Coming right up!

- Now, partition the whole example step by step to understand the logic behind it.
- First, let us go through the explicit state variables example,
- The equation here would be,

\[ A_s = \pi(S) \]

*Equation 4 Calculate Uncertainty through explicit state variables*

| Number of Pizza: |  
|------------------|------------------|
| Variable status: | Undefined        |
| Confidence:      |                  |

*Table 1 Confidence measure 1*

System: How many pizza?

User: Three, please.

<table>
<thead>
<tr>
<th>Number of Pizza:</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable status:</td>
<td>Defined</td>
</tr>
<tr>
<td>Confidence:</td>
<td>Medium</td>
</tr>
</tbody>
</table>

*Table 2 Confidence measure 2*

System: You want two pizza?

User: No.

<table>
<thead>
<tr>
<th>Number of Pizza:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable status:</td>
</tr>
</tbody>
</table>

Table 3 Confidence measure 3

System: How many pizza?

User: I want three.

<table>
<thead>
<tr>
<th>Number of Pizza:</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable status:</td>
<td>Defined</td>
</tr>
<tr>
<td>Confidence:</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Table 4 Confidence measure 4

System: You want three pizza?

User: Yes.

- Secondly, let us go through the example of a confidence measure example,
- The equation here would be,

\[ As = \pi(P(S)) \]

Equation 5 Distribution maintained overstate as a cumulative confidence measure

Figure 11 Dialogue 1

System: How many pizzas?
User: Three, please.

System: Was that two or three?

User: Three

Figure 12 Dialogue 2

Figure 13 Dialogue 3

System: Coming right up!

- Now, we define the problem that, how to choose $A_s$ (system action).
- There are three specifications,
  - Treat as a design problem (for human designers)
  - Specify $R$, choose $A_s$ to maximum immediate reward (no planning)
  - Specify $R$, choose $A_s$ to maximum cumulative reward (planning POMDP)
In the following content, we can objectify policy as a portioning to map the situation into an action.

In the dissertation (Toward effective dialogue management using partially observable Markov decision process) [50] and the paper (Toward effective dialogue management using partially observable Markov decision processes) [26], the framework used for POMDP is explained as below. Take note that, as explained above, only the required tuples are considered in this work, which is different than the traditional POMDP framework.

Let \( S_t, A_t, Z_{t+1} \) and \( R_t \) be random variables taking their values from the sets \( S, A, Z, \) and \( R \) (the set of real numbers), respectively. At each time step \( t \), the environment’s state is \( S_t \). The agent selects an action \( A_t \) and sends it to the environment. The environment’s state changes to \( S_{t+1} \). The agent receives an observation \( Z_{t+1} \) and a reward value \( R_t \). Following this interaction description, the transition function \( T \), observation function \( O \), and reward function \( R \) are formally defined as follows.

The transition function is defined as \( T: S \times A \times S \rightarrow [0, 1] \). Given any state and action, \( s \) and \( a \) the probability of the next possible state, \( s' \), is

\[
P_{ss'}^{a} = T(s, a, s') = P\{S_{t+1} = s' | S_t = s, A_t = a\}, \text{ for all } t
\]

\textit{Equation 6 Transition function}

These quantities are called transition probabilities. Transition function \( T \) is time-invariant and the sum of transition probabilities over the state space

\[
\sum_{s' \in S} P_{ss'}^{a} = 1.
\]

\textit{Equation 7 Transition probabilities}
for all \((s, a)\).

The observation function is defined as \(O : S \times A \times Z \rightarrow [0, 1]\). Given any action and next state, \(a\) and \(s'\), the probability of the next observation, \(z'\), is

\[
\mathcal{P}^{\alpha}_{s'z'} = O(s', a, z') = P\{Z_{t+1} = z'| A_t = a, S_{t+1} = s'\}, \text{ for all } t
\]

*Equation 8 Observation function*

These quantities are called observation probabilities. Observation function \(O\) is also time-invariant and the sum of observation probabilities over the observation space

\[
\sum_{z' \in Z} \mathcal{P}^{\alpha}_{s'z'} = 1.
\]

*Equation 9 Observation probabilities*

for all \((a, s')\).

The reward function\(^3\) is defined as \(R : S \times A \rightarrow R\). Given any current state and action, \(s\) and \(a\) the expected immediate reward that the agent receives from the environment is

\[
R^{\alpha}_s = R(s, a)
\]

*Equation 10 Expected immediate reward*

Let \(R_{\min}\) and \(R_{\max}\) be the lower bound and upper bound of the reward function, that is to say

\[
R_{\min} < R(s, a) < R_{\max}, \text{ for all } (s, a)
\]

*Equation 11 Lower and upper bound reward function*
3.1.1 Belief State (BS)

Belief-State is a probability distribution over all possible states which gives as much information as the entire action-observation history [69].

BS along with transition and observation probabilities helps to transform the problem from partially-observable to completely observable. An agent needs to update its belief upon taking the action \( a \) and observing \( o \). Since the state is Markovian, maintaining a belief over the states solely requires knowledge of the previous belief state, the action taken, and the current observation.

The operation is denoted \( b' = \tau(b, a, o) \). Below we describe how this belief update is computed. After reaching \( s' \), the agent observes \( o \in \Omega \) with probability \( O(o \mid s', a) \). Let \( b \) be a probability distribution over the state space \( S \). \( b(s) \) denotes the probability that the environment is in state \( s \). Given \( b(s) \), then after taking action \( a \) and observing \( o \),

\[
b'(s') = \eta O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s)
\]

*Equation 12: Belief update*

Where \( \eta = 1/\Pr(o \mid b, a) \) is a normalizing constant with

\[
\Pr(o \mid b, a) = \sum_{s' \in S} O(o \mid s', a) \sum_{s \in S} T(s' \mid s, a) b(s)
\]

*Equation 13: Normalizing constant evaluation*

If we are given a belief state for time \( t \) and we perform an action \( a \) and get observation \( o \) we can compute a new belief state for time \( t+1 \) by simple applying Bayes’ rule and using the model parameters [51].
Bayes’ theorem (conditional probability):

\[ P(A \mid B) = \frac{P(B \mid A) P(A)}{P(B)} \]

*Equation 14 Bayes' theorem*

P(A) and P(B): The probabilities of observing A and B exclusively

P(A | B): The probabilities of observing event A given that B is true

P(B | A): The probabilities of observing event B given that A is true

Now, catching up with the context of [50] [26] and compute the BS using the 6-tuple module of POMDP environment.

The state of the user cannot be directly observed. Therefore, to select good actions, the agent needs to maintain a complete trace of all the observations and actions that have happened so far. This trace is known as a history (in a DM context the trace is called dialogue history). It is formally defined as:

\[ H_{t+1} := \{ A_0, Z_1, ..., Z_t, A_t, Z_{t+1} \} \]

*Equation 15 Dialogue history tracing*

History \( H_{t+1} \) can be summarized via a belief distribution. A belief distribution is exactly the belief state of the agent.

\[ B_{t+1}(s') = P\{S_{t+1} = s' \mid B_0, H_{t+1}\} \]

*Equation 16 Belief distribution*
Assuming the Markov property and using Bayes’ rule, the equation above is transformed into the following equation

\[ B_{t+1}(s') = P\{S_{t+1} = s' | Z_{t+1} = z', Z_t = a, B_t = b\} \]

*Equation 17* Belief distribution re-written assuming with the Markov Property and Bayes’ rule

Formally, let the belief space \( B \) be an infinite set of belief states. A belief state \( b \in B \) is encoded as a \(|S|\)-dimensional column vector \((b_1, b_2, ..., b|S|)^T\), where each element \( b_i = b(s_i) \) is the probability that the current state of the environment is \( s_i \), \( b_i \geq 0, \forall i \in [1, |S|] \), and \( \sum_{i=1}^{\left|\mathbb{S}\right|} b_i = 1 \). Geometrically, a belief state is a point in a \((|S| - 1)\)-dimensional belief simplex.

Concretely, the agent starts with an initial belief state \( B_0 = b_0 \). At time \( t \), the agent’s belief is \( B_t = b \), it selects action \( A_t = a \) and sends this to the environment. The state changes to \( S_{t+1} = s' \). State \( S_{t+1} \) cannot be directly observed and the agent only gets observation \( Z_{t+1} = z \). The agent also receives a reward \( R_t = r \), the value of which depends on the actual values of the state \( s \) and agent’s action \( a \). At this moment the agent needs to update its belief state \( B_{t+1} = b' \) given known values for \( b, a, z \). Starting from Equation above, \( b'(s') \) is computed using the basic laws from the probability theory as follows:
\[ b'(s') = P(s'|z, a, b) \]
\[ = \frac{P(z, s'|a, b)P(s'|a, b)}{P(z|a, b)} \]
\[ = \frac{P(z, s'|a)P(s'|a, b)}{P(z|a, b)} \quad (Z_{t+1} \text{ and } B_t \text{ are independent}) \]
\[ = \frac{\mathcal{P}_{s'z}^a \sum_{s \in S} P(s'|a, b, s)P(s|a, b)}{P(z|a, b)} \]
\[ = \frac{\mathcal{P}_{s'z}^a \sum_{s \in S} P(s'|a, s)P(s|a, b)}{P(z|a, b)} \]
\[ = \frac{\mathcal{P}_{s'z}^a \sum_{s \in S} \mathcal{P}_{ss}^a b(s)}{P(z|a, b)} \]
\[ = \frac{\mathcal{P}_{s'z}^a T_s^a b}{P(z|a, b)}, \]
\[ = \eta \mathcal{P}_{s'z}^a T_s^a b, \]

**Equation 18 Belief state**

where \( T_{s'}^a \) is a \(|S|\)-dimensional row vector:

\[ T_{s'}^a = \left( \mathcal{P}_{s_1s'}^a, \ldots, \mathcal{P}_{s|S|s'}^a \right), \quad |S| \text{ is the number of elements of } S. \]

**Equation 19 Element count of S**

\( \eta = 1/P(z|a, b) \) is a normalizing constant, independent of state \( s' \). The belief state \( b' \) is represented as

\[ b' = \eta W_z^a b \]

**Equation 20 Normalizing constant**

where \( W_z^a \) is a \(|S| \times |S|\) matrix.
3.1.2 Policy

**Definition - Policy in POMDP:** In the planning of optimizing the dialogue length, the interaction system constructs a tree with the possible states and actions. By traversing the tree from the root node to leaves, the optimal plan is computed. In POMDP, as the outcomes of acts taken are stochastic, in other words as the branching factor is high, the tree constructed using conventional planning is very deep.

Policy: Belief-State → Action

\[ \pi: b(s') \rightarrow a \]

*Equation 22 Policy function*

In general, while an MDP policy mapped states to actions, an optimal POMDP policy maps belief states to actions. The focus is that the space of all belief states is continuous. This is a big part of why these problems are hard to solve. Nevertheless, there are algorithms that can work in the space and yield optimal solutions: though they are somewhat complex and computationally inefficient. Once the policy has been computed (optimal or otherwise), using the solution is relatively simple and computationally easy. The way in which one would use a computed policy is to start with belief about where you are in the world. Then continually [70]:

\[ W^a_z = \begin{bmatrix} \mathcal{P}^a_{s_1 \mid z} & \mathcal{P}^a_{s_1 s_1} & \mathcal{P}^a_{s_1 \mid z} & \mathcal{P}^a_{s_1 s_2} & \cdots & \mathcal{P}^a_{s_1 \mid z} & \mathcal{P}^a_{s_1 \mid z} \\ \mathcal{P}^a_{s_2 \mid z} & \mathcal{P}^a_{s_2 s_1} & \mathcal{P}^a_{s_2 \mid z} & \mathcal{P}^a_{s_2 s_2} & \cdots & \mathcal{P}^a_{s_2 \mid z} & \mathcal{P}^a_{s_2 \mid z} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathcal{P}^a_{s_{|S|} \mid z} & \mathcal{P}^a_{s_{|S|} s_1} & \mathcal{P}^a_{s_{|S|} \mid z} & \mathcal{P}^a_{s_{|S|} s_2} & \cdots & \mathcal{P}^a_{s_{|S|} \mid z} & \mathcal{P}^a_{s_{|S|} \mid z} \end{bmatrix} \]

*Equation 21 \(|S| \times |S|\) matrix*
1. Use the policy to select an action for current belief state;
2. Execute the action;
3. Receive an observation;
4. Update the belief state using current belief, action and observation; then
5. Repeat.

Again, holding tight with the flow of the paper (Toward effective dialogue management using partially observable Markov decision processes) [50] [26] to take a thorough look at the policy in the DM context can be shown here. A policy is a function:

\[ \pi(b) \rightarrow a. \]

*Equation 23 Policy function*

where \( b \) is a belief state and \( a \) is the action chosen by the policy \( \pi \). An optimal policy \( \pi^* \) is a policy that maximizes the expected cumulative reward:

\[ \pi^* = \arg\max_\pi E \left[ \sum_{t=0}^{\infty} \gamma^t R_t \right] \]

*Equation 24 Optimal policy*

where \( R_t \) is the reward when the agent follows policy \( \pi \). We define value functions \( V_n: B \rightarrow \mathbb{R} \). \( V_n(b) \) is the maximum expected a cumulative reward when the agent has \( n \) remaining steps to go. Its associated policy is denoted by \( \pi_n \). When the agent has only one remaining step to go (i.e. \( n = 1 \)), all it can do is to select an action and send it to the environment, we have:
\[ V_1(b) = \max_{a \in A} \sum_{s \in S} R(s, a) b(s) \]

\[ = \max_{a \in A} r_a b, \]

*Equation 25 Maximum expected cumulative reward*

where \( r_a \) is a row vector,

\[ r_a = \left( R_{s_1}^a, \ldots, R_{s_{|S|}}^a \right) \]

*Equation 26 Row vector*

3.1.3 Shortcomings of POMDP

The first shortcoming of POMDP is that it contains an approximation: we are trying to find representations of the belief which are rich enough to allow good control, but which are also sufficiently parsimonious to make the planning problem tractable.

The second disadvantage is a technical one: while making a nonlinear transformation of the belief space, POMDP planning algorithms which assume a convex value function will no longer work.

For non-trivial modelling, the state/action/observation spaces quickly grow, this is a result of requiring a complete enumeration of all possible states. The discrete, uniform time elapsing between each action-observation pair is not always a good assumption. The assumption that the model will not change over time (non-stationary process) does not always apply.
3.2 Previous works

The primary work that has been carried out in this thesis is to improve policy and to make the system more accurate as measured by dialogue length, through the improved utilization of emotions. There is a great amount of work has already been carried out in this field which is discussed in the following sub-sections.

3.2.1 Emotion detection from speech

A famous and reliable application is known as Vokaturi [71] the field of SDS domain, was conceptualized based on the PRAAT program [72]. PRAAT, a computer program which can analyze, synthesize, and manipulate speech, works as in shown below.

Get a sound by recording
Perform the spectral analysis
Work on the pitch analysis
Carry out the formant analysis including the formant contours
Find the intensity analysis
Handle jitter, shimmer and voice breaks

These steps are followed by the generation of speech synthesis, labelling and segmentation, speech manipulation, etc. and based on parabolic interpolation, sound pressure calibration, counting the spectrum of the frame, the discovery of auto-correlation in a pitch helps in recognizing the emotion from the user-speaker’s speech interaction.

This work is imported because this kind of work can be helpful in a speech-based dialogue management system where an agent interacts with a user on a turn by turn basis. The main
purpose of a speech-based dialogue system is to provide an interface between the user and the agent in order for the agent to understand the need of the user so that adequate services can be provided [73]. The dialogue system, therefore, needs to process the user’s spoken input and be capable of recovering from errors. A speech-based dialogue system typically includes the components of input, output, and knowledge, plus the central component of dialogue management. Dialogue management simulates the task model in the specific domain of SDS. It also processes semantic inputs from fusion and decides what the agent should do in response to the user input.

Kindly refer to Appendix D: Textual and Speech-based emotions discovery from a user’s interaction, for the UI example on speech-based communication and how the ECA finds an emotion from the speech recognition algorithm. This kind of technology can be helpful to build more advanced and intelligent agents to understand human emotions. There is a good amount of work available in terms of detecting user’s intention based on speech [74] [75] [76] [77] [78] [79] and facial expressions provided via video [80] [75] or pictures [81], to know the emotion of the user.

3.2.2 The Contextual Control Model (COCOM)

The COCOM model [82] is based upon cognitive modes. This model suggests that the system needs to decide what actions to take according to the context of the situation. There are four modes of operations strategic, tactical, opportunistic, scrambled. Each control mode has its own characteristics and type performance and the mode of team behaviour varies in terms of the degree of planning.

- Strategic mode - this mode concentrates on long-term planning using a global view and has a higher level of control. The amount of information sought, and user-system coordination required is expected to be extensive.
• Tactical mode - The system and the amount of available information is beyond what is immediately observable but may be limited to what the routine procedure requires.

• Opportunistic mode - The next action is predictable depending upon a current situation where available information is inadequate due to the less effective planning, limited time and incomplete understanding of context.

• Scrambled mode - The choice of next action is basically irrational or completely unpredictable. The type of performance is thus paradoxically characterized by the lack or absence of any control.

Figure 14 COCOM- Contextual Control Model
3.3 Belief state History and Trend Analysis

3.3.1 POMDP-based Dialogue management

Based on the shortcomings of the POMDP approach as shown in 3.1.345, the approach of belief state history and trend analysis approach helps in working with Belief State by using on the HISTORY of belief states and the dynamics of Belief State (BS) [73] [83]. The consequence is inflexibility for human-robot interaction as in the FSM-based approach, incapable of handling any ambiguity as in the frame/Bayes/MDP-based approaches, and insufficiency in dealing with uncertainties as in the POMDP-based approach. To overcome the shortcomings of handling the uncertainty while retaining the advantages of the current POMDP-based approach, this paper proposes a modified planning strategy as illustrated below.

\[ \pi_{\text{new}} : \mathcal{I}_{k-1}' \cup \mathcal{I}_k \rightarrow U \]

*Equation 27* Modified planning strategy

**Definition- Belief State HISTORY (BSH):** As per Equation 27, both \( I_k \) and \( I_{k-1}' \) are still in the form of a belief state, and updating still uses the existing POMDP model. However, the addition of \( I_{k-1}' \) in the modified approach, this state, introduces an important element to dialogue management, i.e., the history of belief state or the dynamics of belief state since the inception of the interaction for current the conversation. Although the history information of observations and actions is not maintained explicitly in \( I_{k-1}' \), the union \( I_k \) and \( I_{k-1}' \) in Equation 27 diminish the negative effect of the Markov Assumption and allows POMDP-based dialogue management to plan for actions with not only the current belief state but also the updated history before reaching the current state.
The algorithm for the new approach is shown with a flow chart in Figure 15. After an initial greeting, the system always updates the belief state using the previous belief state, current action, and the latest observation from the user. At each stage of dialogue, the new approach uses the domain knowledge and constraint database to help to validate the change of belief state. A failed validation results in a roll-back of belief state to the previous state. For this approach, the DM architecture has been updated from Figure 2 to as in Figure 16.
The work of storing the BS and updating with each state will be taken into a consideration in my work, to keep the track of BS History.

3.4 Trend analysis using Belief state

**Definition- Trend Analysis**: Trend analysis is the widespread practice of collecting information and attempting to spot a pattern. Although trend analysis is often used to predict future events, it could be used to estimate uncertain events in the past using the different states which we have stored inside the POMDP model. Trend analysis often refers to techniques for extracting an underlying pattern of behaviour in a time series which would otherwise be partly or nearly completely hidden by noise. If the trend can be assumed to be linear, trend analysis can be undertaken within a formal regression analysis, as described in trend estimation [84].
In general trend analysis is performed on historical data and time series data to predict the subject of interests for the future. The different approaches to trend analysis are as follows [1],

- **Sampling** - the historical data is split into training and testing datasets. The training dataset is used to develop a predictor model and its accuracy is determined using the testing dataset. Random sampling and reservoir-based sampling are example sampling methods.

- **Histogram** - the trend is analyzed by constructing a histogram from the historical data by dividing the entire range of values into a series of intervals and then count how many values fall in each interval. Equi-depth and V-optimal are example histogram approaches.

- **Sketches** - The frequency distribution of historical data is summarized by using hash functions. Count sketches and Count-Min sketches are example techniques.

- **Wavelets** - Mathematical transformations are applied to transform the data into a set of wavelet coefficients representing a different level of granularity to analyze the trend.

**Example** - A **trend analysis** is a method of analysis that allows traders to predict what will happen with a stock in the future. Trend analysis is based on historical data about the stock's performance given the overall trends of the market and indicators within the market. Trend analysis takes into account historical data points for a stock and, controlling for other factors like the general changes in the sector, market conditions, competition for similar stocks, it allows traders to forecast short, intermediate, and long-term possibilities for the stock [85]. When wavelets are used for trend analysis they are used to enhance the trend by removing the possibilities of hidden noise, by approaching the exact and accurate trend over time-series data.
3.5 POMDP using Belief state History information

The user interacts with the agent by providing the observation and the agent responds to the user by performing an action. This work has been carried in this [1] thesis as in Figure 17.

- State Estimator (SE)- It receives the observation as input from the user. SE computes the observation probabilities through NLP and updates the BS value $B(s')$
- Belief History Information Storage- The BS $B(s')$ value computed in the SE is stored in the belief history information $b_{\text{hist}}$ storage module
- Trend Analysis- It receives the history of belief information $b_{\text{hist}}$ as input. Number of sharp variation points $N_{cp}$ is obtained by performing DWT on $b_{\text{hist}}$
- Knowledge level selector- It receives knowledge level $k$ of the user and BS $B(s')$ value as input. In the policy selector, a different set of policies are defined for users at the different knowledge level. Policy $\pi_k$ is selected based on the value of $B(s')$
- Make Action- It is within the Policy selector module which receives the policy $\pi_k$ to execute. Each policy in POMDP is mapped from $B(s')$ to actions. In Make an Action module, action $a$ is chosen through COCOM modes
Figure 17 Architecture diagram of the proposed framework using BSH information

The sequence of computations that are happening for every interaction between user and agent are illustrated in the Figure 18.
This thesis will be inspired by the work of this approach in [1] as a groundwork to develop my model which focuses on improving the policy by improving its accuracy.

3.5.1 POMDP in a programming language

As discussed earlier, the POMDP environmental setup is domain-independent, we include and exclude the sets of the tuple based on our requirements. They have defined primitive 10 tuple rule as, \( <D, R, \Omega, T, O, I, C, D_U, R_U, F> \).

- D: \( D^1 \times \ldots \times D^n \), is a set of domain states
- R: is a set of range states
- Ω: is a set of observation
- T: D x R → [0, 1] is a transition function
- O: R x Ω → [0, 1] is a observation function
- I: is an n-dimensional indicator vector indicating which of the D^i are observable
- C : D → R is a cost function
- D_U: D_U^1 x ... x D_U^n is a set of user domain states
- R_U: is a set of user range states
- F: D_U → R_U is a user function

3.6 Wavelet Theory

**Definition- Wavelet Transform**: In mathematics, a wavelet series is a representation of a square-integrable (real-or complex-valued) function by certain orthonormal series generated by a wavelet [86].

**Definition- Continuous Wavelet Transform (CWT)**: In mathematics, the continuous wavelet transform (CWT) is a formal (i.e., non-numerical) tool that provides an overcomplete representation of a signal by letting the translation and scale parameter of the wavelets vary continuously [87].

**Definition- Discrete Wavelet Transform (DWT)**: In numerical analysis and functional analysis, a discrete wavelet transform (DWT) is any wavelet transform for which the wavelets are discretely sampled. As with other wavelet transforms, a key advantage it has over Fourier transforms is a temporal resolution: it captures both frequency and location information (location in time) [88].

Why DWT? - The DWT decomposes the signal to discrete time and provides enough information for both analysis and synthesis of the original signal. It helps in removing some
of the samples of the signal by reducing the sampling rate to reduce the noise and to gain better accuracy. The time complexity is O(n).

The DWT from the perspective of 1-D and 2-D can be implemented using Haar wavelets available in the JWave open source project.

The DWT considers the input from the BSH database that the function assumes that input is of length $2^n$ where $n>1$. Then the next current length becomes the working area of the output array. The length starts at the half of the array size and every iteration is halved until the length becomes 1. Then we swap the arrays to perform the next iteration.

3.6.1 Sharp Variation points

The Belief State History (BSH) shows different fluctuation characteristics at different interactions for different users. The scales are distinguished in the wavelet transform, are based on zero-crossing points from wavelet transformation which is known as Sharp variation points [1]. We can identify the maximum sharp variation points and the minimum variation points based on the signal.

3.7 Fuzzy Logic

**Definition- Fuzzy logic:** Fuzzy logic is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1. It is employed to handle the concept of partial truth, where the truth value may range between completely true and completely false. By contrast, in Boolean logic, the truth values of variables may only be the integer values 0 or 1 [89].

Why Fuzzy logic?

- A fuzzy logic system can be defined as the non-linear mapping of an input dataset to a scalar output data. It consists of 4 components such as,
The rules for the fuzzy system for this thesis can be organized as written below.

<table>
<thead>
<tr>
<th>FAMM (Fuzzy Associative Memory Matrix)</th>
<th>Fuzzy RULE base selection based on Trend Analysis performed on Belief-state history using COCOM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sentiment Analysis Reward</strong></td>
<td>Strategic</td>
</tr>
<tr>
<td>Negative</td>
<td>Disgust</td>
</tr>
<tr>
<td>Neutral</td>
<td>Fear</td>
</tr>
<tr>
<td>Positive</td>
<td>Happy</td>
</tr>
</tbody>
</table>

*Figure 19 Fuzzy Logic System*
The fuzzy-based approach was used to conduct the usability study by Kaur [2] as shown in Figure 20.

Figure 20 Fuzzy logic system based ECA expression based POMDP model

3.8 Shortcomings of previous approaches (Significant Problems)

The general scenario of shortcomings in recently existing work can be stated as,

- POMDP refrains to capture the history of actions taken and observations made
- Dialogue length can be optimized
- POMDP hasn’t been tested on a huge scale
- The policy is not accurate enough which means belief state tracing is not performed well
• The connection between different belief states and its history is not capable enough to be merged as an agent will count the different approach of same communication as two different DM context and then dialogue length will not be optimized

• Most of the models do not offer the NLP in the dialogue exchange, so that means deep learning or machine learning or artificial intelligence should be able to incorporate POMDPs

• If NLP is taken into consideration, yet patterns are found after data mining that can optimize the dialogue length and can help in improving the accuracy with better training corpus.

3.9 The new approach – Thesis contribution

The main goal of this thesis is to improve the policy π with the help of belief state history. During the interaction, if the agent is capable enough to find the exact intention from the user import then it is an achievement of the proposed model as the policy will be improved using RL techniques and sentiment analysis to improve ECA emotions. When the aim is to reach the goal of the user, the solution offered from an agent to the user, the dialogue length is measured that how long it took for a DM context to reach a goal and based on that belief states will be stored and used as the history. Chapter 4: Proposed method is about the discussion and the detailed description to improve the policy using the user import with two support systems such as intention discovery and emotion detection to optimize the dialogue length by considering the rewards from the RL technique.

3.10 Thesis statement

The components used to achieve the aim of the thesis, ECA as an agent and real person as a user to conduct HCI-based contextual dialogue interaction, using the DM domain module of POMDP with the belief state history to find the sharp variation points, which restores the
important information of belief states through DWT to achieve the policy to be more accurate by improving it than currently existing models with an optimized dialogue length using RL technique called Q-learning and sentiment analysis to improve the intention discovery.

3.11 Structure of the thesis report

The following thesis report is listed as follows,

- Chapter 4: Proposed method discusses the existing POMDP model work with RL technique and proposed methods
- Chapter 5: Implementation and Experiment setup depicts the environmental setup and the experiments
- Chapter 6: Simulation and Results shows the simulation and the result
- Chapter 7: Conclusion and Future work is the conclusion and discusses possible of future work
Chapter 4: Proposed method

4.1 Overview

This chapter discusses the architecture, a method to help POMDP improve the accuracy of policy, sentiment analysis rewards, Q-values as RL rewards.

4.2 Sentiment Analysis

Definition- Sentiment Analysis: The process of computationally identifying and categorizing emotion(s) (sentiments) expressed from a piece of text. It aids in the detection of emotion.

Sentiment analysis helps in achieving end-to-end task completion [90] and it has wide appeal as providing information about the subjective dimension of texts which can be regarded as a classification technique, either binary (polarity classification into positive/ negative) or a multi-class categorization (negative/ neutral/ positive). Most approaches use sentiment lexicons as a component (sometimes only the component). Lexicons can either be general purpose or extracted from a suitable corpus, such as movie reviews with explicit ranking information [91] [92].

4.3 Reinforcement Learning

The goal of Artificial Intelligence is to produce fully autonomous agents that interact with their users (environments) to learn optimal behaviours, improving over time through trial-and-error [93]. A principle mathematical framework for experience-driven autonomous learning is known as Reinforcement Learning [94]. There have been several limitations of several traditional RL approach like memory complexity, computational complexity, scalability [95]. The optimization of statistical dialogue managers using RL methods is an active and promising area of research. In contrast with the traditional discrete action domains like the Atari game (to predict the next move, from self-learning of previous states and moves, part of reinforcement learning), playing has much focus on deep RL research. The
dialogue manager has a broader variety of system dialogue actions available, each associated with distinct semantics. In a task-oriented dialogue, on the other hand, usually consists of fewer turns and each system action can crucially alter the direction or length of the dialogue [96].

The limitations were outperformed with the rise of the deep learning using powerful function approximation and representation learning properties of DNN (Deep Neural Network). The true essence of RL is learning through interaction [93]. The RL agent interacts with its environment and upon the observation, consequences of actions can learn to alter its own behaviour in response to rewards which are received.

The perceptron-action-learning loop, at time \( t \), the agent receives state \( s_t \) from the environment, the agent uses its policy to choose an action \( a_t \). After action execution, environment transitions a step, providing the next step \( s_{t+1} \) as well as the feedback in the form of a reward \( r_{t+1} \). The agent uses knowledge of state transitions, of the form \((s_t, a_t, s_{t+1}, r_{t+1})\), to learn and improve its policy.

![Reinforcement Learning flow diagram example](image)

*Figure 21 Reinforcement Learning flow diagram example*
The dynamics of trial-and-error-learning has its roots in behaviourist psychology, being one of the main foundations of RL [94]. The best sequence of actions is determined by the rewards provided by the environment. Every time the environment transitions to a new state, it also provides a scalar reward $r_{t+1}$ to the agent as feedback. The goal of the agent is to learn policy $\pi$, which maximizes the expected return as a reward. For any given state, a policy returns an action to perform an optimal policy (any policy), that maximizes the expected return in the environment, RL aims to solve the problem of optimal control, where the agent needs to learn about the consequences of actions by trial-and-error [93]. To generate the rewards, the function uses the Bellman equation and takes two inputs, state and action.

$$R = \sum_{t=0}^{T-1} \gamma^t r_t(b_t, a_t)$$

*Equation 28 Deep Reinforcement Learning*

where $R$ is the total return, $t$ is the turn, $\gamma$ is the discount factor, $r$ is the reward, $b$ stands for the dialogue state, and $a$ stand for the action.

4.3.1 Q-Learning

Q-learning is an RL technique used in machine learning. “Q” names the function that returns the reward, used to provide the reinforcement and can be said to stand for the “quality” of an action taken in a given state [97]. Q-Learning can identify an optimal action-selection policy for given infinite exploration time and a partly-random policy. Q-Learning finds a policy that is optimal in the sense that it maximizes the expected value of the total reward over all successive steps, starting from the current state [98]. It is proven that, when the model is trained sufficiently under any policy, the algorithm converges with probability 1 to a close approximation of the action-value function for any target policy. Q-learning learns the optimal policy even when actions are selected according to a random policy [99].
Deep Q-Learning, as a part of deep RL, is the RL method using a deep neural network as a Q-value function approximator [100]. A neural network is used to approximate the Q-values in a decision process [101], where Q-values are parameterized by the belief and the action; belief state history. These modified Q-values can be learned by a neural network. This is an iterative process of updating the values where the Q-function gives the better and better approximations by continuously updating the Q-values in the table. Q-Learning is a value-based reinforcement learning algorithm to find the optimal action-selection policy to maximize the value function Q. It helps maximize the expected reward by selecting the best of all possible actions. \( Q(s, a) \) returns the expected future reward of that action at that particular state, using epsilon greedy strategy.

\[
Q(s, a) = Q(s, a) + \alpha (r + \gamma \max_{a'} Q(s', a') - Q(s, a))
\]

*Equation 29 Deep Q-Learning (Deep Reinforcement Learning)*

where the first \( Q(s, a) \) stands for the new Q value on the left-hand side of the equation, which we need to calculate, the second \( Q(s, a) \) stands for the old Q value on the right-hand side of the equation, \( \alpha \) stands for the learning rate (hyperparameter, initialized with the value 0.001; \( 0 < \alpha \leq 1 \); the value is decided based on the extent that how likely the newly acquired information overrides the old information), \( r \) is the reward, \( \gamma \) is the discount factor (to maximize the future sum of rewards and value initialized at 1), the \( \max_{a'} Q(s', a') - Q(s, a) \) suggest the estimation of optimal future value, and \( (r + \gamma \max_{a'} Q(s', a') - Q(s, a)) \) represents the learned value.

### 4.3.2 Q-Learning Algorithm

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Initialize ( Q(s, a) )</td>
</tr>
<tr>
<td>2</td>
<td>Repeat (for each interaction):</td>
</tr>
<tr>
<td></td>
<td>Initialize ( s )</td>
</tr>
</tbody>
</table>
Repeat (for each step of interaction):

Choose \( a \) from \( s \) using policy derived from \( Q \)

Take action \( a \), observe \( r, s' \)

\[
Q(s, a) = Q(s, a) + \alpha (r + \gamma \max_{a'} Q(s', a') - Q(s, a))
\]

\( S \leftarrow s' \)

Until \( s \) is terminal

| Table 5 Algorithm for Q-Learning |

where [99],

\( \alpha \): the learning rate set between 0 and 1. Setting it to 0 means that Q-values are never updated hence nothing is learned. Setting a high value such as 0.9 means that learning can occur quickly.

\( \gamma \): the discount factor also set between 0 and 1. This represents that, the future rewards are worth less than immediate rewards. The discount factor needs to be less than 0 for the algorithm to converge.

\( \max_{a'} \): the maximum reward that is, attainable in the state following the current one.

4.4 POMDP with updated BS history to improve POLICIES

Considering what is stated by the thesis work [1], 7 tuples are used such as, states, actions, observations, transition, observation probability, reward functions, discount factor, etc. this approach has stated the policy selection based on knowledge level selection among different modes, Expert; Professional; Amateur; Novice, the policy selector module get the mode as input and return respective policy as output. Policies are mapped from belief-state values to action. If we receive the minimum sharp points and the maximum sharp point as high as possible for the EXPERT level that will be counted as the optimized length of dialogue and higher accuracy of policy. But the work shows that over a few datasets, the results are not so good as NOVICE level has a higher minimum and maximum sharp points, so that can be
worked out. The following architecture in Figure 22 shows how knowledge level threshold are selected using BSH.

![Knowledge level selector architecture](image)

*Figure 22 Knowledge level selector architecture*

4.5 System Architecture

*The block reference is given in the following chapters:* Chapter 5: Implementation and Experiment setup *and* Chapter 6: Simulation and Results.

<table>
<thead>
<tr>
<th>Block No.</th>
<th>Module name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>State Estimator</td>
</tr>
<tr>
<td>B</td>
<td>Belief State History</td>
</tr>
<tr>
<td>C</td>
<td>Trend Analysis</td>
</tr>
<tr>
<td>D</td>
<td>Knowledge level selector</td>
</tr>
<tr>
<td>E</td>
<td>Sentiment Analysis</td>
</tr>
<tr>
<td>F</td>
<td>Policy Selector</td>
</tr>
<tr>
<td>G</td>
<td>Fuzzy logic System</td>
</tr>
<tr>
<td>H</td>
<td>The facial expression on ECA</td>
</tr>
<tr>
<td>I</td>
<td>3-D model</td>
</tr>
<tr>
<td>J</td>
<td>Feedback reward</td>
</tr>
<tr>
<td>K</td>
<td>User</td>
</tr>
</tbody>
</table>

*Table 6 Different modules from the Thesis Architecture*
4.6 Discussion- Comparison and Contribution

The comparison with the previously existing work and details about the contribution of the thesis is discussed in Table 7.

<table>
<thead>
<tr>
<th>Contributors</th>
<th>Discussion of the contribution</th>
</tr>
</thead>
</table>
| **Bui, 2008** [50]            | • Worked on traditional POMDP model represents belief updating finding; finding the optimal policy  
• Value-iteration based POMDP is used to compute optimal or near-optimal policy  
• The effective dialogue system is used to issue two inputs of observations of action and state                                                                 |
| **Gasic M., et. al., 2015** [58] | • Hierarchical distributed dialogue architecture in which policies are organized in a class hierarchy aligned to an underlying knowledge graph; Gaussian process-based RL is used to represent within the framework, generic policies can be constructed which provides acceptable user performance |
| **Gasic M., et. al., 2016** [57] | • Data-driven machine learning methods have been applied to dialogue modelling and the results achieved for limited-domain applications are comparable to outperform the traditional approaches  
• Method based on Gaussian processes are particularly effective as they enable good models to be estimated from limited training data  
• Gaussian process RL is an elegant framework that naturally supports a range of methods including prior knowledge |
| **Igl M., et. al., 2018** [41]  | • The model, DVRL (Deep Variational Reinforcement Learning) introduces an inductive bias that allows an agent to learn a generative model of the environment and perform inference in the model to effectively aggregate the available information  
• This method solves POMDPs for given only a stream of observations, without knowledge of the latent state space or the |

69
### Mulpuri, 2016 (UoW) [1]
- Automation of requirement elicitation in the software product line
- Decision-making algorithm for automation is combined with POMDP model; trend analysis on belief-state history to anticipate user knowledge level
- Knowledge level used in addressing policy selection to perform the appropriate action

### Kaur, 2016 (UoW) [2]
- ECA with emotions offers a better understanding of user import
- The fuzzy logic system used to generate facial expressions
- A usability study was conducted to improve the user interface
- Based on user’s opinions, user satisfaction has been improved

### Ruturaj R. Raval, 2019 (UoW)
- Extended work of Mulpuri and Kaur
- Proposed architecture to improve user import using Sentiment Analysis to improve intention discovery; decreasing dialogue length
- Reinforcement Learning is implemented using Q-learning technique to acquire optimal policy to reach the user’s goal
- Sentiment Analysis feeds Negative, Neutral, Positive values to the fuzzy logic system to improve ECA emotion and RL operates on feedback provision from user to agent based on trial-and-error action learning to update the knowledge

**Table 7 Discussion- Contribution and Comparison**

#### 4.7 Algorithm

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>isGoalState ← false</td>
</tr>
<tr>
<td>2</td>
<td>belief ← 1</td>
</tr>
<tr>
<td>3</td>
<td>CREATE empty LIST bhist</td>
</tr>
<tr>
<td>4</td>
<td>ADD belief to LIST bhist</td>
</tr>
<tr>
<td>5</td>
<td>WHILE isGoalState NOT EQUAL true</td>
</tr>
<tr>
<td>6</td>
<td>input ← READ(observation)</td>
</tr>
<tr>
<td>7</td>
<td>IF input EQUAL ‘exit’ THEN</td>
</tr>
<tr>
<td>8</td>
<td>isGoalState ← true</td>
</tr>
<tr>
<td>9</td>
<td>ELSE</td>
</tr>
<tr>
<td>10</td>
<td>b(s’) ← StateEstimator(input, belief)</td>
</tr>
<tr>
<td>11</td>
<td>Rewards (R(r_t+1)) ← StateEstimator(input, belief)</td>
</tr>
<tr>
<td>12</td>
<td>π{m} ← Rewards (R(r_t+1))</td>
</tr>
</tbody>
</table>
### 4.8 Detailed Algorithm

<table>
<thead>
<tr>
<th>Initialization</th>
<th>The system initializes with the belief value as 1 and new list is created to store the belief values as the BSH record.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>isGoalState ← false</td>
</tr>
<tr>
<td>2</td>
<td>belief ← 1</td>
</tr>
<tr>
<td>3</td>
<td>CREATE empty LIST bhist</td>
</tr>
<tr>
<td>4</td>
<td>ADD belief to LIST bhist</td>
</tr>
<tr>
<td>5</td>
<td>WHILE isGoalState NOT EQUAL true</td>
</tr>
<tr>
<td>6</td>
<td>input ← READ(observation)</td>
</tr>
<tr>
<td>7</td>
<td>IF input EQUAL ‘exit’ THEN</td>
</tr>
<tr>
<td>8</td>
<td>isGoalState ← true</td>
</tr>
<tr>
<td>9</td>
<td>ELSE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State Estimator</th>
<th>This module analyses the observation as text using NLP (Natural Language Processing) and compared with the services.</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>StateEstimator(input, belief) ← USER_FEEDBACK</td>
</tr>
<tr>
<td>11</td>
<td>b(s’) ← StateEstimator(input, belief)</td>
</tr>
<tr>
<td>12</td>
<td>tokens ← NLP(input)</td>
</tr>
<tr>
<td>13</td>
<td>s’ ← MATCH_SERVICES(tokens)</td>
</tr>
<tr>
<td>14</td>
<td>b(s’) ← Pr(s’</td>
</tr>
<tr>
<td>15</td>
<td>return b(s’)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q-Learning Feedback</th>
<th>The Q-value rewards are fed in based on the Action-Perceptron-Learning loop.</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>Rewards (R(r{t+1})) ← StateEstimator(input, belief)</td>
</tr>
<tr>
<td>17</td>
<td>π{m} ← Rewards (R(r{t+1}))</td>
</tr>
<tr>
<td>18</td>
<td>ADD b(s’) to LIST bhist</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trend Analysis</th>
<th>This module captures the BSH values and the provides the output as variation points as in Ncp.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>19</td>
<td>Ncp ← TrendAnalysis(bhist)</td>
</tr>
<tr>
<td>20</td>
<td>Ncp ← DWT(bhist)</td>
</tr>
<tr>
<td>21</td>
<td>return Ncp</td>
</tr>
</tbody>
</table>

This module accepts the Ncp as input and returns the k values as expert or professional or amateur or novice.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>k ← KnowledgeLevelSelector(Ncp)</td>
</tr>
<tr>
<td>23</td>
<td>expertThreshold ← READ_FROM_TRAINED_MODEL</td>
</tr>
<tr>
<td>24</td>
<td>professionalThreshold ← READ_FROM_TRAINED_MODEL</td>
</tr>
<tr>
<td>25</td>
<td>amateurThreshold ← READ_FROM_TRAINED_MODEL</td>
</tr>
<tr>
<td>26</td>
<td>noviceThreshold ← READ_FROM_TRAINED_MODEL</td>
</tr>
<tr>
<td>27</td>
<td>IF Ncp &lt; expertThreshold THEN</td>
</tr>
<tr>
<td>28</td>
<td>k ← ‘expert’</td>
</tr>
<tr>
<td>29</td>
<td>ELSE IF Ncp &gt;= expertThreshold AND Ncp &lt; professionalThreshold</td>
</tr>
<tr>
<td>30</td>
<td>k ← ‘professional’</td>
</tr>
<tr>
<td>31</td>
<td>ELSE IF Ncp &gt;= professionalThreshold AND Ncp &lt; amateurThreshold</td>
</tr>
<tr>
<td>32</td>
<td>k ← ‘amateur’</td>
</tr>
<tr>
<td>33</td>
<td>ELSE</td>
</tr>
<tr>
<td>34</td>
<td>k ← ‘novice’</td>
</tr>
<tr>
<td>35</td>
<td>ENDIF</td>
</tr>
<tr>
<td>36</td>
<td>return k</td>
</tr>
</tbody>
</table>

This module returns the rewards based on NLP sentiment analysis.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>R ← SentimentAnalysis{Neg, Neu, Pos}</td>
</tr>
</tbody>
</table>

This module accepts rewards from sentiment analysis rewards and Ncp to produce crisp output to showcase as an expression on a 3D model.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>Ncp ← {ST, T, O, SC}</td>
</tr>
<tr>
<td>39</td>
<td>F_Ncp ← FUZZIFICATION (Ncp)</td>
</tr>
<tr>
<td>40</td>
<td>F_R ← FUZZIFICATION (R)</td>
</tr>
<tr>
<td>41</td>
<td>FAMM ← LOAD_RULE_BASE ()</td>
</tr>
<tr>
<td>42</td>
<td>W ← INFERENCE (F_Ncp, F_R, FAMM)</td>
</tr>
<tr>
<td>43</td>
<td>output ← DEFUZZIFICATION (W, FAMM)</td>
</tr>
<tr>
<td>44</td>
<td>GENERATE_FACIAL_EXPRESSION (output)</td>
</tr>
</tbody>
</table>

This module accepts the input from knowledge level selectors and returns the respective policy based on action taken.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>π{m} ← PolicySelector(k)</td>
</tr>
<tr>
<td>46</td>
<td>CASE k OF</td>
</tr>
<tr>
<td>47</td>
<td>expert: return πexpert</td>
</tr>
<tr>
<td>48</td>
<td>professional: return πprofessional</td>
</tr>
</tbody>
</table>
This models makes action based on the input from the policy selection using COCOM modes.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Make Action</td>
<td>This models makes action based on the input from the policy selection using COCOM modes.</td>
</tr>
<tr>
<td>52</td>
<td>action ← MakeAction(π{m}, b(s'))</td>
</tr>
<tr>
<td>53</td>
<td>action ← GET(Action from Transition of state s to s')</td>
</tr>
<tr>
<td>54</td>
<td>return action</td>
</tr>
<tr>
<td>55</td>
<td>belief ← b(s')</td>
</tr>
<tr>
<td>56</td>
<td>PRINT action</td>
</tr>
</tbody>
</table>

End of one loop interaction.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>End</td>
<td>End of one loop interaction.</td>
</tr>
<tr>
<td>57</td>
<td>ENDIF</td>
</tr>
<tr>
<td>58</td>
<td>END WHILE</td>
</tr>
</tbody>
</table>

Table 9 Table of the detailed algorithm

4.9 Time Complexity

The time complexity is calculated based on the programmatical implementation. The State Estimator tokenizes the words in a nested loop so that the tokens can be identified hence making a time complexity as O(n²). The service matching occurs during the tokenizing only, to match the similar words from the corpus, if available, henceforth O(n*m). The Q-Learning feedback and the Sentiment analysis offers rewards between 0 to 1, making them O(1) time complex. The trend analysis works with the length of the belief state history; thus the nested looping is implemented while going through the length and dividing the length in half so the time complexity is O(n²). The sharp variation points detection runs on all the values individually making the time complexity as O(n). the knowledge level selector, fuzzy logic system, policy selector, making an action have the switch statements, so that only one of the conditions will be true makes it O(1) running time complex.

<table>
<thead>
<tr>
<th>Module</th>
<th>Time complexity</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>State estimator</td>
<td>Tokenizing- O(n²)</td>
<td>n = input string length</td>
</tr>
<tr>
<td></td>
<td>Services matching- O(n*m)</td>
<td>n = input string length</td>
</tr>
<tr>
<td>Function</td>
<td>Time Complexity</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Q-Learning feedback</td>
<td>O(1)</td>
<td>m = number of services</td>
</tr>
<tr>
<td>Trend Analysis</td>
<td>Wavelet Transform- O(n^2)</td>
<td>n = belief history length</td>
</tr>
<tr>
<td>Knowledge level selector</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentiment analysis</td>
<td>O(1)</td>
<td></td>
</tr>
<tr>
<td>Fuzzy logic system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Policy selector</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Make action</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 10 Time Complexity of the algorithm*
Chapter 5: Implementation and Experiment setup

For the implementation of the POMDP approach to detect emotion; to work with the belief state history; and policy to improve the intentions from user input, the following experiments were performed on an Android device in addition with python environment.

5.1 Software information

The comprehensive list of all the libraries; the software; the language(s) and external software(s) used to design an experiment for the DM model is given in a descriptive manner in Table 11.

<table>
<thead>
<tr>
<th>Block reference from Section 4.5</th>
<th>Item</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>Project name</td>
<td>Avatar Interaction</td>
</tr>
<tr>
<td></td>
<td>OS</td>
<td>Android &amp; Windows</td>
</tr>
<tr>
<td>A, B, C, F, G, K</td>
<td>Languages</td>
<td>Java, XML</td>
</tr>
<tr>
<td>D, E, F, J, K</td>
<td></td>
<td>Python</td>
</tr>
<tr>
<td>A, D, H</td>
<td>Technologies</td>
<td>Json</td>
</tr>
<tr>
<td>A, B, C, F, G, K</td>
<td>IDEs</td>
<td>Android Studio 3.1, Notepad++</td>
</tr>
<tr>
<td>D, E, F, J, K</td>
<td></td>
<td>IDLE</td>
</tr>
<tr>
<td>G</td>
<td></td>
<td>R Studio</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Android:design, junit, Android:recycleview,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Android:cardview, io :commons, jitpack.io,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>android :buid :gradle</td>
</tr>
<tr>
<td>H, I, K</td>
<td>Libraries : Others</td>
<td>Debatty :java_string_similarity :0.10, jwnl.jar :1.3.3, google:protobuf:3.4.0</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>I</td>
<td>Libraries: 3D models</td>
<td>jpcj_ae.jar</td>
</tr>
<tr>
<td>H, I</td>
<td>Libraries: Speech recognized emotion detection</td>
<td>Vokaturi:1.03</td>
</tr>
<tr>
<td>H, I</td>
<td>3D model generator software</td>
<td>Facegen</td>
</tr>
<tr>
<td>A, B, C, F, I, K</td>
<td>Simulator</td>
<td>Android studio ADB, Genymotion &amp; Terminal</td>
</tr>
<tr>
<td>B</td>
<td>Database</td>
<td>SqliteDB, CouchDB</td>
</tr>
<tr>
<td>E</td>
<td>Sentiment Analysis</td>
<td>NLTK (Natural Language Tool Kit) for Python</td>
</tr>
<tr>
<td>A, E</td>
<td>Word(s) embedding</td>
<td>Word2vec (Pretrained model- GloVe)</td>
</tr>
<tr>
<td>A, D</td>
<td>Training dataset</td>
<td>Emotions: 7500+ sentences, 1500+ words to train with emotions &amp; Dialogues: ConvAI2, VisDial (354M:17M:10M)</td>
</tr>
<tr>
<td>J</td>
<td>Q-Learning</td>
<td>Tensorflow, PyTorch</td>
</tr>
<tr>
<td>D</td>
<td>Wizard-of-Oz</td>
<td>WoZ: Relational Strategies in Customer Service Dataset</td>
</tr>
<tr>
<td>G</td>
<td>R Studio</td>
<td>fuzzyR, waveslim, pomdp</td>
</tr>
</tbody>
</table>

*Table 11 List of tools required to implement POMDP and emotion detection*

5.2 User Interface (UI)

As of now, this application is developed in an offline mode within Android Studio IDE which produces even the UI as well. There are 4 different components which are described as follows, from the Section 4.5, Block K.
Different modules in the experimental setup application,

- First-half - 3D model, to discover the intention; to detect the emotion, and to display the emotion over the 3D model.
- Second half - Dialogue Management (DM) component where interaction between the user and an agent happens
- The edit text and three buttons - an Edit Text to write textual messages, first button - send button for textual-message, second button - record button for live detection emotion from the speech (voice), third button - record button to record speech to convert it to textual based messages on DM component
- Second half during the live recording session - It represents the probability of emotion which is detected from the voice at the user’s end

Screenshots of the GUI can be found in Appendix D: Textual and Speech-based emotions discovery from a user’s interaction. The working prototype (proof of concept) can be found as a video here\(^1\) and another example with different thought is found as a video is here\(^2\). A tiny video which shows the lip syncing with the speech on ECA model can be found here\(^3\). A speech recording, sentiment discovering demo can be found here\(^4\).

5.3 The project structure

The project structure is given in the Appendix E: Android Studio setup. All the java, XML, obj 3D files, etc. are represented. For the whole project folder, kindly follow the link\(^5\). Please note down, that environmental project setup requires knowledge of Android Studio IDE and how programming for Android coding work, so while setting up if you encounter any errors,

\(^1\) https://youtu.be/UbjucsBzTgU
\(^2\) https://youtu.be/z6F0yO_4UM8
\(^3\) https://youtu.be/lHHRbfcOAOo
\(^4\) https://youtu.be/CND_J32A3MM
\(^5\) https://drive.google.com/open?id=1y8JIIs7PLiBoPQB4hK0HfvfAgzXNGBh1
kindly contact the author of this thesis or follow the FAQs on respective Android development website.

5.4 Datasets
For the basic setup, the service data are used which was given by Mulpuri [1] and is available at the link\(^6\). To improve intentions and to improve the emotion detection accuracy, I have incorporated different datasets from different sources. The full sentence-based emotion detection can be found at the link\(^7\), which has 7000+ **lines** of conversation to train the model and to track the exact emotion, 1500+ emotion-related **words** were gathered which can be found at the link\(^8\). The domain specific brand ambassador-based customization can be developed for ECA using famous people’s human face as their alter ego as shown in Appendix F: Domain specific ECA customization using Facegen Software with Brand Ambassador’s Alter Ego.

5.5 User-Agent conversational chat setup
The conversation setup between the user and the agent was developed on python platform. The obtained results as in Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal are represented in the chart in Figure 24 with the sentiments analysis rewards as in Appendix H: User-Agent chat conversation: Sentiment Analysis rewards for **Online book search portal customization** example, from the Section 4.5, Block \(E \rightarrow G\). The Python, Android codes are available here\(^9\), but a person seeking a grant need to make a request when redirected to the Google drive page, then after your request is granted, the codes will be accessible.

---
\(^6\) https://drive.google.com/open?id=1S0rjk-nW_yanIxQ6S_obF9LVKXE4waVq
\(^7\) https://drive.google.com/open?id=1XO3dfXRHPpSGRUM0uW83M6z0b5oILJW27
\(^8\) https://drive.google.com/open?id=16ucMMiLbucFgn0EDM8ZtdyA1awqsRNvl
\(^9\) https://drive.google.com/open?id=19DU290wo3i-x5QsMEmEpE3kXjthf1ZnMj
Figure 24 Rewards chart: Chat conversation setup
Chapter 6: Simulation and Results

This chapter chronicles the work carried out. The datasets received from [1] are used to set up the basic environment of the Online Book Store customization portal. Additionally, the huge datasets (policy hand-crafted) can be used to train the model for the utmost customization.

From the basic setup, 4 tests have been executed but the result of only 1 test is shown as in the link\(^\text{10}\). These outputs are gathered from the Log data of the Android Studio IDE. As you can see, the knowledge level is Amateur. By knowing the knowledge level, we can get the belief state history and the policy. These results are obtained on 3 different machines using the Android Studio IDE. The 3 devices which are tested are listed below.

- Real Android Samsung A7 device
- Genymotion simulator
- Android Studio ADB Emulator, etc.

The 3D model is simulated as shown in Figure 25. There are 124 possible different expressions if applied only one emotion at a time. If two different expressions/emotions are combined, then there could be 15,376 possibilities and same goes with three at a time compound for 1,906,624 possibilities. The 3D model is generated as in Section 4.5, Block H & I → K.

\[\text{Figure 25 3D model}\]

\(^\text{10}\) [Link to Google Drive]
In another example, the interaction between the agent and the user is as shown below, representing the Section 4.5, Block A & B & D.

The Belief State History (newBelief), Number of Change Points (ncp) using Trend Analysis, and the Knowledge Level (k) as Policy.

6.1 Simulation design and process

- **Design**- Trained the knowledge level selectors with ontology and hand-crafted policies; Performing trend analysis over BSH to find Ncp; Pretended to be the different user each time while maintaining the fluctuations to categorize four different types of users using Ncp; Ncp is used to set the ground truth for the policy generation

- **Process**- Used the keywords present in the Ontology chart as in Figure 27, for the training part in the knowledge level selectors; the technological and knowledge level of the user should be used as one of the aspects when analysing the policy (from business management case study\(^\text{11}\)); Ncp is used to simulate the chat conversation

\(^{11}\) [Link to original source](https://drive.google.com/open?id=1240QQp-ERDJaPvIWIJJocfl2zFS1xhfK)
Figure 26 Ontology Chart
Figure 27 Ontology chart - partial
Ground truth- while training, the threshold is set for different users as \( n\% \) \((n = 0 \text{ to } 100)\) of times Ncp found. 0-25%- Expert | 25-50%- Professional | 50-75%- Amateur | 75-100%- Novice. If Ncp falls into any such category of threshold, then that becomes the policy for the user, for that loop of interaction. Then, the simulated results are compared with the hand-crafted policy and set threshold to match whether policy is improved or not!

Ontology chart- as show in Figure 27 the Ontology chart is used to customize the prototype system. The whole ontology chart is represented in Figure 26, and that can be implemented as well for the whole commercial application.

Based on the interaction and different iterations of the user-agent conversation in Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal, the belief state history database (BSH table for all 4 different users is listed in Appendix I: Belief State History (BSH) database information for different users) is recorded and then the trend analysis is performed over it, so that sharp points (minimum and maximum) can be found. As shown in the Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal, the sharp variation points can be represented: Figure 28 for 26 interaction iteration.
From the section 4.5, Block C, the trend analysis is performed on BSH and sharp points are found as in Figure 29.

The wavelet analysis transforms the signal and represents the signal using positive or negative values. A zero-crossing point between the values of detail coefficients represents the sharp variation points.

The sentiment analysis results (examples) are as shown below from the Section 4.5, Block E.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| A | **Hi, How are you?**  
   | neg: 0.0, neu: 1.0, pos: 0.0, |
| B | **I am fine, not bad! you say!**  
   | neg: 0.0, neu: 0.433, pos: 0.567, |
| C | **You know what you are good for nothing, I have no good reviews...**  
   | neg: 0.0, neu: 0.539, pos: 0.461, |
| D | **I am glad that you brought this up!!**  
   | neg: 0.0, neu: 0.626, pos: 0.374, |
| E | **Well, this is the most untidy thing I have seen urghhh..!!**  
   | neg: 0.0, neu: 0.77, pos: 0.23, |
| F | **I don't care, I don't have trust...**  
   | neg: 0.396, neu: 0.604, pos: 0.0, |
| G | **I don't feel good after this, this is so horrific..**  
   | neg: 0.231, neu: 0.769, pos: 0.0, |
Most automated sentiment analysis tools are shit.

neg: 0.375, neu: 0.625, pos: 0.0

VADER sentiment analysis is the shit.

neg: 0.0, neu: 0.556, pos: 0.444

Sentiment analysis has never been good.

neg: 0.325, neu: 0.675, pos: 0.0

Sentiment analysis with VADER has never been this good.

neg: 0.0, neu: 0.703, pos: 0.297

| H | Most automated sentiment analysis tools are shit.  
   | neg: 0.375, neu: 0.625, pos: 0.0, |
| I | VADER sentiment analysis is the shit.  
   | neg: 0.0, neu: 0.556, pos: 0.444, |
| J | Sentiment analysis has never been good.  
   | neg: 0.325, neu: 0.675, pos: 0.0, |
| K | Sentiment analysis with VADER has never been this good.  
   | neg: 0.0, neu: 0.703, pos: 0.297, |

Table 12 Examples of Sentiment Analysis of different sentences

The examples given in Table 12 can be represented in a chart in Figure 30.

![Average: Sentiment Analysis On Example Data](chart.png)

Figure 30 Sentiment Analysis Example data chart

The output using sentiment analysis and the fuzzy logic system using rewards, final emotions for the ECA can be generated as shown below using the example as in Section 4.5, Block E & F:

Most automated sentiment analysis tools are shit.

neg: 0.375, neu: 0.625, pos: 0.0

Policy: Expert

COCOM: Strategic, Tactical
Fuzzy RULE base selection based on Trend Analysis performed on Belief-state history using COCOM

<table>
<thead>
<tr>
<th>Sentiment Analysis Reward</th>
<th>Strategic</th>
<th>Tactical</th>
<th>Opportunistic</th>
<th>Scrambled</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Negative</strong></td>
<td>Disgust</td>
<td>Anger</td>
<td></td>
<td>Fear</td>
</tr>
<tr>
<td><strong>Neutral</strong></td>
<td>Fear</td>
<td>Sad</td>
<td>Surprise</td>
<td>Sad</td>
</tr>
<tr>
<td><strong>Positive</strong></td>
<td>Happy</td>
<td></td>
<td>Surprise</td>
<td></td>
</tr>
</tbody>
</table>

Table 13 FAMM: Fuzzy Associative Memory Matrix table using COCOM

<table>
<thead>
<tr>
<th>FAMM</th>
<th>Strategic</th>
<th>Tactical</th>
<th>Opportunistic</th>
<th>Scrambled</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Negative</strong></td>
<td>W1</td>
<td>W4</td>
<td>W7</td>
<td>W10</td>
</tr>
<tr>
<td><strong>Neutral</strong></td>
<td>W2</td>
<td>W5</td>
<td>W8</td>
<td>W11</td>
</tr>
<tr>
<td><strong>Positive</strong></td>
<td>W3</td>
<td>W6</td>
<td>W9</td>
<td>W12</td>
</tr>
</tbody>
</table>

Table 14 Weights considered for FAMM table

From the section 4.5, Block E & G & H → I, the final emotion for the ECA will be calculated as in Table 15.

<table>
<thead>
<tr>
<th>FAMM</th>
<th>Strategic</th>
<th>Tactical</th>
<th>Opportunistic</th>
<th>Scrambled</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Negative 0.375</strong></td>
<td>W1*Disgust</td>
<td>W4*Anger</td>
<td>W7*Anger</td>
<td>W10*Fear</td>
</tr>
<tr>
<td><strong>Neutral 0.625</strong></td>
<td>W2*Fear</td>
<td>W5*Sad</td>
<td>W8*Surprise</td>
<td>W11*Sad</td>
</tr>
<tr>
<td><strong>Positive 0.0</strong></td>
<td>W3*Happy</td>
<td>W6*Happy</td>
<td>W9*Surprise</td>
<td>W12*Surprise</td>
</tr>
</tbody>
</table>

From the section 4.5, Block E & G & H → I, the final emotion for the ECA will be calculated as in Table 15.

<table>
<thead>
<tr>
<th>FAMM</th>
<th>Strategic</th>
<th>Tactical</th>
<th>Opportunistic</th>
<th>Scrambled</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Negative 0.375</strong></td>
<td>W1*Disgust</td>
<td>W4*Anger</td>
<td>W7*Anger</td>
<td>W10*Fear</td>
</tr>
<tr>
<td><strong>Neutral 0.625</strong></td>
<td>W2*Fear</td>
<td>W5*Sad</td>
<td>W8*Surprise</td>
<td>W11*Sad</td>
</tr>
<tr>
<td><strong>Positive 0.0</strong></td>
<td>W3*Happy</td>
<td>W6*Happy</td>
<td>W9*Surprise</td>
<td>W12*Surprise</td>
</tr>
</tbody>
</table>

87
Table 15 Emotion decision taken based on COCOM table and weight table, to produce final emotion

Now, the probability of $W_n$, who has a higher weight (based on the Ncp and the COCOM rewards) will be affecting the final selection of the ECA emotion. In this case, $W_2$ or $W_5$ will be selected and respectively Fear or Sad emotion will be shown on the ECA’s face as the emotion. The matrix values can be calculated from the defuzzification step of the Fuzzy Logic System Section 4.5, Block G. The work of Fuzzy logic system was brought by the thesis by Kaur [2] and I am using those formulas to calculate the $W_n$ value, where $n$ varies from 1 to 12, for FAMM table. The equation for the same is given below.

$$X = \frac{\sum W_n \times FAMM_n}{\sum W_n}$$

*Equation 30 Defuzzification to generate crisp output*

where $W_n$ is the degree to which the $n$th rule matches the input data. We calculate the $W_n$ values using the membership function, it could be triangular or trapezoidal form. Thus, it can be constructed that the values of $W_n$ are the deciding factor in terms of generating final emotion in terms of deciding which final emotion the ECA will simulate.

<table>
<thead>
<tr>
<th>Fear</th>
<th>Sad</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Fear Image]</td>
<td>![Sad Image]</td>
</tr>
</tbody>
</table>

*Table 16 Final ECA emotion after crisp Fuzzy Logic System output*
The Q-Learning rewards are as given in Figure 31 which affects the decision-making process.

The Q-values are given in the Appendix J: Q-Learning: Q-values Rewards data.

We choose the action $a$ and state $s$ based on the Q-table we get. Initially every Q-value starts at 0. In the beginning as explained in 4.3.1 the epsilon rates will be higher until the exploration happens and suitable but random action is chosen, the logic is the agent doesn’t
know anything as such in terms of handling the uncertainty. As agent explores more, the rate decreases, which offers the 0 value until the last interaction with a few fluctuations in between as well, if the confidence measure fluctuates. As explained in 4.3, the perceptron-action-learning loop for each interaction, provides the Q-values to get the new state s’ based on the action a made using the reward r, from the section 4.5, Block J.

6.2 Policy improvement discussion

The policies were hand-crafted during the knowledge level training for the simulation purpose. Then the goal is to improve the policy using the self-optimization while the conversation last.

![Policy chart](image)

*Figure 32 Policy chart*

Henceforth, having the intention discovery for each interaction iteration, action is made by the agent in terms of policy, authenticated by the user. The Figure 32 represents the difference between the old and the new improved policy based on the data given in Appendix K: Policy improvement. Where, 1=Expert; 2=Professional; 3=Amateur; 4=Novice. As
shown, the last point represents the average of all the policies and it is proven that, Policy has been improved.

6.3 Results and Accuracy discussion

From section 4.6, we can analyze the results to see if the goal of the thesis was achieved, in terms of accuracy; dialogue length, improvement in the ECA emotion. The accuracy is calculated on average accuracy of 4 different kinds of knowledge levels which are, Expert, Professional, Amateur, and Novice; where the goal was to reach certain decision to resolve user’s goal in the decision-making process. Average dialogue length is measured using the different iterations over different 4 knowledge levels of the users. The improvement in the results of an ECA emotion can be concluded based on the user expressing fewer negative emotions and more neutral and positive emotions generation, as users will not accommodate the frequent anger; sad; fear; etc. emotions, therefore sentiment analysis helps in determining the more neutral when not the positive to leave enhancing impact on the user.

![Final results: representation of Accuracy; Dialogue length; Neutral/Positive intention discovery ratio](image)

*Figure 33 Accuracy calculation*

The accuracy is calculated based on the interaction iteration that how many times user was able to reach the goal and how long it took (dialogue length).
### Table 17 Results: Accuracy; Dialogue length; Neutral/Positive intention discovery ratio

<table>
<thead>
<tr>
<th>Users</th>
<th>Accuracy (in %)</th>
<th>Dialogue Length (in units)</th>
<th>Neutral/Positive Intention Discovery (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert</td>
<td>100</td>
<td>4</td>
<td>100</td>
</tr>
<tr>
<td>Professional</td>
<td>92</td>
<td>6</td>
<td>100</td>
</tr>
<tr>
<td>Amateur</td>
<td>85</td>
<td>8</td>
<td>87.5</td>
</tr>
<tr>
<td>Novice</td>
<td>68</td>
<td>14</td>
<td>92.86</td>
</tr>
<tr>
<td>Average</td>
<td>86.25</td>
<td>8</td>
<td>95.09</td>
</tr>
</tbody>
</table>

6.4 Discussion

As mentioned in the Table 17, when we traverse through different knowledge levels of policy selection, there is a fall in the accuracy measurements starting from expert to professional to amateur to novice. But, to note in the case of dialogue length it is complete opposite as
numbers of dialogue increases for the same traversing path. That way conclusion can be made that, accuracy and the dialogue length are proportional to each other. The intention discovery improvement ratio is calculated based on number of times positive or neutral emotions were generated.

6.5 Limitations

There are still challenges, user input in slang (lingo) will not offer good accuracy to reach optimal policy, as sentiment cannot be detected and misunderstanding might occur (to overcome this limitation, training can be done such a way that regional-slangs are understood); RL operates on trial-and-error while solving the optimal control aim, which generates uncertain consequences as actions in the environment are still under training; the optimal policy must be inferred by trial-and-error interaction with the environment, the only learning signal the agent receives is the reward; the observations of the agent depend on its actions and can contain strong temporal correlations; agents must deal with long-range time dependencies to handle the consequences of action which only materialize after many iterations of the environment which is known as a credit assignment problem.
Chapter 7: Conclusion and Future work

7.1 Conclusion

The following conclusions can be made based on the work done as follows,

Based upon the research, ECA is currently believed to be the best efficient medium for HCI. POMDP outperforms all the DM based domain’s approaches under the uncertainty and under the stochastic environment, using the knowledge level selectors which enhances the higher threshold out of training to get the optimal policy and its following action. To optimize the dialogue length, traditional approaches fail miserably as they are not intelligent enough to compete with the uncertainty present in human conversation. POMDP-based DM helps in improving intention discovery for ECA and eventually helps in improving policy using the Q-Learning rewards and sentiment analysis helps in understanding emotion detection from user input to decide goal-driven aim achievement with the help of human-like emotions on the 3D model face; where the rewards are considered and converted to the emotions using the fuzzy logic system which produces the crisp emotion using COCOM. Reinforcement Learning helps in learning optimal policy which helps in reducing dialogue length, making dialogue conversation smooth than the former.

7.2 Future work

This section discusses the future work that can be carried out is as follows:

- SDS can be incorporated on the future POMDP models.
- The best possible training datasets with over one million dialogues along with hand-crafted policy which can train supervised knowledge level selection for each single dialogue to solve credit assignment problem: requires man-power and an investment, can be referred to train the model to improve of the policy in the Reinforcement
Learning context. That helps in emotion detection in improving on a large scale to handle the unexpected uncertainty.

- Discovery of user intention will be the essential part of the work to enhance the policy selection over knowledge threshold which eventually satisfies the user, but the knowledge level training can be made such a way that, the domain-specific customization doesn’t affect the emotion detection from user and representing at agent’s end on 3D model.

- If possible, the full-fledged working commercial application can be developed from the proof of concept developed as part of this thesis. The policy should be trained using huge datasets to improve the scalability.

- Some limitations such as, sarcasm detection, slangs, can be trained to be handled with plethora of word embeddings and the relevance.

- Word embeddings can be visualized in a stochastic environment to plot the similar wordings in a corpus to cluster them, which can enhance the limitation of language-accent-barrier; locale-regional barrier.

- The audio can be improved as there is still a scope to make it sound like more humanish than the current robotic sound. Apart from that, verbal communication can be improved to make ECA lip sync the conversation with the printed text.
References


[Accessed 16 August 2018].


Application and Social Agents, Beijing, China, 2018.


[67] J. D. Williams, P. Poupart and S. Young, “Factored Partially Observable Markov Decision Processes for Dialogue Management,” in *4th Workshop on Knowledge and*


2018].


14 January 2019].


Appendices

Appendix A: HCI infrastructure model

Figure 34 A computer's Interface to a user

Figure 35 A user's interface to a computer
Appendix B: ECAs as a reflex agent

**Simple reflex agents**

Simple reflex agents act only based on the current percept, ignoring the rest of the percept history. The agent function is based on the condition-action rule: if condition then action. This agent function only succeeds when the environment is fully observable. Some reflex agents can also contain information on their current state which allows them to disregard conditions whose actuators are already triggered. Infinite loops are often unavoidable for simple reflex agents operating in partially observable environments. Note: If the agent can randomize its actions, it may be possible to escape from infinite loops.

![Diagram of a reflex agent](image)

**Model-based reflex agents**

A model-based agent can handle partially observable environments. Its current state is stored inside the agent maintaining structure which describes the part of the world which cannot be seen. This knowledge about "how the world works" is called a model of the world, hence the name "model-based agent". A model-based reflex agent should maintain some sort of internal model that depends on the percept history and thereby reflects at least some of the unobserved aspects of the current state. Percept history and impact of the action on the
environment can be determined by using an internal model. It then chooses an action in the same way as a reflex agent.

**Goal-based agents**

Goal-based agents further expand on the capabilities of the model-based agents, by using goal information. Goal information describes situations that are desirable. This allows the agent a way to choose among multiple possibilities, selecting the one which reaches a goal state. Search and planning are the subfields of artificial intelligence devoted to finding action sequences that achieve the agent's goals.
Utility-based agents

Goal-based agents only distinguish between goal states and non-goal states. It is possible to define a measure of how desirable a state is. This measure can be obtained using a utility function which maps a state to a measure of the utility of the state. A more general performance measure should allow a comparison of different world states according to exactly how happy they would make the agent. The term utility can be used to describe how "happy" the agent is. A rational utility-based agent chooses the action that maximizes the expected utility of the action outcomes - that is, what the agent expects to derive, on average, given the probabilities and utilities of each outcome. A utility-based agent has to model and keep track of its environment, tasks that have involved a great deal of research on perception, representation, reasoning, and learning.

Learning agents
Learning has the advantage that it allows the agents to initially operate in unknown environments and to become more competent than its initial knowledge alone might allow. The most important distinction is between the "learning element", which is responsible for making improvements, and the "performance element", which is responsible for selecting external actions. The learning element uses feedback from the "critic" on how the agent is doing and determines how the performance element should be modified to do better in the future. The performance element is what we have previously considered being the entire agent: it takes in precepts and decides on actions. The last component of the learning agent is the "problem generator". It is responsible for suggesting actions that will lead to new and informative experiences.
Appendix C: Different Avatars

Figure 36 ECA smile
Figure 37 ECA laugh
Figure 38 ECA nose wrinkle
Figure 39 ECA lip close

Figure 40 ECA closed smile
Figure 41 ECA wide smile
Figure 42 ECA small OH
Figure 43 ECA big OH

Figure 44 ECA smirk right
Figure 45 ECA surprise
Figure 46 ECA sneer right
Figure 47 ECA right smile
Figure 60 ECA walleye
Figure 61 ECA look up
Figure 62 ECA look right
Figure 63 ECA head up

Figure 64 ECA head left
Figure 65 ECA wink right eye
Figure 66 ECA blink left the eye
Figure 67 ECA suck

Figure 68 ECA mouth stretch
Figure 69 ECA tongue out
Figure 70 ECA depress
Figure 71 ECA normal
Appendix D: Textual and Speech-based emotions discovery from a user’s interaction

Figure 72 Splash/ welcome screen

Figure 73 First half 3D model, 2nd half
Dialogue Management manager

Figure 74 Edit Text to write a message, press blue button to send

Figure 75 DM module shows the interaction between user and agent- TEXT based
Figure 76 Speech recognized interaction scenario by pressing the last button of recording audio.

Figure 77 2nd button in red, directly to recognize the emotion from the live recording audio.

Figure 78 SAD emotion detected from the user's speech.

Figure 79 ANGER emotion detected from the user's speech.
Figure 80 NEUTRAL emotion detected from the user's speech

Figure 81 FEAR emotion detected from the user's speech
Appendix E: Android Studio setup

Figure 82 Project structure 1

Figure 83 Project structure 2

For detailed file structure kindly follow the link given above of project to supervise each file in detail. (I might have to grant the access, but as soon as I receive the notification, I will grant the access.)

Figure 84 Project structure 3
Appendix F: Domain specific ECA customization using Facegen Software with Brand Ambassador’s Alter Ego

Scarlet Johansson

Ruturaj Raval

Angelina Jolie

Jim Parsons

Vin Diesel

Robert Downey Jr.

Tom Cruise

Dwayne Johnson
Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal

Where, U: User; A: Agent; n: 1 to 26 (dialogue iterations)

<table>
<thead>
<tr>
<th>n</th>
<th>A: Hi, I am AVATAR. Welcome to use the interactive software customization tool. The functionality of Search a book is essential and it has been added as a required feature. As for the way of searching a book, how would you like your system to provide details?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>U: May i know what do you mean by providing details?</td>
</tr>
<tr>
<td>A:</td>
<td>Functionality get detailed info of a book is a sub-requirement of searching a book. Do you need it?</td>
</tr>
<tr>
<td>2</td>
<td>U: Yes i would love to know more about searching a book functionality.</td>
</tr>
<tr>
<td>A:</td>
<td>When you input the id of a book, get detailed info of a book, returns detailed information about the book, like publication info and contents. It has sub-requirements.</td>
</tr>
<tr>
<td>3</td>
<td>U: Yes, i prefer my system to provide detailed information of book.</td>
</tr>
<tr>
<td>A:</td>
<td>As this optional feature requires the functionality of get detailed info of a book, they have also been added to your system.</td>
</tr>
<tr>
<td>4</td>
<td>U: That's great.</td>
</tr>
<tr>
<td>A:</td>
<td>How would you like your system to allow user to search for a book?</td>
</tr>
<tr>
<td>5</td>
<td>U: It should be simply a keyword-based searching.</td>
</tr>
<tr>
<td>A:</td>
<td>To display the searching result, do you want the list of books being sorted?</td>
</tr>
<tr>
<td>6</td>
<td>U: Yes please add sort the books functionality.</td>
</tr>
<tr>
<td>A:</td>
<td>Functionality pick a book from a list has already been selected due to the decision on search a book.</td>
</tr>
<tr>
<td>7</td>
<td>U: What else is included apart from pick a book under search a book functionality?</td>
</tr>
<tr>
<td>A:</td>
<td>We offer quick view; detailed view; and sorting by title or latest or popular.</td>
</tr>
<tr>
<td>8</td>
<td>U: Which other searching functionality do you offer?</td>
</tr>
<tr>
<td>A:</td>
<td>We have basic search and advanced search and search by author or title or publication functionality.</td>
</tr>
<tr>
<td>9</td>
<td>U: And what do you offer in keyword matching module?</td>
</tr>
<tr>
<td>A:</td>
<td>We have broad match and exact match. That will be added by default as you</td>
</tr>
</tbody>
</table>
have added for keyword-based searching functionality.

10 U: Ok i understand. How can i add advanced search?
A: Functionality advanced search is a sub-requirement of searching a book. Do you want it?

11 U: Sure. Please add advanced search.
A: The quality constraint broad match is related to search in book keywords. Do you need it?

12 U: What?
A: Your response cannot be recognized. Please answer with the suggested response.

13 U: Okay.
A: The quality constraint broad match is related to search in book keywords. Do you need it?

14 U: Can you please explain?
A: When you search a book, not only books with keywords exactly like your inputs will be returned, but books with keywords similar to your inputs will also be returned. If you choose constraint broad match, quality constraint exact match cannot be selected.

15 U: Now it makes sense.
A: Do you want broad match?

16 U: No, i don't think so.
A: The quality constraint exact match is related to search in book keywords. Do you need it?

17 U: I guess, i will opt in for exact match
A: Do you want to add, manage shopping cart functionality?

18 U: What else do i get in cart functionality?
A: You get to add book to a cart, remove from a cart and list all the items in a cart.

19 U: Okay i want to have the functionality of managing cart.
A: Okay sure. Do you want to add the payment gateway?

20 U: What is payment gateway and how can it be useful?
A: Functionality payment gateway comes with high and low security to receive online payment for the book ordered online.

21 U: Okay add payment gateway.
A: Functionality get summary is part of place an order functionality, would you like
<table>
<thead>
<tr>
<th></th>
<th>U:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Yes add summary functionality.</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Do you want to add set delivery address functionality?</td>
<td>Yes.</td>
</tr>
<tr>
<td>24</td>
<td>Okay. Set payment information functionality has been added under payment gateway functionality.</td>
<td>That's good.</td>
</tr>
<tr>
<td>25</td>
<td>All the requirements have been evaluated. Would you like to make any changes?</td>
<td>No.</td>
</tr>
<tr>
<td>26</td>
<td>The customization process is complete. Thank you for your cooperation.</td>
<td>Quit</td>
</tr>
<tr>
<td></td>
<td>Thank you and see you soon.</td>
<td></td>
</tr>
</tbody>
</table>

*Table 19 Chat conversation setup User-Agent*
Appendix H: User-Agent chat conversation: Sentiment Analysis rewards

 où, n: 1 to 26 Interaction iterations

<table>
<thead>
<tr>
<th>n</th>
<th>Sentiment rewards</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>May i know what do you mean by providing details?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>2</td>
<td>Yes i would love to know more about searching a book functionality.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.7845, neg: 0.0, neu: 0.537, pos: 0.463,</td>
</tr>
<tr>
<td>3</td>
<td>Yes, i prefer my system to provide detailed information of book.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.4019, neg: 0.0, neu: 0.769, pos: 0.231,</td>
</tr>
<tr>
<td>4</td>
<td>That's great.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.6249, neg: 0.0, neu: 0.196, pos: 0.804,</td>
</tr>
<tr>
<td>5</td>
<td>It should be simply a keyword-based searching.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>6</td>
<td>Yes please add sort the books functionality.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.6124, neg: 0.0, neu: 0.5, pos: 0.5,</td>
</tr>
<tr>
<td>7</td>
<td>What else is included apart from pick a book under search a book functionality?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>8</td>
<td>Which other searching functionality do you offer?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>9</td>
<td>And what do you offer in keyword matching module?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>10</td>
<td>Ok i understand. How can i add advanced search?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.4939, neg: 0.0, neu: 0.543, pos: 0.457,</td>
</tr>
<tr>
<td>11</td>
<td>Sure. Please add advanced search.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.6808, neg: 0.0, neu: 0.233, pos: 0.767,</td>
</tr>
<tr>
<td>12</td>
<td>What?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,</td>
</tr>
<tr>
<td>13</td>
<td>Okay.</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.2263, neg: 0.0, neu: 0.0, pos: 1.0,</td>
</tr>
<tr>
<td>14</td>
<td>Can you please explain?</td>
</tr>
<tr>
<td></td>
<td>Compound: 0.3182, neg: 0.0, neu: 0.566, pos: 0.434,</td>
</tr>
<tr>
<td>15</td>
<td>Now it makes sense.</td>
</tr>
</tbody>
</table>
No, I don't think so.  

I guess, I will opt in for exact match.

What else do I get in cart functionality?

Okay I want to have the functionality of managing cart.

What is payment gateway and how can it be useful?

Okay add payment gateway.

Yes add summary functionality.

Yes.

That's good.

No.

Quit

Table 20 User-Agent interaction: Sentiment Analysis Rewards
Appendix I: Belief State History (BSH) database information for different users

The BSH information based on the interaction of the user and the agent as in Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal the BSH information is listed below.

<table>
<thead>
<tr>
<th>Interaction Iteration</th>
<th>BSH value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0.14</td>
</tr>
<tr>
<td>3</td>
<td>0.39</td>
</tr>
<tr>
<td>4</td>
<td>0.046</td>
</tr>
<tr>
<td>5</td>
<td>0.17</td>
</tr>
<tr>
<td>6</td>
<td>0.1</td>
</tr>
<tr>
<td>7</td>
<td>0.12</td>
</tr>
<tr>
<td>8</td>
<td>0.12</td>
</tr>
<tr>
<td>9</td>
<td>0.12</td>
</tr>
<tr>
<td>10</td>
<td>0.15</td>
</tr>
<tr>
<td>11</td>
<td>0.07</td>
</tr>
<tr>
<td>12</td>
<td>0.12</td>
</tr>
<tr>
<td>13</td>
<td>0.01</td>
</tr>
<tr>
<td>14</td>
<td>0.04</td>
</tr>
<tr>
<td>15</td>
<td>0.12</td>
</tr>
<tr>
<td>16</td>
<td>0.43</td>
</tr>
<tr>
<td>17</td>
<td>0.12</td>
</tr>
<tr>
<td>18</td>
<td>0.12</td>
</tr>
<tr>
<td>19</td>
<td>0.17</td>
</tr>
<tr>
<td>20</td>
<td>0.2</td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
</tr>
<tr>
<td>21</td>
<td>0.16</td>
</tr>
<tr>
<td>22</td>
<td>0.15</td>
</tr>
<tr>
<td>23</td>
<td>0.01</td>
</tr>
<tr>
<td>24</td>
<td>0.02</td>
</tr>
<tr>
<td>25</td>
<td>0.4</td>
</tr>
<tr>
<td>26</td>
<td>0.04</td>
</tr>
</tbody>
</table>

*Table 21 BSH database data for different users*
Appendix J: Q-Learning: Q-values Rewards data

This result is fetched using Q-Learning technique performed on the Atari game, so the basic idea is to use the Score value and the Q-table values to predict the next action based on the previous Q values, current Q values and the predicted Q values. And same approach can be used in this context of dialogue management.

<table>
<thead>
<tr>
<th>State</th>
<th>Action 1</th>
<th>Action 2</th>
<th>Action 3</th>
<th>Action 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.36</td>
<td>3.34</td>
<td>3.67</td>
<td>1.14</td>
</tr>
<tr>
<td>2</td>
<td>4.35</td>
<td>2.73</td>
<td>2.43</td>
<td>2.19</td>
</tr>
<tr>
<td>3</td>
<td>1.34</td>
<td>1.28</td>
<td>1.39</td>
<td>1.17</td>
</tr>
<tr>
<td>4</td>
<td>2.61</td>
<td>3.47</td>
<td>2.07</td>
<td>1.2</td>
</tr>
<tr>
<td>5</td>
<td>4.18</td>
<td>2.97</td>
<td>4.47</td>
<td>7.95</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7.92</td>
<td>1.38</td>
<td>7.94</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1.54</td>
<td>9.97</td>
<td>8.94</td>
<td>1.46</td>
</tr>
<tr>
<td>10</td>
<td>2.96</td>
<td>1.48</td>
<td>8.92</td>
<td>2.98</td>
</tr>
<tr>
<td>11</td>
<td>2.99</td>
<td>1.99</td>
<td>9.99</td>
<td>1.46</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>1.88</td>
<td>9.97</td>
<td>9.97</td>
<td>1.98</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1.99</td>
<td>1.99</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 22 Rewards: Q-values data*
Appendix K: Policy improvement

Based on the conversation as given in Appendix G: User-Agent chat conversation Ontology-based customization: Online Book Search portal, for 26 interaction iteration, the old (hand-crafted) and the improved policies are given in Table 23.

<table>
<thead>
<tr>
<th>Interaction iteration</th>
<th>Old Policy (hand-crafted)</th>
<th>Improved Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Novice</td>
<td>Professional</td>
</tr>
<tr>
<td>2</td>
<td>Professional</td>
<td>Expert</td>
</tr>
<tr>
<td>3</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>4</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>5</td>
<td>Expert</td>
<td>Professional</td>
</tr>
<tr>
<td>6</td>
<td>Novice</td>
<td>Expert</td>
</tr>
<tr>
<td>7</td>
<td>Novice</td>
<td>Amateur</td>
</tr>
<tr>
<td>8</td>
<td>Expert</td>
<td>Amateur</td>
</tr>
<tr>
<td>9</td>
<td>Amateur</td>
<td>Amateur</td>
</tr>
<tr>
<td>10</td>
<td>Amateur</td>
<td>Professional</td>
</tr>
<tr>
<td>11</td>
<td>Professional</td>
<td>Expert</td>
</tr>
<tr>
<td>12</td>
<td>Novice</td>
<td>Novice</td>
</tr>
<tr>
<td>13</td>
<td>Novice</td>
<td>Professional</td>
</tr>
<tr>
<td>14</td>
<td>Amateur</td>
<td>Novice</td>
</tr>
<tr>
<td>15</td>
<td>Professional</td>
<td>Expert</td>
</tr>
<tr>
<td>16</td>
<td>Amateur</td>
<td>Novice</td>
</tr>
<tr>
<td>17</td>
<td>Professional</td>
<td>Expert</td>
</tr>
<tr>
<td>18</td>
<td>Professional</td>
<td>Expert</td>
</tr>
<tr>
<td>19</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td></td>
<td>Amateur</td>
<td>Professional</td>
</tr>
<tr>
<td>----</td>
<td>---------</td>
<td>--------------</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>22</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>23</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>24</td>
<td>Expert</td>
<td>Expert</td>
</tr>
<tr>
<td>25</td>
<td>Expert</td>
<td>Professional</td>
</tr>
<tr>
<td>26</td>
<td>Professional</td>
<td>Professional</td>
</tr>
</tbody>
</table>

*Table 23 Policy table*
Vita Auctoris

Name: RUTURAJ Rajendrakumar RAVAL

Birth Place: Surat, Gujarat, India

Birth Year: 1994

Education: Bachelor of Engineering (B.E.), 08/2011-06/2015

Computer Engineering (CE)

C. K. Pithawalla College of Engineering and Technology, Surat, Gujarat, India

(affiliated under Gujarat Technological University, Gujarat, India (GTU))

Master of Science (MSc.), 01/2018-04/2019

Computer Science (CS)

School of Computer Science

University of Windsor, Windsor, Ontario, Canada (UoW)